Interactive Embedded Systems Learning using the Prairie Learn framework

Design Document

Team Number: 46
Client: Phillip Jones
Advisers: Phillip Jones
Team Members:
Benjamin Stroup
Caden Last
Cody Prochaska
Emmanuel Paz
Jack Kennedy
Ryan Bumann
Ryan Dela Merced
Team Email:
sdmay-46@iastate.edu
Team Website:
https://sdmay23-46.sd.ece.iastate.edu

Executive Summary

Development Standards & Practices Used

We are practicing an agile style of development along with heavy focus on test driven development. Overall, we are striving for our product to be easily adapted by any developer while keeping a high level of quality and sustainability.

Summary of Requirements

- Generate random interactive homework questions based on a specific problem type within the given parameters
- Automatically export/save the graded questions
- Date based accessibility for due date enforcement
- Different homework assignments
- Ability to maintain/update homework questions
- Online accessibility for a few hundred Users
- Must be secure; data must be safe and unable to be accessed by unauthorized persons.
- Store/export user and question data
- Full Stack web application
- Ability to be redeployed

Applicable Courses from Iowa State University Curriculum

CPRE 288 is applicable, because that is the class for which we are creating the questions.

COMS 309 is applicable because it taught us how to work on software in a group.

COMS 252 helped us learn how to use linux so we can work on the server.

CPRE 430 helped us learn enough networking to understand the networking necessary for the PrairieLearn web application.

New Skills/Knowledge acquired that was not taught in courses

We mostly all have previous experience programming in small personal projects so we’ve gotten experience from that. Furthermore, some of us have prior experience with Embedded Systems.

Table of Contents

1 Team

1.1 Team Members

1.2 Required Skill Sets for Your Project

(if feasible – tie them to the requirements)

1.3 Skill Sets covered by the Team

(for each skill, state which team member(s) cover it)

1.4 Project Management Style Adopted by the team

1.5 Initial Project Management Roles

2 Introduction

2.1 Problem Statement

2.2 Requirements & Constraints

2.3 Engineering Standards

2.4 Intended Users and Uses

3 Project Plan

3.1 Project Management/Tracking Procedures

3.2 Task Decomposition

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

3.4 Project Timeline/Schedule

3.5 Risks And Risk Management/Mitigation

3.6 Personnel Effort Requirements

3.7 Other Resource Requirements

4 Design

4.1 Design Context

4.1.1 Broader Context

4.1.2 User Needs

4.1.3 Prior Work/Solutions

4.1.4 Technical Complexity

4.2 Design Exploration

4.2.1 Design Decisions

4.2.2 Ideation

4.2.3 Decision-Making and Trade-Off

4.3 Proposed Design

4.3.1 Design Visual and Description

4.3.2 Functionality

4.3.3 Areas of Concern and Development

4.4 Technology Considerations
4.5 Design Analysis
4.6 Design Plan
5 Testing
5.1 Unit Testing
5.2 Interface Testing
5.3 Integration Testing
5.4 System Testing
5.5 Regression Testing
5.6 Acceptance Testing
5.7 Security Testing (if applicable)
5.8 Results
6 Implementation
7 Professionalism
7.1 Areas of Responsibility
7.2 Project Specific Professional Responsibility Areas
7.3 Most Applicable Professional Responsibility Area
8 Closing Material
8.1 Discussion
8.2 Conclusion
8.3 References
8.4 Appendices
8.4.1 Team Contract

List of figures/tables/symbols/definitions

1 Team

1.1 Team Members

- Ryan Dela Merced
- Ben Stroup
- Emmanuel Paz
- Caden Last
- Jack Kennedy
- Cody Prochaska
- Ryan Bumann

1.2 Required Skill Sets for Your Project

- Knowledge of CPRE 288 Concepts
- Creating dedicated server
- Python programming

1.3 Skill Sets covered by the Team

- Cybersecurity
- Python, C, HTML, Javascript
- Knowledge with reading data sheets
- Embedded Systems knowledge
- Server knowledge

1.4 Project Management Style Adopted by the team

- We used the agile management style

1.5 Initial Project Management Roles

- Ryan Dela Merced - Project Team Lead
- Cody Prochaska - Technical Team Lead

2 Introduction

2.1 Problem Statement

Our team is tasked with creating a way to generate interactive homework questions for CPRE 288 students. The homework generated should also include an automated grader for the students' submissions all while using the PrairieLearn Framework.

2.2 Intended Users and Uses

a. Who will use the product?

- i. CPRE 288 students, CPRE 288 instructors indirectly

b. Who will benefit?

- i. Mostly CPRE 288 instructors since they don't have to make problems and grade them.

c. Who cares

- i. CPRE 288 instructors for the same reason as before. Students wouldn't really care where the problems come from

d. Relevant Users/User Groups

- i. CPRE 288 students, CPRE 288 instructors
- ii. Only 2 groups because nobody else will be using it outside of CPRE 288

e. Persona

- i. CPRE 288 student
    1. Likely wants to just finish the homework fast
    2. Wants the questions to be easy
    3. Really wants the questions to at least make sense
- ii. CPRE 288 instructor
    1. Wants to save time creating problems and grading them
    2. Wants students to understand the material as much as possible
    3. Wants students to learn from the homework

f. Needs related to the project

i. Instructors need a way to generate problems so they don’t have to do it every Semester

ii. Instructors need a way to automatically grade problems so they don’t have to do the same thing over and over

iii. Instructors need a way to automatically put the grades in Canvas so they don’t have to type all the grades in over and over

iv. Students need the assignments to be easily understandable so that they aren’t constantly asking instructors, especially since the instructors did not make the problems and may not be of good help

g. How they might benefit

i. Instructors will save a lot of time from not having to create problems, grade them, and type the grades on canvas. Each semester this is used will increase the amount of time saved

ii. Students might benefit slightly from the questions likely being similarly formatted, etc. Making them easier to understand

.

2.3 Requirements & Constraints

Functional Requirements

- Generate random interactive homework questions based on a specific problem type within the given parameters
- Automatically export/save the graded questions
- Date based accessibility for due date enforcement
- Different homework assignments
- Ability to maintain/update homework questions
- Online accessibility for a few hundred Users
- Must be secure; data must be safe and unable to be accessed by unauthorized persons.
- Store/export user and question data
- Full Stack web application
- Ability to be redeployed

Resource Requirements

- Hosting server provided by ISU

Aesthetic Requirements

- easily navigable by students
- provide high quality images for the corresponding question

User Experimental Requirements

- TA’s need to be able to modify questions as needed

- Students need to be able to easily answer and input HW question answers into the system via multiple choice, short answer, and drop down choices.
- Unique user profiles
- Different User privileges
- Web based user interaction

UI Requirements

- Display questions including images when needed
- Input for the answers (text, multiple choice, etc.)

2.4 Engineering Standards

The main standard is that we are required to use the PrairieLearn framework. This is one of the main requirements as our client wants for everything to be done with the PrairieLearn framework. Furthermore, we are implementing standard coding standards such as readability, documentation, comments etc. This is important because most of our project will be done in code.

3 Project Plan

3.1 Project Management/Tracking Procedures

We plan on using the agile methodology. One of our main goals is to continuously roll out homework assignments with randomized parameters and overall migrate these assignments from static/on paper to fully be on PrairieLearn. Using sprints can help us stay on track with creating homeworks and focusing on the questions that will have randomized parameters. So each sprint will be dedicated to a homework assignment and testing. Towards the end, the focus will be on hosting our own PrairieLearn server on ISU dedicated servers.

Tracking process:

- GitLab boards
- Git
- Weekly meetings with our client
- WSR (weekly service review) documents to track progress up to that point and get everyone same page

3.2 Task Decomposition

Sprint 1: Get used to VM's and using PrairieLearn

Sprint 2: Get GitLab connected to prairielearn and make sure everybody was connected

Sprint 3: Begin creating homework 1 with help from TA and Professor

Sprint 4: Homework 1

Sprint 5: Homework 2

Sprint 6: Homework 3

Sprint 7: Homework 4

Sprint 8: Homework 5

Sprint 9: Homework 6

Sprint 10: Homework 7

Sprint 11: Homework 8

Sprint 12: Create users locally / Create user privilege tiers

Sprint 13: Get PL Production on ISU servers / Connect PL to Canvas

Sprint 14: Testing

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestones:

1. Get Prairie Learn running locally on each group members computer

1.1: Milestone Subtasks for each group member

1.1.1: Install docker

1.1.2: Pull PL in a docker container

1.1.3: run PL locally

1.1.4: connect PL locally

1.1.5: Pull gitlab course project in container

1.1.6: Mount gitlab course project in container

1.1.7: Run PL with course project

1.1.8: Create a temporary HW

1.1.9: create temporary questions

1.2: Progress Evaluation Criteria

1.2.1: percentage of group members with temporary questions

1.3: Completion Criteria

1.3.1: All group members have implement a running local PL with temporary HW that has temporary questions

2. Home Work 1

2.1: Milestone Subtasks

2.1.1: Create hw on PL course page

2.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

2.1.3: Implement question with verification

2.2: Progress Evaluation Criteria

2.2.1: percentage of questions implemented from course assignment

2.3: Completion Criteria

2.3.1: Demo each Hw question to professor

2.3.2: Professor accepts each question

2.3.3: Professor has no additional requests

3. Home Work 1

3.1: Milestone Subtasks

3.1.1: Create hw on PL course page

3.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

3.1.3: Implement question with verification

3.2: Progress Evaluation Criteria

3.2.1: percentage of questions implemented from course assignment

3.3: Completion Criteria

3.3.1: Demo each Hw question to professor

3.3.2: Professor accepts each question

3.3.3: Professor has no additional requests

4. Home Work 2

4.1: Milestone Subtasks

4.1.1: Create hw on PL course page

4.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

4.1.3: Implement question with verification

4.2: Progress Evaluation Criteria

4.2.1: percentage of questions implemented from course assignment

4.3: Completion Criteria

4.3.1: Demo each Hw question to professor

4.3.2: Professor accepts each question

4.3.3: Professor has no additional requests

5. Home Work 3

5.1: Milestone Subtasks

5.1.1: Create hw on PL course page

5.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

5.1.3: Implement question with verification

5.2: Progress Evaluation Criteria

5.2.1: percentage of questions implemented from course assignment

5.3: Completion Criteria

5.3.1: Demo each Hw question to professor

5.3.2: Professor accepts each question

5.3.3: Professor has no additional requests

6. Home Work 4

6.1: Milestone Subtasks

6.1.1: Create hw on PL course page

6.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

6.1.3: Implement question with verification

6.2: Progress Evaluation Criteria

6.2.1: percentage of questions implemented from course assignment

6.3: Completion Criteria

6.3.1: Demo each Hw question to professor

6.3.2: Professor accepts each question

6.3.3: Professor has no additional requests

7. Home Work 6

7.1: Milestone Subtasks

7.1.1: Create hw on PL course page

7.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

7.1.3: Implement question with verification

7.2: Progress Evaluation Criteria

7.2.1: percentage of questions implemented from course assignment

7.3: Completion Criteria

7.3.1: Demo each Hw question to professor

7.3.2: Professor accepts each question

7.3.3: Professor has no additional requests

8. Home Work 7

8.1: Milestone Subtasks

8.1.1: Create hw on PL course page

8.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

8.1.3: Implement question with verification

8.2: Progress Evaluation Criteria

8.2.1: percentage of questions implemented from course assignment

28.3: Completion Criteria

8.3.1: Demo each Hw question to professor

8.3.2: Professor accepts each question

8.3.3: Professor has no additional requests

9. Home Work 8

9.1: Milestone Subtasks

9.1.1: Create hw on PL course page

9.1.2: Professor approvaes each Hw question prototype based on the given course assignment question

9.1.3: Implement question with verification

9.2: Progress Evaluation Criteria

9.2.1: percentage of questions implemented from course assignment

9.3: Completion Criteria

9.3.1: Demo each Hw question to professor

9.3.2: Professor accepts each question

9.3.3: Professor has no additional requests

10. Create user accounts locally

10.1: Milestone Subtasks

10.1.1: Run PL on production locally

10.1.2: Create new user account

10.1.3: Login into User account

10.2: Progress Evaluation Criteria

10.2.1: percentage of steps completed towards running PL in production locally

10.3: Completion Criteria

10.3.1: Demo to professor

11. Have user privilege tiers

11.1: Milestone Subtasks

11.1.1: Set different permissions for created users

11.1.2: Set permission requirements for course editing

11.1.3: Test user permissions

11.2: Progress Evaluation Criteria

11.2.1: percentage permission implemented and tested

11.3: Completion Criteria

11.3.1: Demo to professor

12. Have PL output users grades on assignments

12.1: Milestone Subtasks

12.1.1: Output grade for each student user based on selected HW

12.2: Progress Evaluation Criteria

12.2.1: percentage of questions properly graded

12.3: Completion Criteria

12.3.1: Demo to professor

13. Professor can modify HWs

13.1: Milestone Subtasks

13.1.1: Admin users can edit each: hw, questions, due dates, grading, access and create new question and hws

13.2: Progress Evaluation Criteria

13.2.1: percentage of eidtiable option implemented

13.3: Completion Criteria

13.3.1: Demo to professor

14. Have production PL on ISU servers

14.1: Milestone Subtasks

14.1.1: Set up ISU server

14.1.2: Run PL on ISU in Production mode

14.2: Progress Evaluation Criteria

14.2.1: percentage of previous milestone functional

14.2.2: percentage of steps completed for running Prodction PL on ISU server

14.3: Completion Criteria

14.3.1: Demo to professor

15. Connect Canvas with PL on ISU servers

15.1: Milestone Subtasks

15.1.1: Get access to Canvas API

15.1.2: User accounts are connected to canvas

15.1.3: User grades are updated via canvas

15.2: Progress Evaluation Criteria

15.2.1: percentage of sub task completed

15.3: Completion Criteria

15.3.1: Demo to professor

3.4 Project Timeline/Schedule

Chart link:
https://docs.google.com/spreadsheets/d/1iyzIXPBDKoEGpdNPpuI7LrRjG8i2VHRceaZuIXnzNwI/edit?usp=sharing

3.5 Risks And Risk Management/Mitigation

Students may be able to cheat on the questions, Probability: <5%

Student information (grades) may be accessible to other students, Probability: <5%

Students may have admin rights, Probability: <5%

(as long as the server and PrairieLearn service is secure these won't happen)

Not many risks since we are running this on an already created software

3.6 Personnel Effort Requirements

	hours
Creating homeworks	8(?) homeworks * ~5 questions * 2 hrs = 80 hrs
Connecting to Canvas	10(?) hrs (getting permission + actually connecting)
Account creation/server hosting	5(?) hrs (likely easy just need to figure out how)

3.7 Other Resource Requirements

- PrairieLearn Docs
- PrairieLearn Slack
- ISU dedicated server
- Canvas API
- TA/professor input

4 Design

4.1 Design Context

4.1.1 Broader Context

Area	Description	Examples
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities)	Cybersecurity is a rising concern of many people. We will have a secure application for the users and their accounts.

Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	Make sure that students actually learn the content for the class and don't cheat their way through it.
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	Make sure the server doesn't take up too much energy. Eliminates use of paper for the course and helps stop deforestation.
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	Make sure the server doesn't take up too many resources that would require more money than needed.

4.1.2 Prior Work/Solutions

Similar Products:

Zybooks and Canvas Quizzes are similar products. They are autograding homework and quizzes. Also other prairielearn classes.

Online classes or textbooks often have accompanying software that also contains online autograded materials. One can expect there to be an online provider, pre existing generated assignments, assignment due dates and visibility properties, assessment functionality, user profiles or account, different user privileges, grading exporting, and technical assistance.

Advantages/shortcomings:

One of the advantages is that PrairieLearn provides an example class and some examples of how a question is set up.

PrairieLearn has good documentation for setting up a new project and iterating upon it. PrairieLearn also has an active community for technical assistance. PrairieLearn has its own hosting services that include distribution, deployment, data storage, and tight security. PrairieLearn allows the instructor to create their own questions and define how they are graded. This is an advantage because the instructor can tailor the assignment and assessments for the class.

One shortcoming of working from previous prairielearn classes is that there is not much documentation or ways of seeing how the other the classes work.

A Lot of the shortcomings stem from not using PrairieLearn's hosting services as native run applications are not supported. This is because the client does not want to pay for PrairieLearn's hosting services.

Cites:

https://www.prairielearn.org/

https://canvas.iastate.edu/

https://www.zybooks.com/

PrairieLearn allows the instructor to create their own questions and define how they are graded. This is an advantage because the instructor can tailor the assignment and assessments for the class. PrairieLearn can be hosted natively thus not requiring the payment for hosting.

4.1.3 Technical Complexity

To allow for a wide range of access the application will be accessible from the web. We are responsible for ensuring the servers public or university specific availability through the internet. The application will require the storage of user account information. This will need a database and security for sensitive information. The professor wants to be able to edit the assignments. We need to document the editing process for other teams and make a modular design to support it. The application needs to be secure so students can not find answers in the front end. We need to build a front end application capable of displaying questions and gathering user input that is dynamic and changes.

4.2 Design Exploration

4.2.1 Design Decisions

We decided to host the PrairieLearn server on our own instead of using PrairieLearn's paid hosting option

We are going with simple questions to allow for a greater variety of questions. This also speeds up the development of questions. The client can also see more prototype questions and iterate on them. The UI for the questions will be in html, and either JS or python scripts. This is a simple methods that has a lot of information available leading to an easy development.

We will be running PrairieLearn through a microservice tool such as docker. This simplifies the deployment and building of the project.

We are going with the database postgres that is built into the PrairieLearn container. This eliminates a lot of margins of error that could lead to a security risk and speeds up development time.

4.2.2 Ideation

Design Decision: data storage implementation

1. PrairieLearn container's native database implementation
2. Official ISU database servers
3. Deploy a MySQL server
4. Integrate a 3rd party database service
5. Canvas

Design Decision: Hosting

1. Host our own PrairieLearn server
2. Let PrairieLearn host our course

4.2.3 Decision-Making and Trade-Off

When deciding where we wanted to host our course, we wanted to make sure that it was time and cost efficient. After also deciding that we want a project to be long term, easily manageable and be able to oversee everything. Then receiving information from our client that PrairieLearn will charge for hosting a course due to demand, we decided on cost efficiency being more important. So ultimately we decided to host our own PrairieLearn server which will take more time to configure and deploy but in the long run not cost us anything and be able to be independent from the PraireLearns' servers if any issues arise.

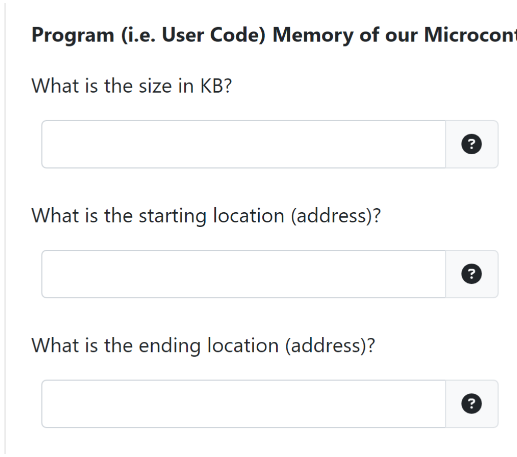
4.3 Proposed Design

4.3.1 Overview

We are creating a class in the PrairieLearn Framework in which the professor of the CPRE 288 class can create homeworks, quizzes, and exams that are automatically graded. This is broken down into the class instance, for example: CPRE288 Fall2022. Then into assignment types, which would be homeworks, quizzes and exams. Lastly broken into questions which would hold the content of the course and test the students on their knowledge. This application will be hosted on an ISU server and available for students in the class to access.

4.3.2 Detailed Design and Visual(s)

We will be using Prairielearn which is a software that is made for creating and grading homework assignments. For each question there is a generated json, python, and html file. We modify those files to create questions relating to each homework assignment given to us by the professor. That is all that there is to the homework assignments then we repeat for each question. It will look something like this.



Program (i.e. User Code) Memory of our Microcontroller

What is the size in KB?

What is the starting location (address)?

What is the ending location (address)?

Then we host it on an ISU server.

4.3.3 Functionality

The user will be a CPRE 288 student. They will be able to access their homework assignments, exams, etc. As students submit their homework assignments, we will set up PrairieLearn to automatically grade and turn in their assignments to canvas. Students will be able to look at their grades for each assignment and be able to redo questions that were incorrect (if assignment is still available).

4.3.4 Areas of Concern and Development

Our current design fits well with its users needs and requirements. Our primary concerns right now are the user profiles and privileges as well as getting a hold of a server to host everything on. Our plans to solve these are researching PrairieLearn documentation on handling users and talking with our client about getting a hosting server set up. Some questions might include figuring out the necessary auto grading parts that will be used in each homework.

4.4 Technology Considerations

The technologies used mainly consist of PrairieLearn and its internal tools, python and HTML for the auto grading and web page, and docker for running it inside of a portable container. An alternative to using PrairieLearn for automatic grading is using canvas quizzes instead. The strength of PrairieLearn and therefore weakness of canvas is that PrairieLearn can have students submit code to be compiled and have tests run for grading. Canvas doesn't have this function and it will be very useful in the later homeworks in the class.

4.5 Design Analysis

So far we have implemented the class instance and the homeworks set up in PrairieLearn. Next we are going to be setting up the server and putting the application onto the server before continuing the work. We are also developing the questions for the homework questions. Eventually when we are finished with the homeworks we will look into if our shareholder wants any additional features on the application like exams or quizzes. Lastly we will try to integrate our application into canvas so it will automatically update the student grades with their scores.

5 Testing

5.1 Unit Testing

Each individual homework question is tested based on the inputs and the expected outputs. This testing is all done manually for each problem as each problem is different. The extensiveness of the test depends mostly on if the problem is parameterizable or static.

5.2 Interface Testing

The PrairieLearn framework handles most of the interface communication between modules. So we have had little to no testing in this area.

5.3 Integration Testing

The critical parts of our project are making sure the automated grading and question randomization work reliably and correctly. This fits our user's needs and lets us make sure the questions are being created correctly. This is being tested by creating a few variants and seeing if any errors come up, and if they do, they are fixed. The only tools we would use during this would be PrairieLearn itself.

5.4 System Testing

Our system testing is pretty simple since the only thing that we have to worry about is getting the server running with prairie learn on it. Prairie learn has everything else handled so once we make sure that it is up on the server with ISU authentication then we should be good.

5.5 Regression Testing

Our regression tests go with our unit tests and we will just have to double check them to make sure that they still work after we add anything. They are pretty straightforward and just need to make sure that each question works.

5.6 Acceptance Testing

Since PrairieLearn handles most of the testing from the framework by itself, it will give us a good estimate of the problem so the problem won't affect the homeworks on the production side. If there is nothing wrong with the code then PrairieLearn will run fine and accept it no matter the outcome.

5.7 Security Testing (if applicable)

5.8 Results

The results from our testing is that if something is wrong in our code for the homeworks information, question information, or questions logic then PrairieLearn will give us an error screen with what is wrong and why. This is very helpful as we are making many variants to complicated questions and something unexpected could happen with just one of the many in our pool of questions. I do not think that our testing design is the greatest that it could be, but because of our restriction of using PrairieLearn and because it handles so many errors internally, there is not much room for us to make a better design.

6 Implementation

See Design Section.

7 Professional Responsibility

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Responsibility

Work Competence

- IEEE - Talks more about having work that is technically competent and that the people are qualified to complete the work that is assigned to them. Work also must be of high quality.

Financial Responsibility

- ACM - They mention that "Not engage in deceptive financial practices such as bribery, double billing, or other improper financial practices." This differs from the NSPE version

Communication Honesty

- ACM - They state that "Computing professionals should respect the right of those involved to transparent communication about the project. Professionals should be cognizant of any serious negative consequences affecting any stakeholder that may result from poor quality work and should resist inducements to neglect this responsibility." Somewhat similar to NSPE but they mention that anyone involved should be cognizant of major negative consequences.

Health, Safety, Well-Being

- IEEE - IEEE states that "to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment;" This differs to NSPE because they mention sustainability and ethical designs.

Property Ownership

- SE - It states that "Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent." This differs from NSPE because they state that use of property that is not yours should be authorized.

Sustainability

- ACM - States that "human well-being requires a safe natural environment. Therefore, computing professionals should promote environmental sustainability both locally and globally." They differ because they mention a safe natural environment.

Social Responsibility

- IEEE - They state that "to improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems;" this differs due to the part that states to help improve individuals and society.

7.2 Project Specific Professional Responsibility Areas

Work Competence - HIGH

- Currently we are all working at a high level and producing our project in an effective and efficient way.

Financial Responsibility - LOW

- Our project doesn't really have any financial aspect involved.

Communication Honesty - HIGH

- As a team, we are communicating at a high level and are able to be honest with each other to ensure we are all on the same page with nobody left behind.

Health, Safety, Well-Being - LOW

- Our project does not really endanger anyone but something that should be looked upon is that our data is secure.

Property Ownership - MEDIUM

- Everyone is doing their part to contribute to the project but we can put more priority at making sure that we are taking credit for what we have done individually.

Sustainability - LOW

- Our project doesn't have any environmental impacts so this is of low priority for our project.

Social Responsibility - HIGH

- Our project is going to be created for others so currently it is of high importance to make sure our project is benefiting the community and whoever is going to use it in the future.

7.3 Most Applicable Professional Responsibility Area

One area of professional responsibility that is important for our project and that our team has demonstrated a high level of proficiency in would be work competence. I think that as a team, we have been able to provide work that is of high quality, professional, and on time. To elaborate, our team has already created work of high quality that has been demonstrated to our project advisor.

8 Closing Material

8.1 Discussion

The preliminary results of our project so far are meeting the requirements as we are successfully able to generate randomized homework questions in the Prairie Learn framework. Furthermore, these questions are able to be automatically graded. We are currently continuing to develop homework questions and are confident in our ability to continue developing.

8.2 Conclusion

The current stage of our project is on track to be demod by actual TAs of CPRE 288 to see if the standards are met. Specifically, the requirement of having PrairieLearn hosted on Iowa State infrastructure and be secured is met. It is currently being finalized and documented. The requirement for the homework assignments is currently progressing smoothly. Some questions may be tougher to automate parameters for future assignments but we are making steady progress.

8.3 References

8.4 Appendices

Documentation used:

https://prairielearn.readthedocs.io/en/latest/

8.4.1 Team Contract

Team Name: Senior Design 46

Team Members:

1) Benjamin Stroup 2) Caden Last

3) Cody Prochaska 4) Emmanuel Paz

5) Jack Kennedy 6) Ryan Bumann

7) Ryan Dela Merced

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

Tuesday, 2:15 PM, Weekly

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

Online via Discord

3. Decision-making policy (e.g., consensus, majority vote):

Majority Vote

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be

shared/archived):

Shared document with the team

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

Every member is expected to attend all meetings unless emergency

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

Complete all assignments and meet all deadlines on time

3. Expected level of communication with other team members:

Communicate frequently through discord

4. Expected level of commitment to team decisions and tasks:

We want every member to be committed and have a say in the decisions being made

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):

Each member will be taking responsibility for everything, and we will all have equal parts in leading the team.

2. Strategies for supporting and guiding the work of all team members:

Talk about our progress in meetings and communicating well with each other

3. Strategies for recognizing the contributions of all team members:

Tracking progress per each team member using a shared document

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

Emmanuel, Caden, Jack, and Ben are software engineering majors and we will do the bulk of the coding. Ryan, Cody and Ryan have experience with CPRE 288. Cody also has experience with security.

2. Strategies for encouraging and supporting contributions and ideas from all team members:

Team members that have contributed poorly will get a poor team evaluation at the end of the semester.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

Team members will be expected to communicate their problems and the team as a whole will be responsible for resolving any issues that come up.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

Have a full conceptual design ready by the end of the semester

2. Strategies for planning and assigning individual and team work:

Team members will be splitting work evenly between everyone.

3. Strategies for keeping on task:

Teammates will motivate each other and keep each other on task to make sure deadline is not missed.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

We will talk to the member that broke the contract and get them back on track.

2. What will your team do if the infractions continue?

We will communicate to the professor, and he can decide what happens then.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Benjamin Stroup	DATE 9/8/22
2) Caden Last	DATE 9/8/22
3) Cody Prochaska	DATE 9/8/22
4) Emmanuel Paz	DATE 9/8/22
5) Ryan Bumann	DATE 9/8/22
6) Jack Kennedy	DATE 9/8/22
7) Ryan Dela Merced	DATE 9/8/22