User stories with their appropriate acceptance criteria

Title:	UserStory-1
As a	directory of center for teaching, learning, and assessment
I want	to assess courses and programs achievement of learning outcomes over time
so that	I can compare what strategies work to get the most out of courses
Acceptance Criteria:	
Given	the results from the post-course professor survey of a courses information including learning outcomes and assessment info
When	we can find which courses/programs are successful and which don't meet their goals
Then	adjust the courses
Title:	UserStory-2
As a	professor
I want	to assess if my students are meeting the courses learning outcomes
so that	I can tell if my course is meeting its goals
Acceptance Criteria:	
Given	the results from the post-course professor survey of a courses information including learning outcomes for individual courses
When	we can find the percentage of students meeting the learning outcomes over time
Then	compare if any changes to the course helped or harmed the effectiveness of a course
Title:	UserStory-3
As a	Administrative assistant
I want	to examine if courses are meeting the learning outcomes they claim to be
so that	we can confirm that students have opportunity to take courses that meet each outcome
Acceptance Criteria:	
Given	the results from the post-course professor survey of a courses information including learning outcomes for individual courses
When	we examine the professors course descriptions
Then	we quantify what courses meet each leaning outcomes

Describing the process and development to make your work reproducible, for example, tidying data. (If applicable)

Vis 1: First, we looked at our data and made necessary adjustments. This included tidying as well as creating new data. For the first visualization, we created our own fake data as we didn't have a large enough time range to answer the question. For the visualization to work, there must be a 'LO' column with the digit for a corresponding learning outcome, 'Program' column with the program title, 'Test_Year' column with the year of the test, and 'Test_Perc' with a 0-1 value for percentage of student meeting the learning outcome.

Collapsible Tree: First, in order for the CWLO and Course Num to read off as type numeric instead of type character (in order for there to be different ways users can color nodes in the collapsible tree), we take the process of changing to numeric in the following format: LearningGoalsDataset$"CWLO"<- as.numeric(LearningGoalsDataset$"CWLO"). Then, we filtered out the NA data in interpretations so that the collapsible tree would accurately match to the number of nodes: "LearningGoalsDataset <- filter(LearningGoalsDataset, Interpretation != "NA")."

Wordcloud: First we used the same filters as described above (filtering out the "NA" interpretation). Then we analyzed the learning outcome descriptions on Grinnell's website to find keywords for each learning outcome. We added all these keywords to a list for each learning outcome. We then analyzed the course description to see if it contained any of these keywords, and if it did we added that learning outcome to a list of all the potential learning outcomes. For example, if a course descrition contained the words "analyze" and "fairness", then we would assign it to learning outcomes 1 and 2. Following that we went through every course, tallied up the number each learning outcome we had generated, and then created a word cloud based on how many times each learning outcome showed up. We also ensured that the user would be able to filter based on department in order to be able to look at select programs.

Finally, we went through every course (filtered by what the user selected) and we looked at each learning outcome selected by the professor, and checked to see if it was one of the possible learning outcomes generated by our program. We tallied this up for each learning outcome, the calculated the percent of courses where the professor's selected learning outcome was one of our generated learning outcomes. We then displayed with as a bar chart, and calculated the standard deviation to display as well so the user could visualize our confidence interval.

Architectural design

Trade offs:

- Aesthetic for information
- Functionality for style

- Size for space

DATA	**Transfer Layer**	**Visual Layer**	**Interaction Layer**	**USER**
Given survey results	Renaming column names to workable values	Removing unnecessary information to avoid cluttering screen	Visualizations can be filtered and rearranged by variables like program, learning outcome, and year to allow user to control what they are looking at	CLS
	Randomizing values off of given data to give more information for visualizations	Using accessible colors so all users can interpret data	Tree nodes can be clicked to reveal more layers and information	
	Separating individual courses and academic programs	Trees show connections and layers between programs and learning outcomes	Scatterplot points can be hovered to reveal more information such as what learning outcome they represent.	
		Scatterplot shows student achievement of learning outcomes over a range of years with fit line showing general trend		
		Box plot and word cloud show how often programs actually meet learning outcomes compared to how often they claim too		

Design Decision

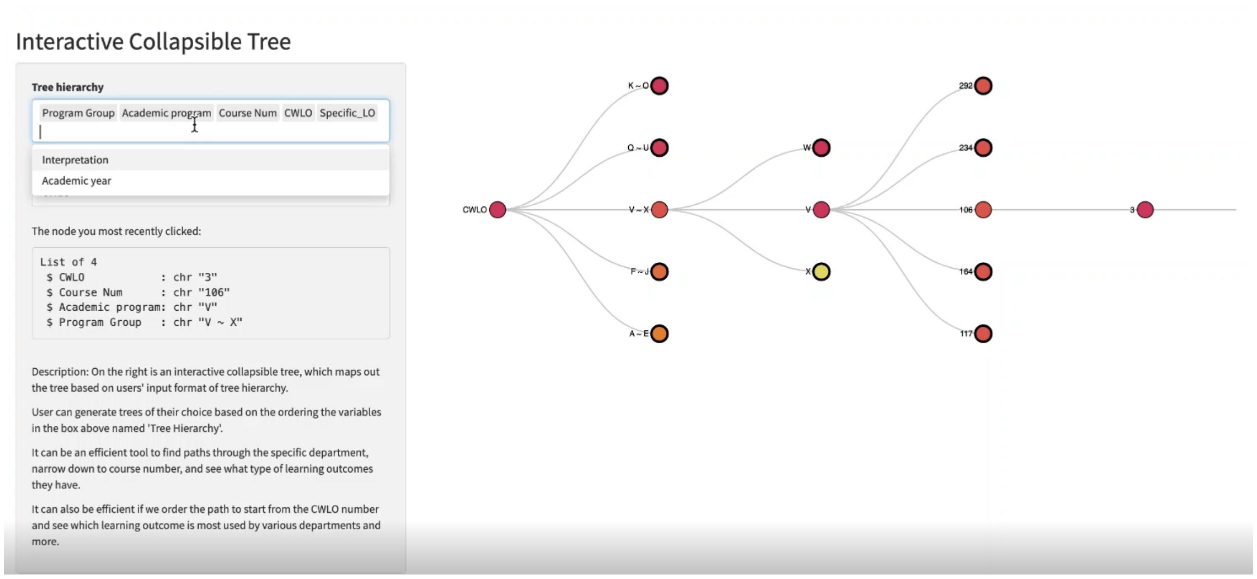
Visualizations	**What**	**Why**	**How**
Scatterplot	Filter by Learning Outcome, Program, or Year	Allowing users to control the data they are examining	Adding a checkbox, text field, or slider to change the variables

Scatterplot	Changing colors	Help users with visibility difficulties	Drop down menu with 4 color options that should work for different types of colorblindness
Scatterplot	Trendline	Allow user to see general trend of data	Adding line of best fit based on the points in the plot
Collapsible Tree	Tree that retracts and expands based on clicking of node of choice to find path	To allow interaction with app and user to create trees of their choice depending of which variables they want to specifically look into	Adding tree hierarchy selection tool, node coloring tool to format the tree of their choice
Vertical Tree	Standard tree displaying the 6 learning outcomes that expands to specific courses that utilize that CWLO	Based on client's proposal for such tree that can easily be followed along	Plotting the tree hierarchy by CWLO, Course Num, to Specific Learning outcomes given by Professors in that order
Learning outcome text analysis program	Generate a learning outcome based on course description	To analyze which could had multiple LOs, and which were the most prevelant	Search for key words in course description
Word cloud	Word cloud visualize the prevalence of each LO based on our LO generation program	Visualize which LOs were the most prevelant	See how many courses were considered each LO then create word cloud
LO Match Bar Chart (with standard deviation)	Bar chart visualize what percent of courses in each LO had the professor’s selected LO agree with out generate LOs	Understand if professors are selecting the correct LO	Compared selected LO to generated LO then calculate percent correct and standard deviation

Design Patterns

http://www.cs.umd.edu/hcil/trs/2005-23/2005-23.html

(Collapsible Tree Visualization)



Interactive Collapsible Tree

Tree hierarchy

Program Group | Academic program | Course Num | CWLO | Specific_LO

Interpretation

Academic year

The node you most recently clicked:

```
List of 4
 $ CWLO          : chr "3"
 $ Course Num    : chr "106"
 $ Academic program: chr "V"
 $ Program Group  : chr "V ~ X"
```

Description: On the right is an interactive collapsible tree, which maps out the tree based on users' input format of tree hierarchy.

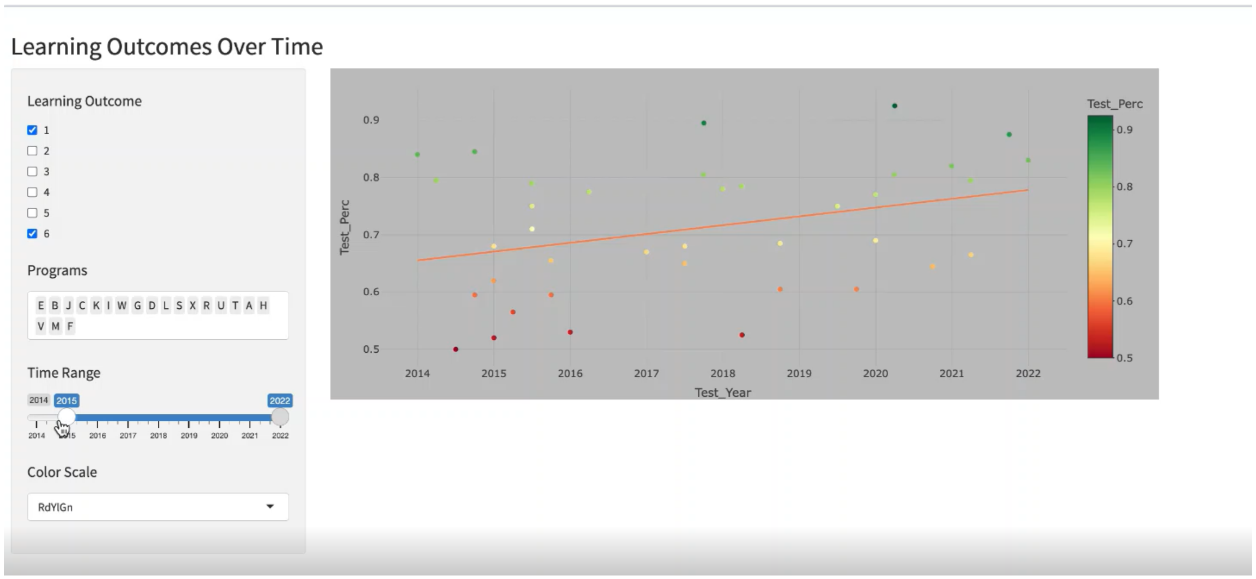
User can generate trees of their choice based on the ordering the variables in the box above named 'Tree Hierarchy'.

It can be an efficient tool to find paths through the specific department, narrow down to course number, and see what type of learning outcomes they have.

It can also be efficient if we order the path to start from the CWLO number and see which learning outcome is most used by various departments and more.

The first design pattern we used was "Seed then Grow". Here we allow users to click on nodes to expand information after the nodes, then retract nodes by clicking on them again. This allows the user access to all the information and visualization, without crowding the entire screen by having a fully expanded tree.

Visual Query:



Learning Outcomes Over Time

Learning Outcome

- [x] 1
- [ ] 2
- [ ] 3
- [ ] 4
- [ ] 5
- [x] 6

Programs

E B J C K I W G D L S X R U T A H V M F

Time Range

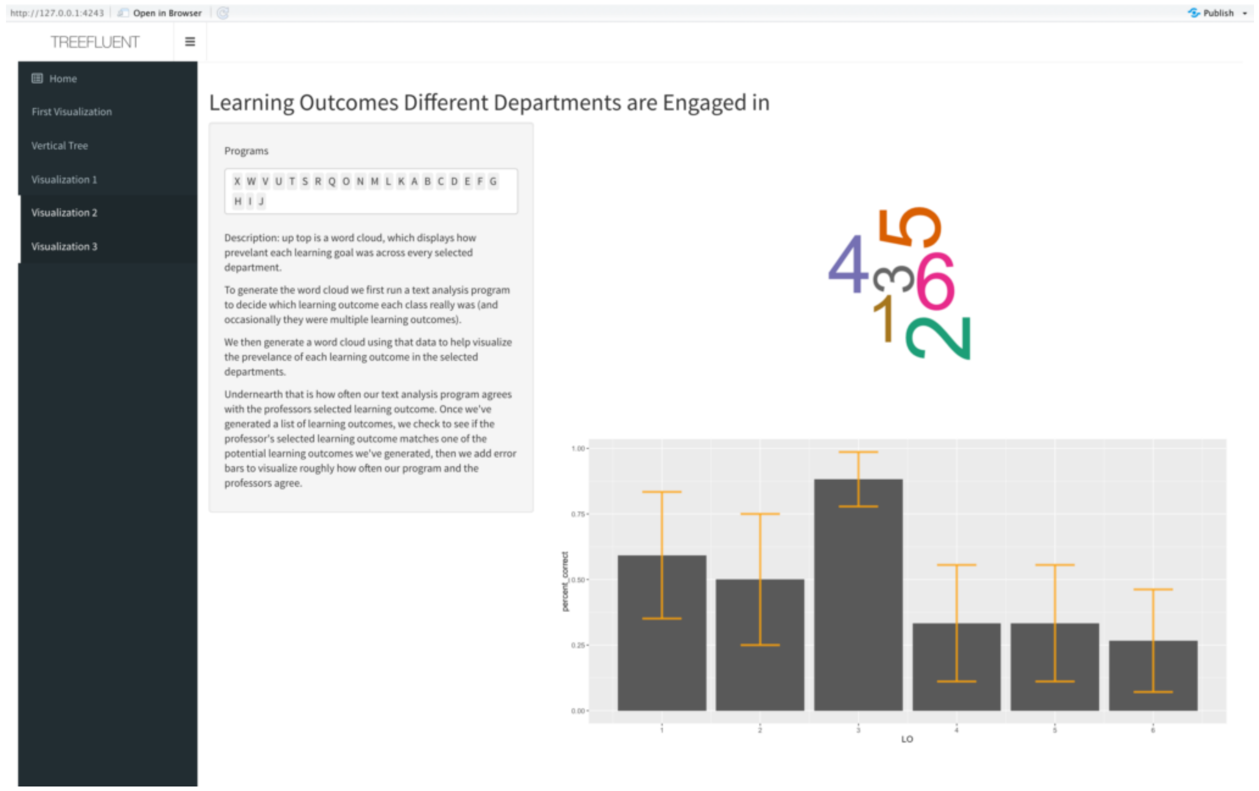
2015 – 2022

Color Scale

RdYlGn

We also used visual query. Here, data is laid out in a manner so that viewers can see progress over time. They can see both the trend line, but they can also see the general distribution of dots allowing them to better understand our visualization. Finally, the scaling colors also helps them understand how ideally the data ill go from more red, to more green (depending on the color scale), thus increasing the percent of students who achieved the learning goal.

Cross View Brushing:



Learning Outcomes Different Departments are Engaged in

Programs

X W V U T S R Q O N M L K A B C D E F G H I J

Description: up top is a word cloud, which displays how prevelant each learning goal was across every selected department.

To generate the word cloud we first run a text analysis program to decide which learning outcome each class really was (and occasionally they were multiple learning outcomes).

We then generate a word cloud using that data to help visualize the prevelance of each learning outcome in the selected departments.

Undernearth that is how often our text analysis program agrees with the professors selected learning outcome. Once we've generated a list of learning outcomes, we check to see if the professor's selected learning outcome matches one of the potential learning outcomes we've generated, then we add error bars to visualize roughly how often our program and the professors agree.

Here we use cross view brushing to visualize both the word cloud and the accuracy of the professor’s learning outcome assignments. By viewing this simultaneously we can see things such as that learning outcome 3 was the smallest, but we agree with a lot of the professors who said their class aim was learning outcome 3. We can also see learning outcome 6 was very prevalent, but we disagree with a lot of the professors who said their class was learning outcome 6. Viewing these things simultaneously help the viewer better understand the whole picture.

Code Construction Guidelines (Coding standards)

Coding Standards and Guidelines

Team Treefluent

1. Thoroughly plan your code before implementing an idea
    a. Declare variables close to where they're used
    b. Order declarations logically
    c. Create an outline for different sections of the app before writing
2. Don't document more than absolutely necessary
    a. Explain purpose of functions or complicated lines
    b. Excess explanation only distracts the reader
3. Write readable code
    a. Clear and concise variable names
        i. camelCase
        ii. Consistent for similar functions
    b. Function names should reflect their purpose
        i. Also do camelCase
4. Security
    a. Keep individual user data private
    b. Only use aggregate data in visualizations
5. Reduce the complexity of programs
    a. Find simple implementations
        i. Don't overcomplicate code wherever possible
6. Test and review new implementations
    a. Make sure new code works
    b. Look for improvement on prior code
7. Iterate on your designs and repeatedly measure progress
    a. Communicate with Vanessa to find issues with current code
    b. Ask group members to check code for possible improvements
    c. Save progress to learn from previous implementations
8. Formatting
    a. Use empty lines to separate groups
    b. Within a block, align all the statements to the same tab
    c. Use indentation to show the hierarchy of the code

d. Use spaces around operators

References

A. Khan, “Interactive collapsible tree diagrams using 'd3.js' [R package collapsibletree version 0.1.7],” *The Comprehensive R Archive Network*, 22-Aug-2018. [Online]. Available: https://cran.r-project.org/web/packages/collapsibleTree/index.html.

“Collapsibletree source: INST/Examples/03shiny/app.r,” *collapsibleTree source: inst/examples/03shiny/app.R*. [Online]. Available: https://rdrr.io/cran/collapsibleTree/src/inst/examples/03shiny/app.R.

“Create a new ggplot,” *ggplot*. [Online]. Available: https://ggplot2.tidyverse.org/reference/ggplot.html.

E. L. Pennec, “ggwordcloud: a word cloud geom for ggplot2,” *Ggwordcloud: A word cloud geom for GGPLOT2*, 01-Jun-2019. [Online]. Available: https://cran.r-project.org/web/packages/ggwordcloud/vignettes/ggwordcloud.html.

M. Bostock, “Collapsible tree,” *Observable*, 26-Aug-2020. [Online]. Available: https://observablehq.com/@d3/collapsible-tree.

“Plotly,” *Plotly r graphing library in R*. [Online]. Available: https://plotly.com/r/.

“Scatterplot,” *The R graph library* [Online]. Available:

https://r-graph-gallery.com/scatterplot.html.