CST 238: Graphical User Interfaces

Puzzle 8 - Oregon Tech Student Planner

Provided Material:

Locate and clone the puzzle repository:

- https://github.com/StewartTaylor/CST-238/tree/master/Puzzles/Week%209 (Qt/Qml)

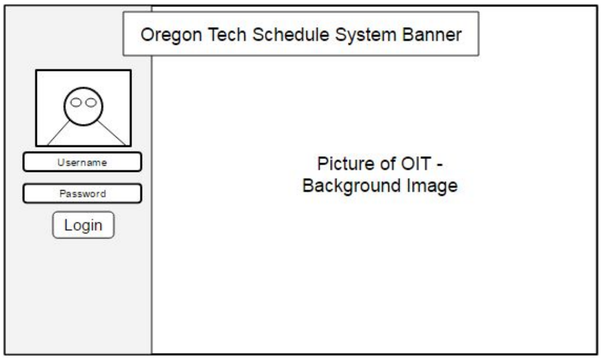
After cloned you will have the base project for the final puzzle. In the past a near complete project was provided and students were expected to fill in the missing pieces. For this puzzle you have been provided extensive content (images and data) and are expected to produce a full scene. Use your existing code base to accelerate the completion of this puzzle.

Puzzle:

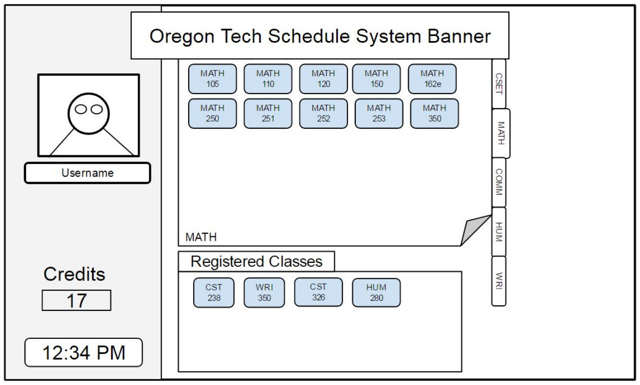
It is your final year at **Oregon Institute of Technology**. The dean of students has received extensive complaints about the **scheduling system** over the years. Despite OIT's efforts to stay current, the existing class schedule system isn't "flashy" enough for 2025 (Yeah you ended up being here a few more years than you planned). The Dean has asked the **CSET** department if they could find a talented individual to help.

Ever since your amazing project in **CST 238**, everyone has **known you have skills**. When the CSET professors thought of the need for a new "GUI" schedular, your project was the first thing to pop into their head. After working out an appropriate compensation and award for you at graduation, you agree to doing it.

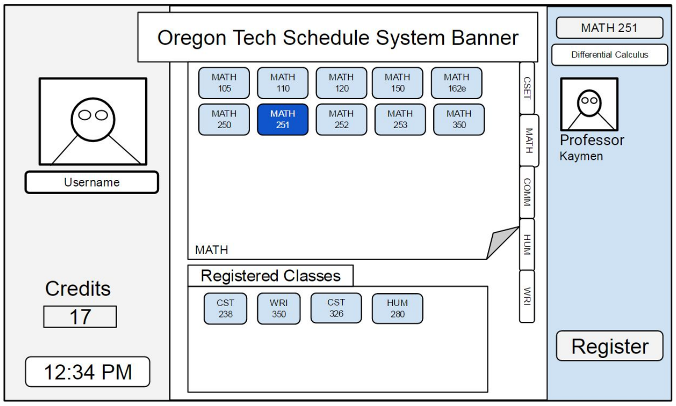
In general the schedule system aims to be minimal. Many of the computers that will run the client application will be old. The schedule itself has 3 main states. You do NOT need to use qml "States" - I simply mean 3 different views produced depending on what the state of the application is.



Logged Out/ HOME



Logged In

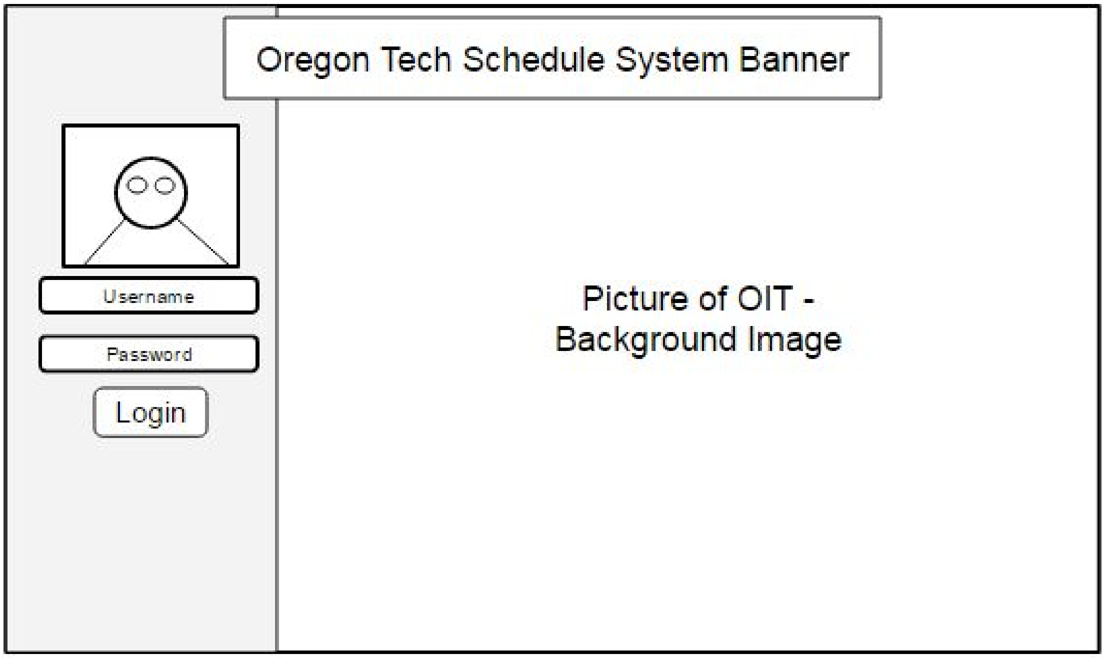


Class Active

Make sure to investigate the images provided in starter project before beginning work.

CST 238: Graphical User Interfaces

Puzzle 8 - Oregon Tech Student Planner



Logged Out State Requirements:

★ Before the user is logged in, the "**avatar**" image should be displayed.

★ The user must be able to login using one of three test profiles:

○ Username: **admin**	Password: **IRuleOver4LL**
○ Username: **student**	Password: **student**
○ Username: **tester**	Password: **BetaFish**

○ **NO OTHER USERNAME OR PASSWORD COMBINATION SHOULD WORK!**

★ After successful login the displayed user image should change to the corresponding image based on the **Username**.

★ After successful login the background image should disappear and leave a passive color for the background.

★ The banner must be created, use images (you find on your own), text, or whatever you want to develop it.

★ After successful login the application should transition into "**Logged In**" state.

★ If the user enters incorrect login information they are **notified visually**.

★ When the user logs out, the user image changes back to "**avatar**".

○ Make sure password data gets cleared appropriately

CST 238: Graphical User Interfaces

Puzzle 8 - Oregon Tech Student Planner

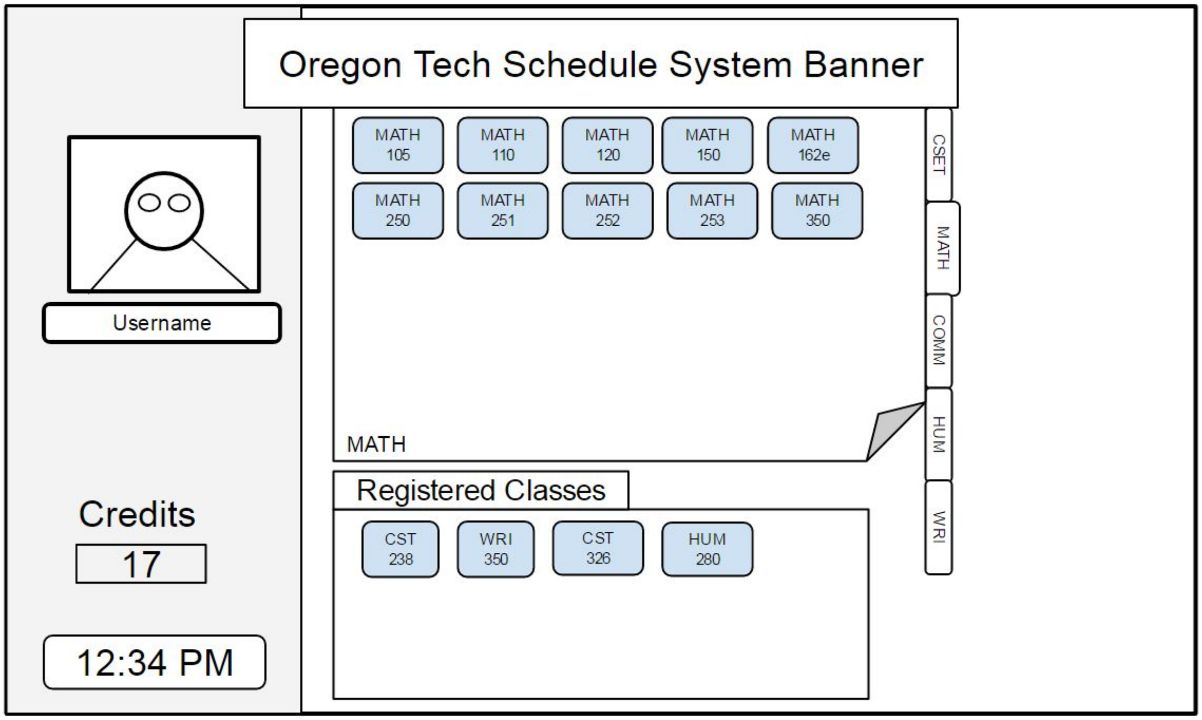
Final Steps: You have the option to implement **1 of 2** feature requirements.

1.) **Clock**: The clock will use a C++ object QTime to get the current time in string form. Use a signal to pass the string up into QML. The Clock class should also have a QTimer . The timer will be used to properly update the time in QML after a minut has passed. (Make sure your clock handles changing to the next hour). If you decide to do the clock, you must still have the feature to register a class. You can do this however you prefer - I recommend using a similar technique to the one used in puzzle 7.

2.) **ClassView**: If you decide to do the ClassView you will not need to impliment a clock. However, your ClassView should be the requirements listed + a little extra. Make a transition (or animation) for adding and removing items from the lists. See puzzle week 6 for hints. To be clear ClassView is the object on the right side that shows the professor and more details about the course.

If you choose to do both you will be awarded Extra Credit.
Logout must be added in either case. And in both cases you will have a working “Registered Classes” list. Credits should also be adjusted correctly.

Puzzle 8 - Oregon Tech Student Planner

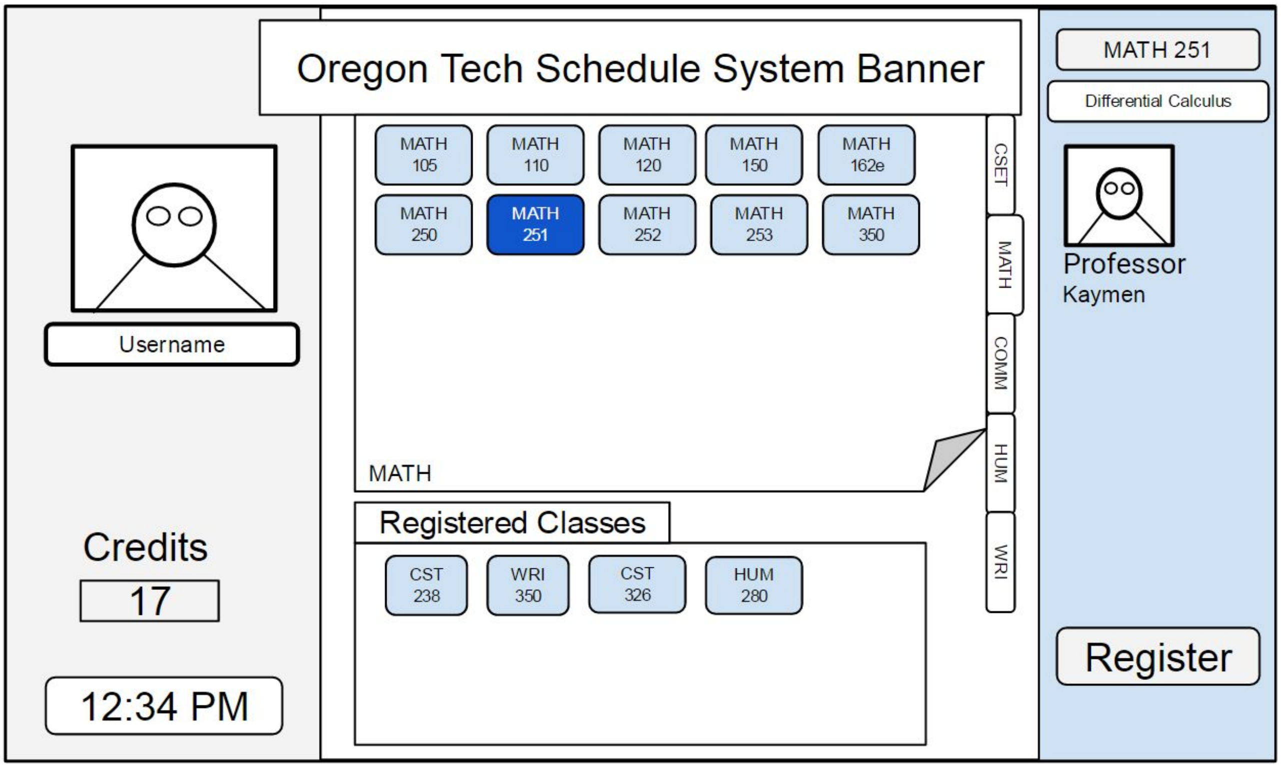


Logged In State Requirements:

- After the user logs in - the clock, credit counter, Master class list, and Registered classes list should "*load*" and become visible.
- The lists should load into a "clean state" with an empty registered classes list. (Persistence will be added later down the road)
- The current selected "Classes Tab" should be visually distinct from the unselected tabs.
- The class list shown should change based on what Tab is selected.
- The application must calculate correct credit count when the user adds or removes from their "Registered Classes" list.
- The application must visually display the current classes added.
- The user must be able to "*click*" on a class and move from **Logged In** state to the **Class Active** state.

CST 238: Graphical User Interfaces

Puzzle 8 - Oregon Tech Student Planner



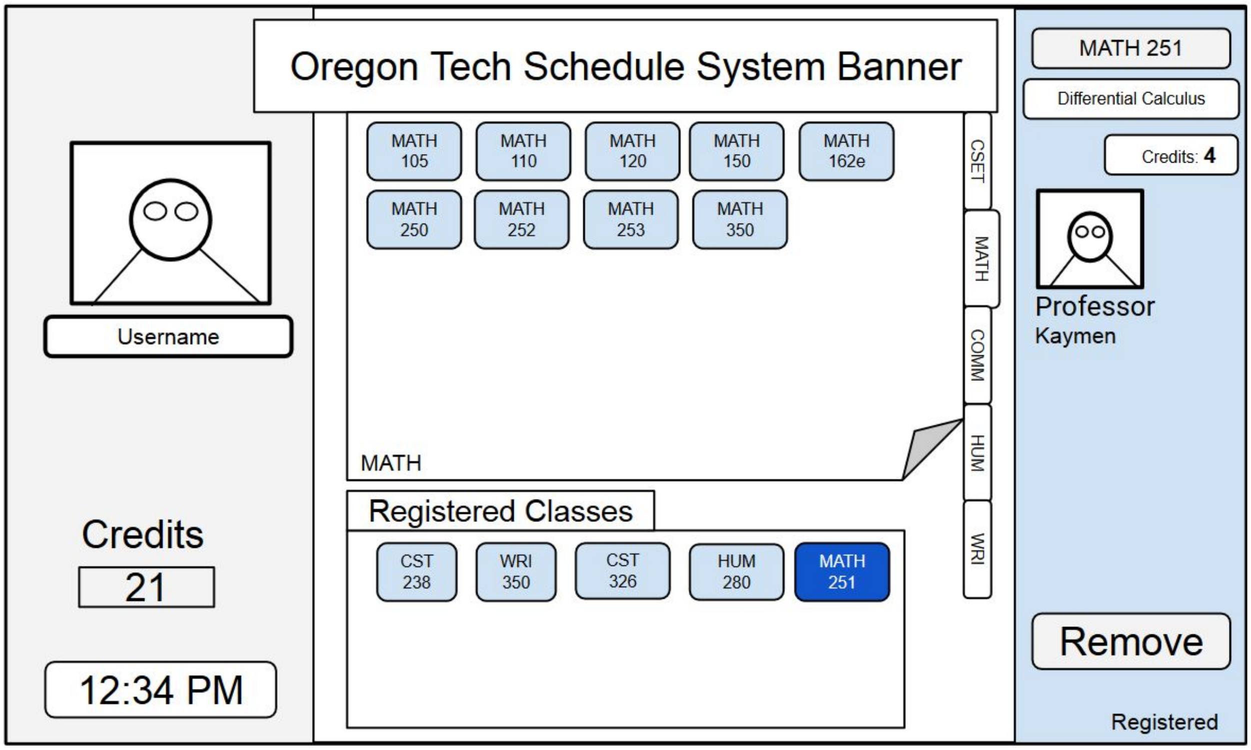
Class Active State Requirements:

- ★ When a class is selected, the ClassView object will "slide-in" from the right.
- ★ When a class is deselected (focus moves to another object) the ClassView will slide back out as the application transitions back to **Logged In** state.
- ★ For all classes show Professor Kayman as the professor. Classes will have instructors added later.
- ★ Generate 3 classes for each major for the master class list. Each Class will need to contain at least 3 pieces of data:
    - ○ **name** - human readable name - "Differential Calculus"
    - ○ **course** - The value shown in the class delegate - "MATH 251"
    - ○ **major** - Holds the value of major the class belongs to "Math"
    - ○ **credits** - Credits the class is worth

```
ListElement{
    name: "Differential Calculus"
    course: "MATH 251"
    major: "MATH"
    credits: 4
}
```

CST 238: Graphical User Interfaces

Puzzle 8 - Oregon Tech Student Planner



Class Active State Requirements (continued):

- ★ When the class is “registered” it should move to the registered list.
- ★ Class view should **show** the class is registered when it is selected and then provide a “Remove” button (or “Register” depending on what list it is in).
- ★ When a new class is selected the ClassView should <u>update its content accordingly</u>.
- ★ The clock (if implemented) must show the correct time.
- ★ Finally add the ability to “Logout” of the current user and login with another. (The logout button is not shown in the wireframe) - be creative and decide where you feel it fits best.

Remember you are encouraged to use your existing Components to build this puzzle very quickly.