Project 3: Building an Auto Grading System in PrairieLearn.

Emiel van der Poel
Prajeet Didden
Luis Lucio
Siqiao Yuan

Our Client

- Dr. Ramon Lawrence
  - Computer Science Professor at The University of British Columbia.
  - Focused research on Database Systems.
  - Introduction to Databases Instructor.
  - Plans to be using this project in the upcoming semester.

The Problem

Auto ER

- Scalability

The existing system for solving database design questions is currently only able to host ER design questions

- Usability

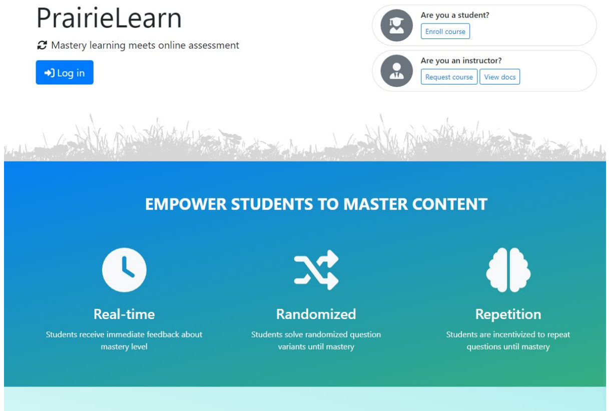
Cannot view student grades easily. No menu for questions. Strong knowledge of SQL is required to create new questions.

- Accessibility

Login to system is done through creating of user accounts from an excel sheet. Students are given a username and password.

What is PrairieLearn?

PrairieLearn is a problem-driven learning system for creating homeworks and tests. Questions can be randomized and autograded, thus making it powerful learning tool.



PrairieLearn

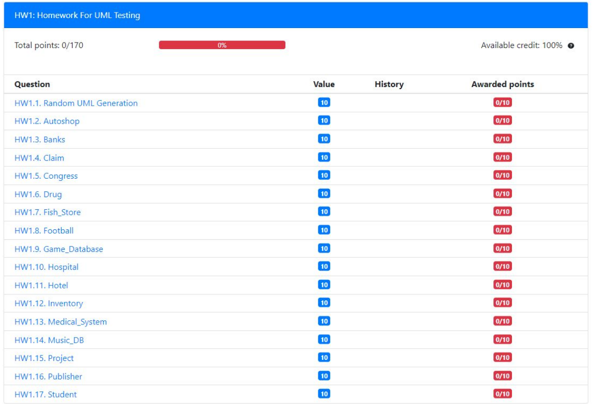
Mastery learning meets online assessment

EMPOWER STUDENTS TO MASTER CONTENT

Real-time
Students receive immediate feedback about mastery level

Randomized
Students solve randomized question variants until mastery

Repetition
Students are incentivized to repeat questions until mastery



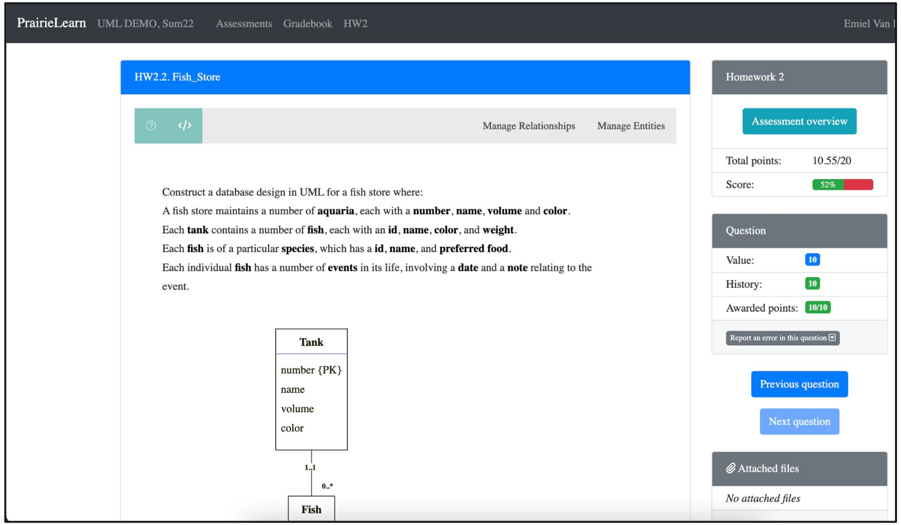
HW1: Homework For UML Testing

Total points: 0/170 — 0% — Available credit: 100%

Question	Value	History	Awarded points
HW1.1. Random UML Generation	10		0/10
HW1.2. Autoshop	10		0/10
HW1.3. Banks	10		0/10
HW1.4. Claim	10		0/10
HW1.5. Congress	10		0/10
HW1.6. Drug	10		0/10
HW1.7. Fish_Store	10		0/10
HW1.8. Football	10		0/10
HW1.9. Game_Database	10		0/10
HW1.10. Hospital	10		0/10
HW1.11. Hotel	10		0/10
HW1.12. Inventory	10		0/10
HW1.13. Medical_System	10		0/10
HW1.14. Music_DB	10		0/10
HW1.15. Project	10		0/10
HW1.16. Publisher	10		0/10
HW1.17. Student	10		0/10

What is our Project?

Goal: Integrate AutoER into PrairieLearn and deploy on a live server.



Construct a database design in UML for a fish store where:

A fish store maintains a number of **aquaria**, each with a **number**, **name**, **volume** and **color**.

Each **tank** contains a number of **fish**, each with an **id**, **name**, **color**, and **weight**.

Each **fish** is of a particular **species**, which has a **id**, **name**, and **preferred food**.

Each individual **fish** has a number of **events** in its life, involving a **date** and a **note** relating to the event.

Total points: 10.55/20

Score: 52%

Value: 10

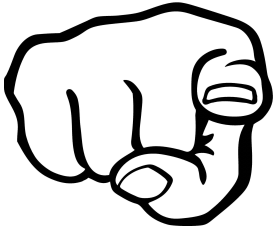
History: 10

Awarded points: 10/10

Target Users

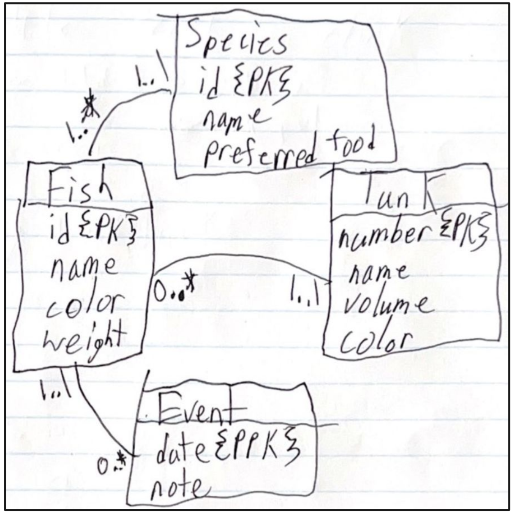
- Academic World
  - Instructors
  - Teaching Assistants
  - Students
- Any User looking to practice ER design questions

You



Usage Scenario

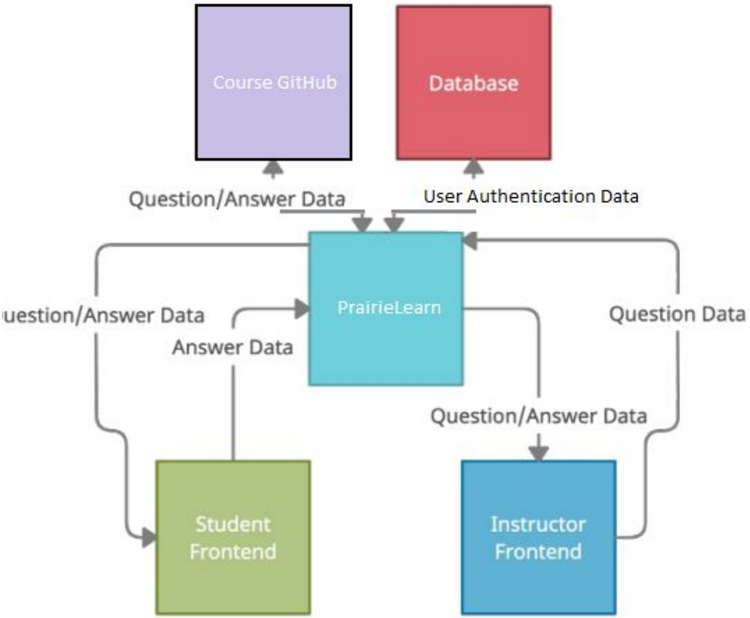
- Introduction to Databases Course.
    - Requires drawing ER design questions.
    - Very difficult to test and practice these questions as it requires drawing.
- This enables students to practice in an easy format.
- This enables instructors to provide an easy testable environment.
- Easier for grading of these questions as they are not hand drawn.

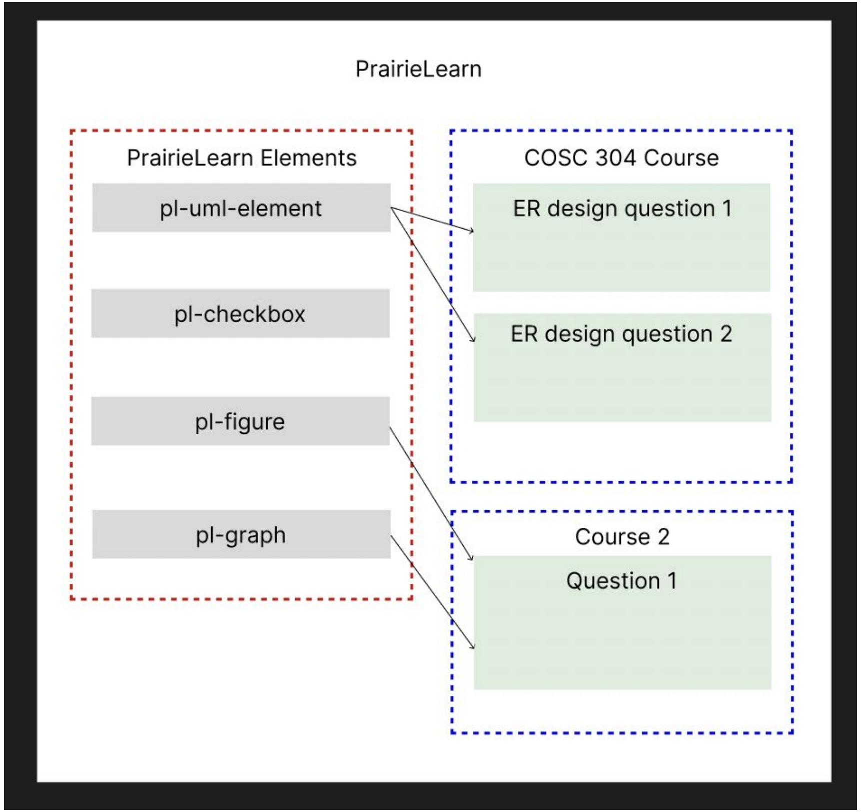


Requirements

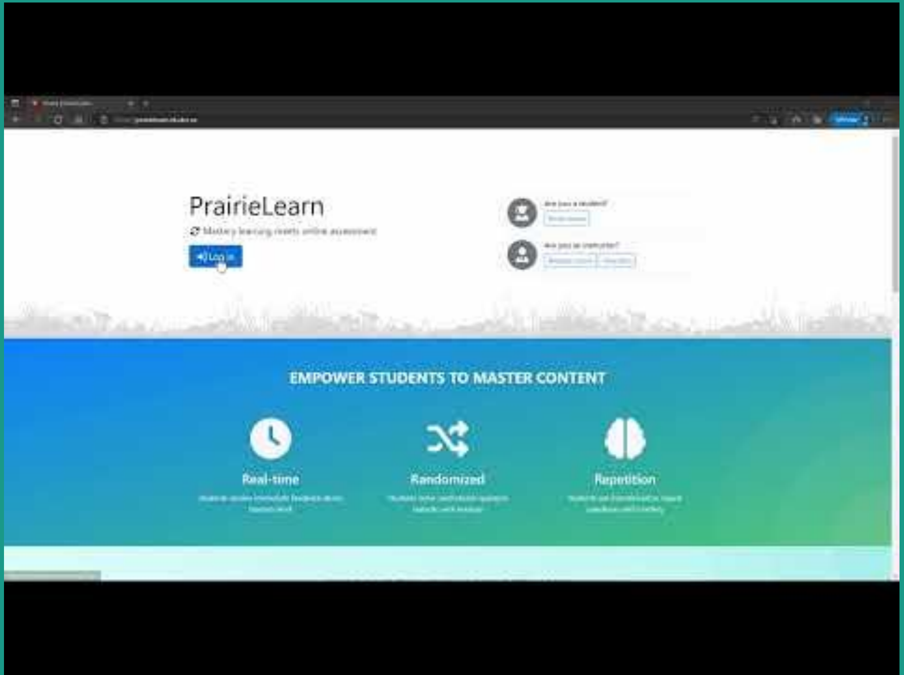
1. Bring ER design question into PrairieLearn.
2. Convert ER question to be a custom PrairieLearn element for easy modification and creation.
3. Dockerize PrairieLearn.
4. Create UBC wide self hosted PrairieLearn server.

System Architecture





Video Demonstration



PrairieLearn

Mastery learning meets online assessment

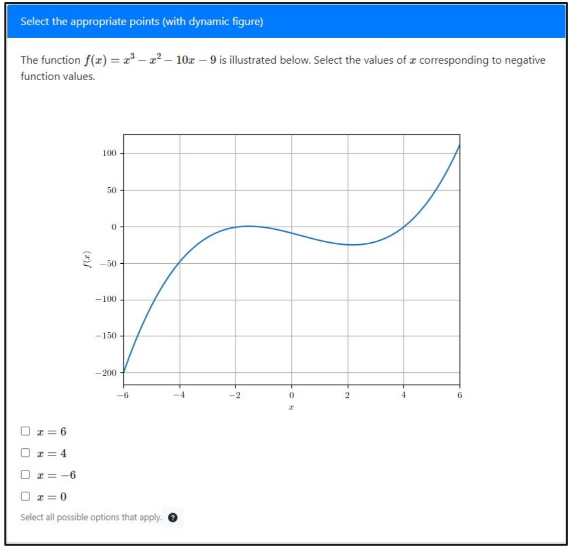
The Solution

Moving to PrairieLearn from the original system has brought many benefits both teacher and students users in regards to the following:

- Scalability
- Usability
- Accessibility

Key Feature

Compared to the old system (AutoEd), PrairieLearn supports multiple question types.



Select the appropriate points (with dynamic figure)

The function $f(x) = x^3 - x^2 - 10x - 9$ is illustrated below. Select the values of x corresponding to negative function values.

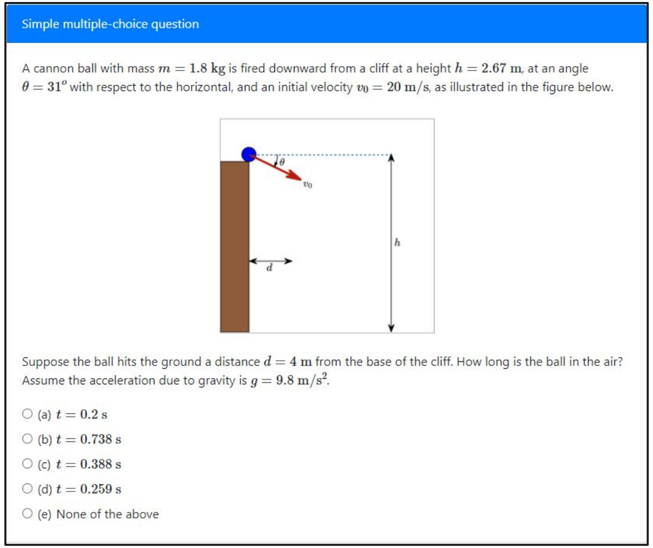
☐ $x = 6$

☐ $x = 4$

☐ $x = -6$

☐ $x = 0$

Select all possible options that apply.



Simple multiple-choice question

A cannon ball with mass $m = 1.8$ kg is fired downward from a cliff at a height $h = 2.67$ m, at an angle $\theta = 31^o$ with respect to the horizontal, and an initial velocity $v_0 = 20$ m/s, as illustrated in the figure below.

Suppose the ball hits the ground a distance $d = 4$ m from the base of the cliff. How long is the ball in the air? Assume the acceleration due to gravity is $g = 9.8$ m/s^2.

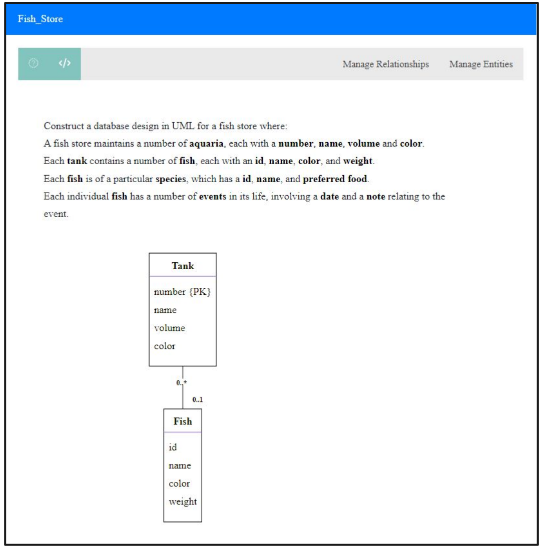
○ (a) $t = 0.2$ s

○ (b) $t = 0.738$ s

○ (c) $t = 0.388$ s

○ (d) $t = 0.259$ s

○ (e) None of the above



Fish_Store

Construct a database design in UML for a fish store where:

A fish store maintains a number of **aquaria**, each with a **number**, **name**, **volume** and **color**.

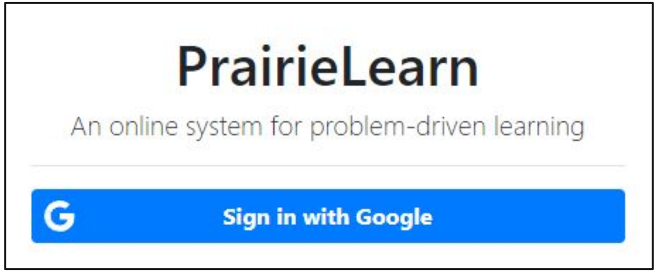
Each **tank** contains a number of **fish**, each with an **id**, **name**, **color**, and **weight**.

Each **fish** is of a particular **species**, which has a **id**, **name**, and **preferred food**.

Each individual **fish** has a number of **events** in its life, involving a **date** and a **note** relating to the event.

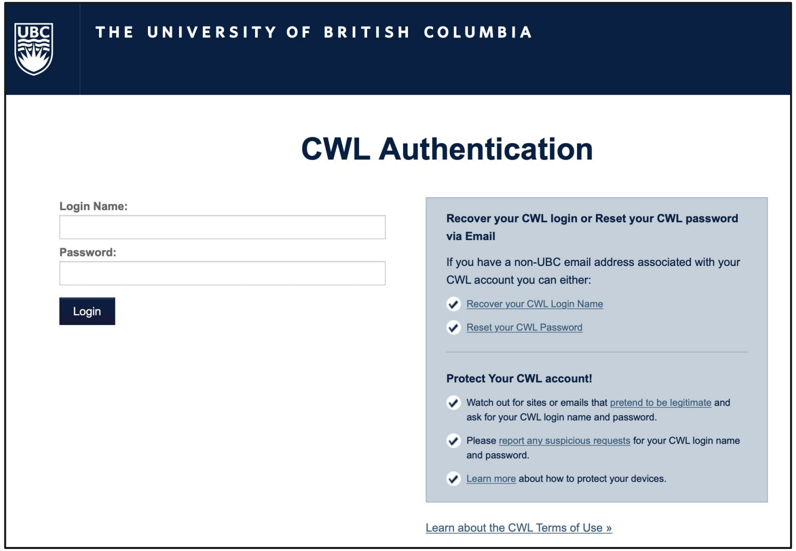
Key Feature

Students can sign in using Google or with CWL (coming soon)



PrairieLearn

An online system for problem-driven learning



THE UNIVERSITY OF BRITISH COLUMBIA

CWL Authentication

Login Name:

Password:

Recover your CWL login or Reset your CWL password via Email

If you have a non-UBC email address associated with your CWL account you can either:

- ✓ Recover your CWL Login Name
- ✓ Reset your CWL Password

Protect Your CWL account!

- ✓ Watch out for sites or emails that pretend to be legitimate and ask for your CWL login name and password.
- ✓ Please report any suspicious requests for your CWL login name and password.
- ✓ Learn more about how to protect your devices.

Learn about the CWL Terms of Use »

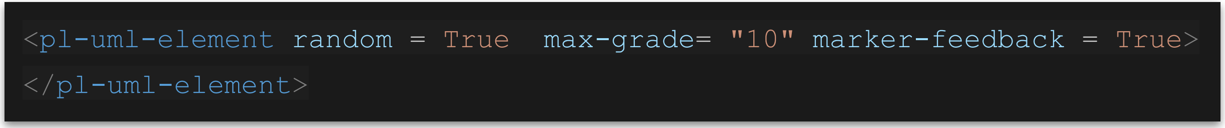
What is an element?

- Professors can create questions simply through the use of elements.
- PrairieLearn comes equipped with many useful elements out of the box such as multiple choice elements.
- PrairieLearn is open source and has a large active community, it is only natural to see many new elements emerge as time passes.

Our Element

Our group was tasked with creating an element, which would allow a professor to easily create an ER question through the use of basic html and markdown, which would then be auto formatted into the ER question.

For example, to create a randomized question is as simple as:

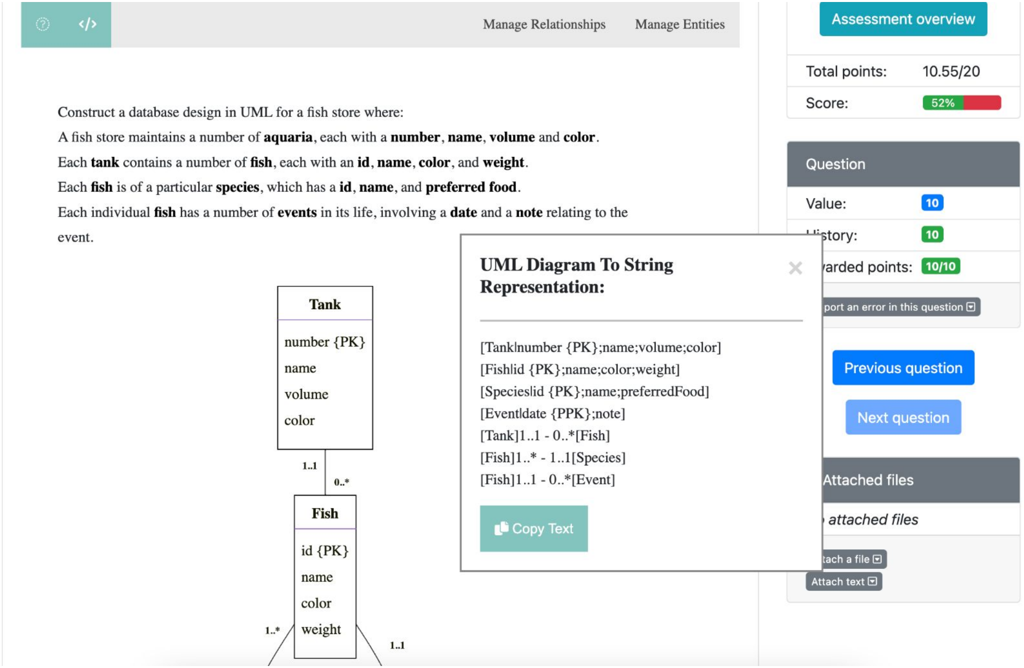


```
<pl-uml-element random = True  max-grade= "10" marker-feedback = True>
</pl-uml-element>
```

Our Element

Things get a bit more interesting when we want to create a set question:

Key Feature



Construct a database design in UML for a fish store where:
A fish store maintains a number of **aquaria**, each with a **number**, **name**, **volume** and **color**.
Each **tank** contains a number of **fish**, each with an **id**, **name**, **color**, and **weight**.
Each **fish** is of a particular **species**, which has a **id**, **name**, and **preferred food**.
Each individual **fish** has a number of **events** in its life, involving a **date** and a **note** relating to the event.

UML Diagram To String Representation:

```
[Tank|number {PK};name;volume;color]
[Fish|id {PK};name;color;weight]
[Species|id {PK};name;preferredFood]
[Event|date {PPK};note]
[Tank]1..1 - 0..*[Fish]
[Fish]1..* - 1..1[Species]
[Fish]1..1 - 0..*[Event]
```

Total points: 10.55/20
Score: 52%

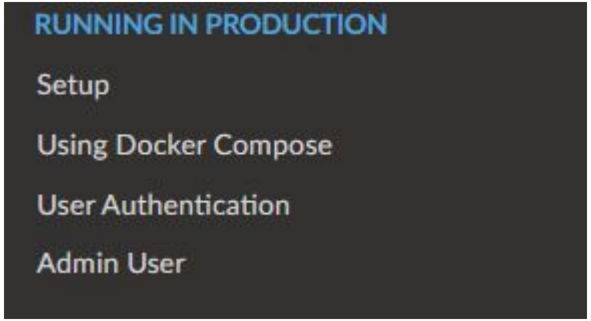
Value: 10
istory: 10
arded points: 10/10

attached files

Major Challenge

- PrairieLearn was designed to run natively, and was not dockerized for production.
- The team spent a large amount of time reworking the docker files and writing documentation for running the system in production on docker.

Dockerized PrairieLearn Code and Documentation Merged Into Codebase!!!



RUNNING IN PRODUCTION

Setup

Using Docker Compose

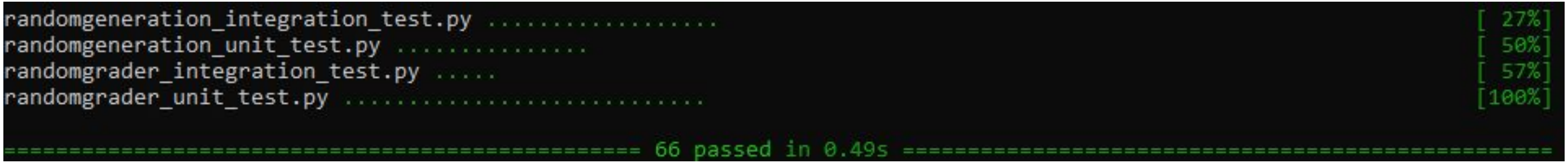
User Authentication

Admin User

Testing

- 8.6k+ total tests implemented in PrairieLearn

- 66 total tests written for our custom ER element
  - **randomgrader.py:** 87% line coverage (33 tests)
  - **randomgeneration.py:** 95% line coverage (33 tests)



```
randomgeneration_integration_test.py ..................                                                              [ 27%]
randomgeneration_unit_test.py ...............                                                                        [ 50%]
randomgrader_integration_test.py .....                                                                               [ 57%]
randomgrader_unit_test.py ............................                                                               [100%]

================================================= 66 passed in 0.49s ==================================================
```

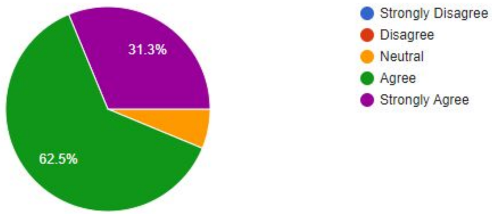
User Testing Results

- System Usability Scale (SUS) was used.

- Majority of the UI Problems are with PrairieLearn itself:
    - Tabs made it cluttered.
    - Submit button location.
    - Difficulty with navigation.

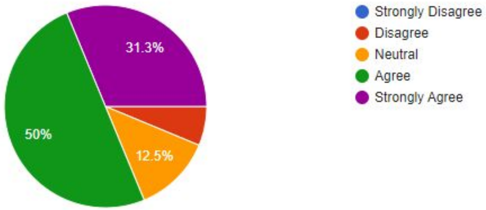
I think I would like to use this system for my assignments.

16 responses



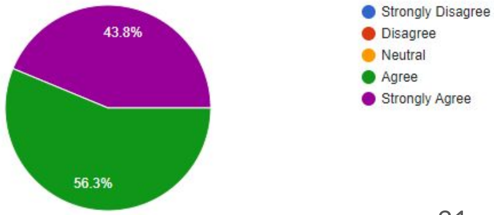
I thought the system was easy to use.

16 responses



Overall I find this system useful

16 responses



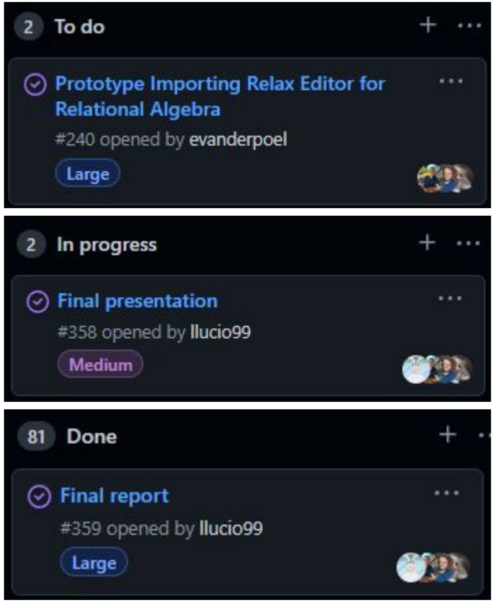
User Testing Results

- Bugs found:
  - Info Button Crashing applications **(FIXED)**.
  - Font of PrairieLearn changed with question **(FIXED)**.
- Suggestions For ER Design Question:
  - Add text field to auto generate ER from string form.
  - Disable clicking on add attribute to stop menu closures.
  - Font Style.

Workflow/Deployment

- Languages
  - Python
  - Javascript
- Project Management
  - Github Projects
- Code Spaces
  - Pycharm

- Deployment
  - Docker
  - Linux Server
- CI/CD
  - Drone IO



2 To do

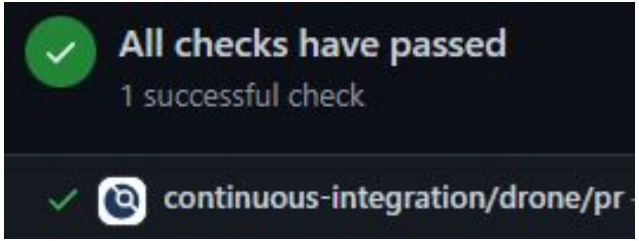
Prototype Importing Relax Editor for Relational Algebra
#240 opened by evanderpoel
Large

2 In progress

Final presentation
#358 opened by llucio99
Medium

81 Done

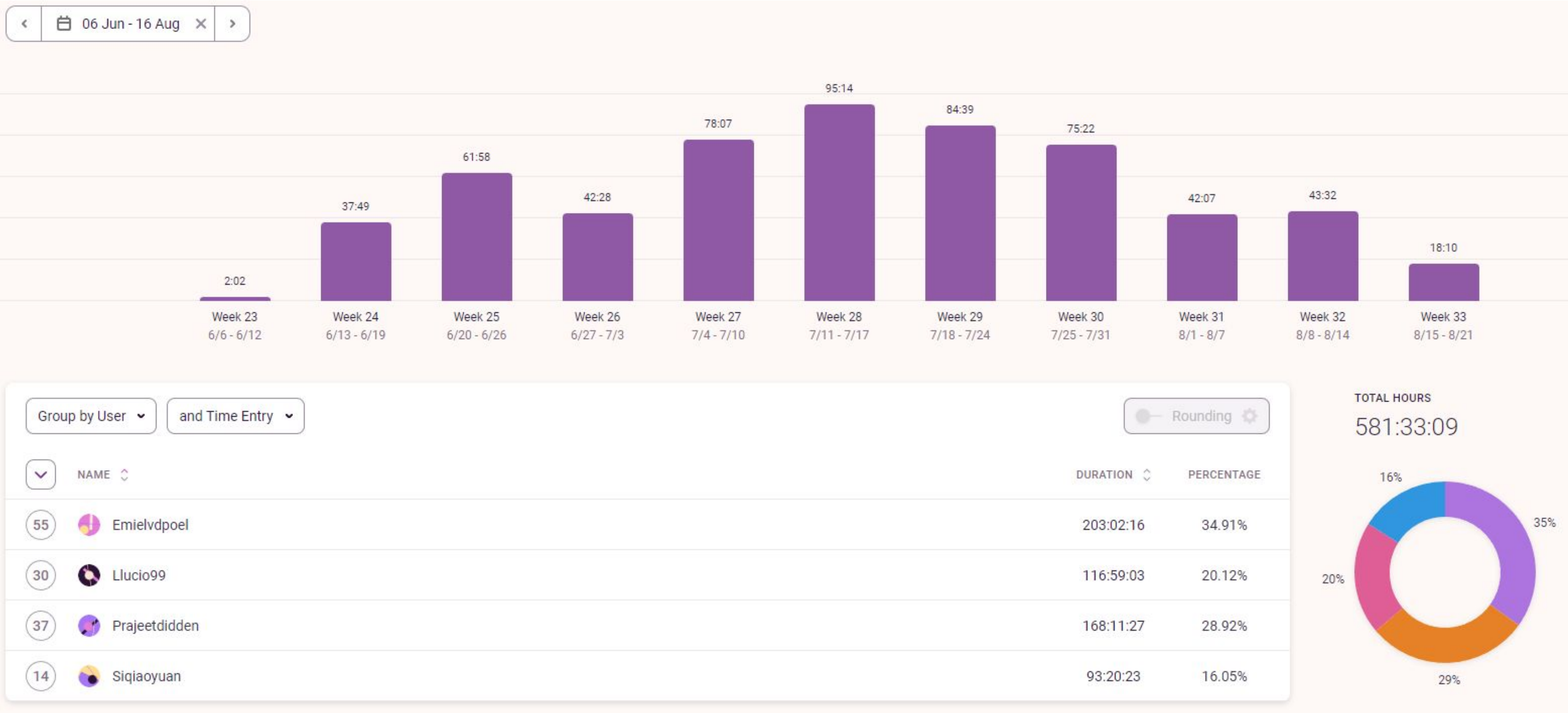
Final report
#359 opened by llucio99
Large



All checks have passed
1 successful check

✓ continuous-integration/drone/pr

Project Statistics (Toggl Time Report)



06 Jun - 16 Aug

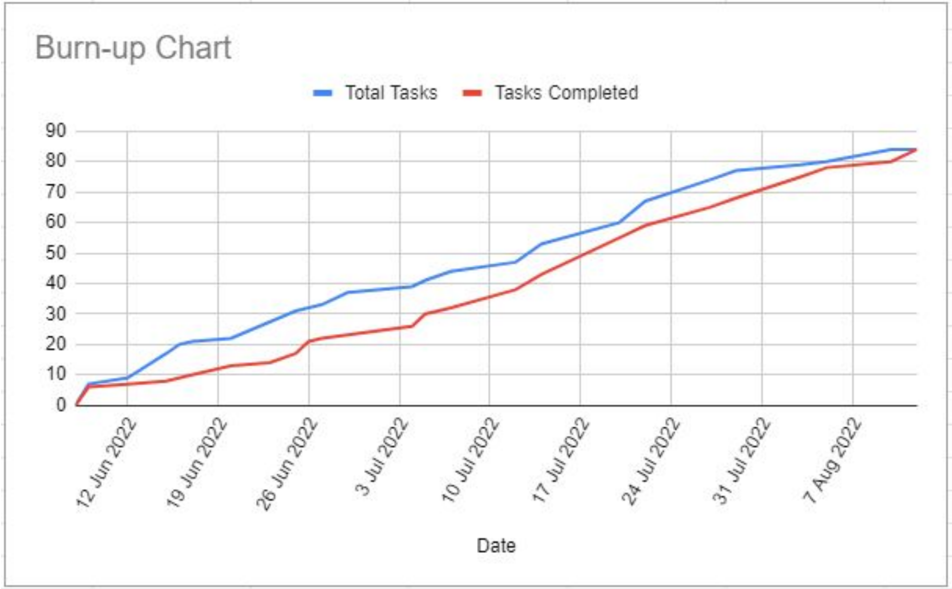
Group by User

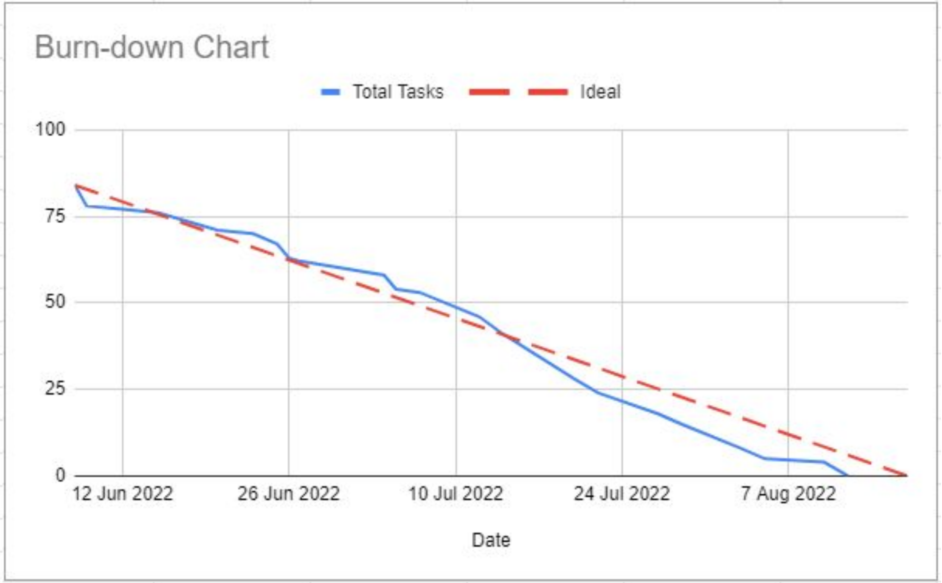
and Time Entry

TOTAL HOURS
581:33:09

	NAME	DURATION	PERCENTAGE
55	Emielvdpoel	203:02:16	34.91%
30	Llucio99	116:59:03	20.12%
37	Prajeetdidden	168:11:27	28.92%
14	Siqiaoyuan	93:20:23	16.05%

Project Statistics (Burn up/Burn down)



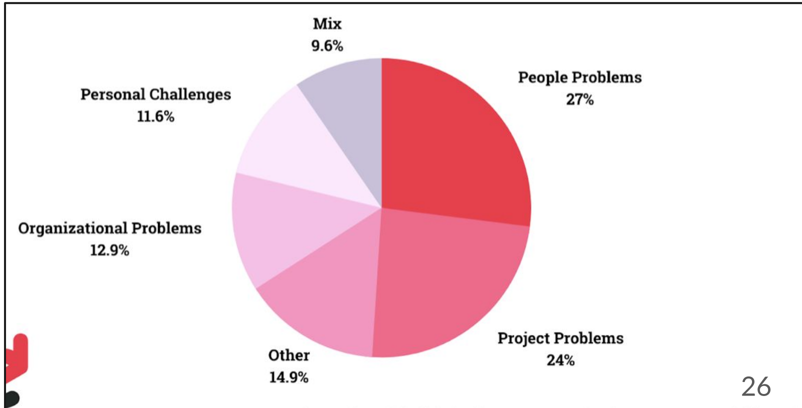


What have we Learned?

- No matter the difficulty of the task don’t underestimate how much work it may take.
- Always expect the unexpected.
- Working with Legacy Software is difficult and takes time to understand the codebase.

What would we do differently?

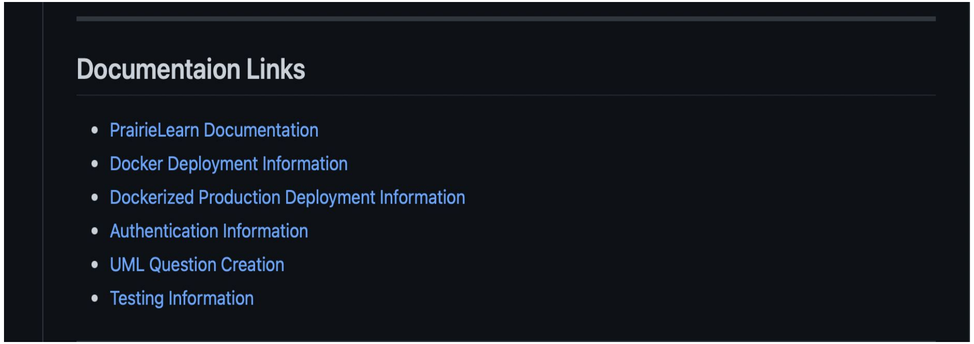
- Put a large focus on docker early on.
- Evaluate prototypes on a deeper scale.



Source: https://thedigitalprojectmanager.com/project-management-challenges/

Summary

- As project is deployed, handoff will be seamless.
- Project documentation can be found on project repository.

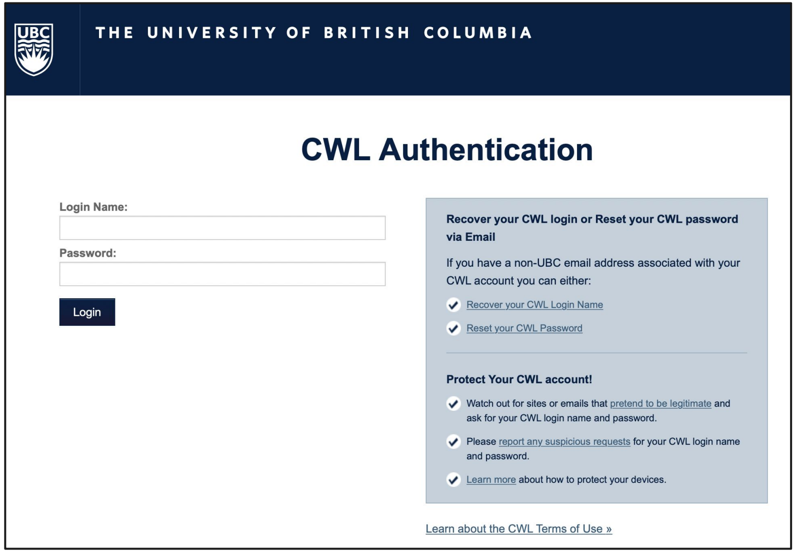


Documentaion Links

- PrairieLearn Documentation
- Docker Deployment Information
- Dockerized Production Deployment Information
- Authentication Information
- UML Question Creation
- Testing Information

Future Work

- CWL Integration
- Importing Relax Editor for relational Algebra questions



THE UNIVERSITY OF BRITISH COLUMBIA

CWL Authentication

Login Name:

Password:

Recover your CWL login or Reset your CWL password via Email

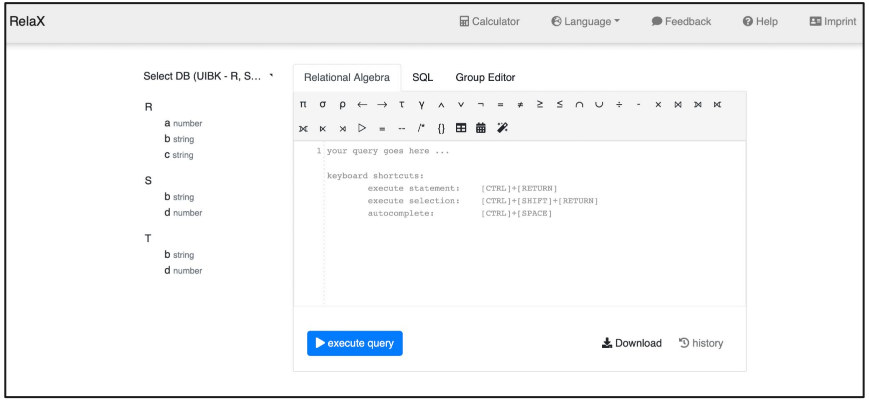
If you have a non-UBC email address associated with your CWL account you can either:

- ✓ Recover your CWL Login Name
- ✓ Reset your CWL Password

Protect Your CWL account!

- ✓ Watch out for sites or emails that pretend to be legitimate and ask for your CWL login name and password.
- ✓ Please report any suspicious requests for your CWL login name and password.
- ✓ Learn more about how to protect your devices.

Learn about the CWL Terms of Use »



R
- a number
- b string
- c string

S
- b string
- d number

T
- b string
- d number

```
your query goes here ...

keyboard shortcuts:
        execute statement:     [CTRL]+[RETURN]
        execute selection:     [CTRL]+[SHIFT]+[RETURN]
        autocomplete:          [CTRL]+[SPACE]
```

Thank You!

Questions?