Library Website

Edward Rees

TEC 674: MAT Capstone Project

University of San Francisco

Table of Contents

Unit Introduction

Having just graduated with my bachelor's degree in Computer Science, I knew I wanted to do my unit plan for a future Introductory Computer Science course. While I'm pursuing my Single Subject Credential in Mathematics, I ideally would like to teach Computer Science. Therefore, when I had the opportunity to approach this course to create a unit plan for a Computer Science course, I jumped at the opportunity.

This unit plan provides an opportunity for students to apply what they have learned throughout the school year so far. Before this unit, students have already learned the fundamentals of programming in the language of JavaScript. Students have learned how to use variables, data types, conditional statements, control flow through iteration, functions, strings, collection structures, exception handling, pseudocode, reading input, writing to output, asynchronous programming with promise handling, and a foundational understanding of recursion. While the projects covered in previous units reinforce the concepts learned, the project in this unit heavily focuses on applying the concepts and gaining a more conceptual understanding of the foundational concepts.

The first week of the unit will consist of an introduction to the project and various elements of the project. This week will also review pseudocode, symbols, and sorting algorithms. Students have implemented a simple sorting algorithm before, however, they haven't implemented a formal sorting algorithm. The first formative assessment will happen at the end of the first week. In the second week, students will review their implementation of the sorting algorithms, review recursion, implement merge sort, and implement various search algorithms.

They will end the week with a partial search implementation. In the third week, students will begin reading and parsing files, creating the foundation of their backend service. In the fourth week, students will implement user authentication routes, hash passwords, tokenize information, and begin developing their front-end system. In the fifth week, students will focus on their frontend implementation, focusing on the communication between the frontend and backend services. In the sixth and final week, students will finish their frontend design and ensure all systems are connected and working.

The project itself is a complete Library website system, with a functional registration system, login system, book reservation, and catalog viewer. This requires a backend to handle the data processing and a frontend to show the information to the user in an interactive manner. This unit expects students to have a strong foundation of the basic programming concepts, therefore the main concepts being taught in the unit revolve around writing a functional backend web server, connecting it to a functional web application, and writing efficient code by implementing different specific algorithms to make the user experience more effective. Students will have to make conscious decisions about their implementations and which algorithms they will use in their main project. Additionally, students will also be given various ranges of expected execution times. This will be used to provide students with references of how long it should take their code to run. Students will then use that information to review and revise their implementations to try to make them more efficient.

Ultimately, this unit and this project provide the answer to the question students often ask: “Why do we need to know this?” In creating this project, students will actively apply what they have learned.

Unit Calendar

Week 1:		
Learning Objectives for the Week	Language Goals	Vocabulary
Students will be able to define the components of a functional website. Students will be able to identify the problems a website solves. Students will be able to justify their reasoning for the problems solved by a website. Students will be able to break down the various components of developing the culminating project. Students will be able to recognize pseudocode. Students will be able to develop functional code based on the pseudocode provided. Students will be able to develop functional code based on the pseudocode of sorting algorithms.	- Students will be able to explain the benefits of developing a website to solve a problem. - Students will be able to explain the steps of pseudocode.	- Authentication - Collection - List - Parse - Sort

Mon.	**Tues.**	**Wed/Thurs.**	**Fri.**
Enduring Understandings			
Websites/Web Applications serve as accessible platforms providing services to users.	Websites/Web Applications serve as accessible platforms providing services to users.	Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.	Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.
Essential Questions			
How can a website help serve a community?	How can a website help serve a community?	How can we use algorithms to solve non-computational problems?	How can we use algorithms to solve non-computational problems?
Learning Objectives			
Students will be able to define the components of a functional website. Students will be able to identify the problems a website solves. Students will be able to justify their reasoning	Students will be able to break down the various components of developing the culminating project.	Students will be able to recognize pseudocode. Students will be able to develop functional code based on the pseudocode provided.	Students will be able to develop functional code based on the pseudocode of sorting algorithms.

for the problems solved by a website.			
WHERETO			
W: Introduce the culminating project. H: Provide students with a live functional example of the final library website.	W: Introduce the journal portion of the unit.	E: Review reading and interpreting pseudocode. E: Review converting pseudocode into functioning code. T: Progressively more complicated pseudocode to push student understanding.	R: Upon writing the two algorithms, students will test their code against benchmarks and time their code. They will then have an opportunity to try to revise and improve the efficiency of the algorithms. O: Sorting algorithms will be added to their backend system later.
Activities			
Introduce the project. Students will experiment with using the Library website. Students will be given a website, taking notes on the various technical elements	Introduce the journal portion and how it will be used to track their process throughout creating the Library website. Students will then go to the live functioning	In the beginning, a review of the technical elements found on Tuesday will be done. This will ensure students are on the same page with the various concepts required.	A review of the code generated by the pseudocode from the previous lesson will be given. This will help debrief the exercise and establish the foundation for

used in the given website. Students will then take 20 minutes researching a website of their choosing and take notes on what technical elements are used.	example of the Library website and analyze the technical elements used in the Library Project.	Students will then review the components of pseudocode, reviewing how to read and interpret the pseudocode and translate it into functioning code. Students will then implement 10 simple algorithms using the pseudocode provided, progressively getting more complicated.	writing the sorting algorithms. Students will then be given the pseudocode for both sorts. Originally, they will sort a physical hand of unsorted cards using the logic from the algorithms. After sorting the cards multiple times, they will have time to write the code.
Assessments			
			(Formative) Implement Insertion and Bubble Sort
Modifications			
- For the Pseudocoe review, as the symbols may be confusing, it may help to introduce the symbols one at a time. In doing so, explain each symbol as an individual component before moving into seeing the symbols collectively.			
Potential Misunderstandings			

<table>
<tr>
<td>The website students produce should not look identical to the example. It also shouldn’t have the same design. It should be unique to each student and their own style.</td>
<td>The journal will only be seen by the student and the teacher. Because of this, students may write more honestly and genuinely than if they believe other students may see their entries.</td>
<td>Confusing different shapes from the Pseudocode may occur.</td>
<td>Confusion over the two sorting algorithms and mixing up parts of both may occur.</td>
</tr>
<tr>
<td colspan="4">Materials & Resources</td>
</tr>
<tr>
<td colspan="4">- Computer
- Notebook
- Writing Utensils
- Cards (Friday only)</td>
</tr>
</table>

<table>
<tr>
<th colspan="3"><b>Week 2:</b></th>
</tr>
<tr>
<td><u>Learning Objectives for the Week</u>
Students will be able to compare the efficiency of</td>
<td><u>Language Goals</u>
- Students will be able to share their implementation of code and explain various steps of an algorithm.</td>
<td><u>Vocabulary</u>
- Algorithm
- Sort
- Search
- Recursion
- Sequential Search</td>
</tr>
</table>

three different sorting algorithms. Students will be able to interpret the pseudocode of two search algorithms. Students will be able to implement both search algorithms. Students will be able to analyze the pros and cons of two search algorithms. Students will be able to justify when to use one algorithm over another. Students will be able to implement a partial search algorithm.	- Students will be able to provide feedback to other students around their implementations, with efficiency and effectiveness in mind.	- Binary Search - Partial Search

Mon.	**Tues.**	**Wed/Thurs.**	**Fri.**
Enduring Understandings			
Algorithms provide the building blocks for	The way algorithms are programmed affects the	Algorithms provide the building blocks	The way algorithms are programmed affects the

creating systems and services that can help solve a wide range of problems, computational or otherwise.	efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.	for creating systems and services that can help solve a wide range of problems, computational or otherwise.	efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.
Essential Questions			
How can we use algorithms to solve non-computational problems?	How do algorithms impact the efficiency of a program?	How can we use algorithms to solve non-computational problems?	How do algorithms impact the efficiency of a program?
Learning Goals & Objectives			
Students will be able to compare the efficiency of three different sorting algorithms.	Students will be able to interpret the pseudocode of two search algorithms. Students will be able to implement both search algorithms.	Students will be able to analyze the pros and cons of two search algorithms. Students will be able to justify when to use one algorithm over another.	Students will be able to implement a partial search algorithm.
WHERETO			
E: Review Merge Sort.	E: Review the Pseudocode of both search algorithms.	R: Students will be given the expected	H: Give a warmup on brainstorming how to

R: Reflect on the run time difference between Insertion Sort, Bubble Sort, and Merge Sort. Revise their algorithms and implement Merge Sort based on the Pseudocode.	E: Review writing recursive functions. O: Both search algorithms will be used in their backend service.	range of run times. They will then have an opportunity to reflect on their implementation and revise their code.	solve the partial search problem. E2: Reflect on the various algorithms and what they've learned at the end of the two weeks. O: Partial Search will be implemented into their backend service.
Activities			
Review Insertion Sort and Bubble Sort code from Friday. Review Recursion. Review Merge Sort Pseudocode. Check run time and compare run times between sorting algorithms. Review Searching Algorithms.	Review writing recursive functions. Review the concepts of Sequential Search and Binary Search. Review the pseudocode of both algorithms. Students will implement both algorithms.	Debrief the Search algorithm code. Provide students with a range of expected run times to search through an unsorted and sorted list of values. Analyze the pros and cons of both algorithms. Include one of the Sort Algorithms before both Search	Introduce the concept of a partial search. Introduce the problem with an example of searching for a keyword and returning all of the results containing that keyword. Students will have 10 minutes to explore a functional implementation of a partial search.

		algorithms and compare run times. Present an argument for the use cases of both search algorithms. When is one algorithm preferable over the other?	After the exploration, students will brainstorm the pseudocode of a partial search algorithm. Students will implement their partial search. Closing: Add a new journal entry reflecting on the week and what was accomplished.
Assessments			
	(Formative) Implement Sequential Search and Binary Search		(Formative) Implement a Partial Search
Modifications			
Merge sort may be conceptually confusing, therefore it may help to provide students with more time practicing with a shuffled deck or hand of cards using the steps from Merge Sort.	If Binary Search and writing the Recursive functions are too confusing, start by writing the Sequential Search as that is the more straightforward algorithm to code.	Students may struggle with finding the pros and cons for both algorithms, therefore start by finding the pros and cons of one algorithm. Once the	If Partial Search is conceptually confusing, it may help to break it down by thinking of how to check for whether a string of text is inside another string of text, rather than

		first is done, start with the second.	whether two strings of text are identical. This should help develop a stronger understanding of the concept of a Partial Search.
Potential Misunderstandings			
Recursive Base Case may be invalid and cause an error. Interpreting the pseudocode of Merge Sort with the two recursive functions of Merge and Merge Sort may be confusing.	The recursive base case for the Binary Search may confuse some students. Address this by reiterating the concept of a base case and preventing an error with recursion.	Binary search only works when the data is sorted. The intention is for students to find this when analyzing the pros and cons, but this may not be concluded. This should be addressed at the end.	Partial search and Exact search may be confusing for some. Providing concrete examples may help clear up this confusion.
Materials & Resources			
Computer Cards (to sort) Pseudocode shape manipulatives			

<table>
<tr><th colspan="5">Week 3:</th></tr>
<tr>
<td colspan="2">Learning Objectives for the Week
Students will be able to read a file containing triple-colon-delimited values. Students will be able to implement a generic file parser based on their prior knowledge. Students will be able to explain their file parsing function to another student.
Students will be able to critique and provide advice to each other on revising the function and improving efficiency. Students will be able to create the foundation of their backend.
Students will be able to combine their code previously written into their backend.</td>
<td>Language Goals
- Students will be able to explain the steps of reading files and writing out files.</td>
<td colspan="2">Vocabulary
- Delimiter
- Data
- File
- Parse
- Function
- Generic
- Backend</td>
</tr>
<tr>
<td><b>Mon.</b></td>
<td colspan="2"><b>Tues.</b></td>
<td><b>Wed/Thurs.</b></td>
<td><b>Fri.</b></td>
</tr>
</table>

<table>
<tr><th colspan="4">Enduring Understandings</th></tr>
<tr><td>Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.</td><td>Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.</td><td>Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.</td><td>Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.</td></tr>
<tr><th colspan="4">Essential Questions</th></tr>
<tr><td>How is data stored and accessed by a user through a program?</td><td>How is data stored and accessed by a user through a program?</td><td>How do algorithms impact the efficiency of a program?</td><td>How do algorithms impact the efficiency of a program?</td></tr>
<tr><th colspan="4">Learning Goals & Objectives</th></tr>
<tr><td>Students will be able to read a file containing triple-colon-delimited values.</td><td>Students will be able to implement a generic file parser based on their prior knowledge.</td><td>Students will be able to explain their file parsing function to another student.
Students will be able to critique and provide advice to each other on revising</td><td>Students will be able to create the foundation of their backend.
Students will be able to combine their code previously written into their backend.</td></tr>
</table>

		the function and improving efficiency.	
WHERETO			
E: File reading is important for the file parsing component. Provide students with the foundation for reading a file, which will carry into the file parsing component. O: Setup for the File Parsing Summative.	H: Review reading a file from Monday's class. O: File Parsing will be a critical component for the Backend service, as it will provide the data used by the server.	R: Students will have an opportunity to reflect on their implementation, rethink after discussing with peers, and revise their implementations to improve efficiency. E2: Students will evaluate and create a new journal entry around their original thought process behind their generic file parser. They will also explain what they took away from discussing with a peer and how it changed their implementation.	O: Start working on the backend, using the various functions written throughout the previous weeks.
Activities			

Walkthrough reading a file. Walkthrough reading a file and pulling out specific content from the string constructed from the file contents. Review error handling with file reading. Challenge students to parse the file into a string, then into a list of strings.	Review file reading. Introduce the summative as implementing a generic file parser. This file parser must be a function that takes in the name of the file, and the data delimiter (separator), and should return a list containing an object for each line. The first line of the file will give the keys for the inner object. These keys should be used to generate the objects. Concrete examples will be provided in the prompt.	Review their file parser. Students share with each other their implementation and critique the implementation. The critique will focus on efficiency and whether there are more efficient methods of writing the code. Students will revise their code if they see fit. Students will create a new journal entry around the whole process. This process includes their original thought process, original implementation, discussing with a	Begin the backend of the project. Introduce working with npm with NodeJS. Add the ExpressJS package. Create the initial routes. Implement the sorting, search, partial search, and file parsing algorithms written. On load, have the server read the file and load the contents into memory.

		peer, taking and providing criticism, and revising the code if they see fit.	
Assessments			
	(Summative 1) File Parser Summative		
Modifications			
Reading the main file may be too complicated, therefore start by introducing smaller files to read. This will also help with providing opportunities to think of how to interpret and parse the data from the files.	Introduce the procedure of writing functions to read each individual file. Once the initial file parsing is done, provide students with probing questions about what code is similar in all the functions and what code is different. This may help with transitioning the code to a more generic implementation.	If some students haven't finished their generic file parser, they can use what they have currently for the critique. Given this is a block period, students could also be given some time in class to finish it, however, since this is a part of the summative assessments, only what they wrote on the prior day will be graded.	Start by writing multiple routes to see what they each do.

Potential Misunderstandings			
Delimiters may be confusing but clearing up that confusion will help with the file parsing portion. Delimiters are there to separate data and allow the data to be read in the expected way.	Their code may start by being too specific at first. This is okay, however, it should eventually be more generic and work on different files with different delimiters.		Without a "listen" function call, students may be confused as to why the server isn't running or why no values are being shown. Therefore, the first step after creating the initial routes should be to include the "listen" function.
Materials & Resources			
Computers			

Week 4		
Learning Objectives for the Week Students will be able to develop code to safely store user credentials.	Language Goals - Students will be able to describe their authentication implementation.	Vocabulary - Frontend - Backend - Npm - Node Packages - Router

<table>
<tr><td>Students will be able to create backend routes to serve data to the front end.
Students will be able to implement a functional user authentication system through the backend service.
Students will finalize the backend system with all of the required routes.
Students will be able to serve web pages to the front end from the backend, based on the user’s requested data.</td><td></td><td colspan="2">- ReactJS
- Bcrypt
- Hashing
- Jsonwebtoken
- Token</td></tr>
<tr><th>Mon.</th><th>Tues.</th><th>Wed/Thurs.</th><th>Fri.</th></tr>
<tr><td colspan="4">Enduring Understandings</td></tr>
<tr><td>The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.</td><td>The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.</td><td>Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.</td><td>Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with</td></tr>
</table>

			each other to serve the user.
Essential Questions			
How do algorithms impact the efficiency of a program?	How is data stored and accessed by a user through a program?	How does data transfer between a website and a server?	How does data transfer between a website and a server?
Learning Goals & Objectives			
Students will be able to develop code to safely store user credentials. Students will be able to create backend routes to serve data to the front end.	Students will be able to implement a functional user authentication system through the backend service.	Students will finalize the backend system with all of the required routes.	Students will be able to serve web pages to the front end from the backend, based on the user's requested data.
WHERETO			
E: Discuss hashing algorithms to prepare students for the user authentication system.	E2: Students will add a journal entry at the end of class outlining the process of implementing user authentication. O: User Authentication will be a key feature in the culminating backend project,	R: Collaboration will also include providing criticism toward each other's projects and reflecting on what routes are required and what routes aren't.	O: Begin working on the front end.

	as users will have to be logged in to reserve a book.	O: Finish the backend system.	
Activities			
Add bcrypt and jsonwebtoken to the backend packages. Introduce working with bcrypt to hash data and store it locally in memory. Compare two values hashed and check for equality using bcrypt's built-in comparison functions. Introduce tokenizing data using jsonwebtokens and send tokens to the front end. Introduce reading jsonwebtokens and parsing for equality.	Review bcrypt and jsonwebtokens. Collaboratively brainstorm the pseudocode for handling user authentication routes. Students will begin working on implementing user authentication either based on their own thought process or on their group pseudocode. Upon finishing implementing user authentication, students will add a new journal entry, explaining and reflecting on the process.	Check-in with the user authentication system. Ensure the system works as expected. Work period. Students will brainstorm which additional routes are required and work through implementing the routes. Students can collaborate and work together or work individually. Collaboration is encouraged.	Introduce working on the frontend system using ReactJS. Review creating a new React project and setting up the boilerplate code. Review ReactJS syntax and required code to get data/pages shown. Add the npm packages of react-router-dom and Axios. Introduce working with react-router-dom to serve different pages.

			Students will be asked to serve 5 different pages through the Router system.
Assessments			
	(Summative 2) User Authentication Backend		(Formative) Serve 5 pages to the front end
Modifications			
If hashing and using bcrypt is too confusing, start by using some encryption system and try converting between encrypted and decrypted text. Introduce bcrypt and how to use bcrypt.hash() to hash passwords. For jwt, introduce consolidating information into a token and sending that out. Think about how much information would be sent if it wasn't tokenized.	When creating the authentication routes, if students are struggling with encrypting and hashing data, it may help to remind them to encrypt the password and data in some manner.	N/a - Work period	Start by serving one or two pages with simple designs. From there, start by adding more pages and thinking about what information should be shown per page.

Potential Misunderstandings			
The concept of hashing may be confusing. The values are not encrypted to be decrypted but rather encrypted to be compared with other encrypted values later. In order to clear up these misconceptions, it would help to provide practice with hashing values using analogies.	Authentication routes may take more information than they need. Register should only take in a username, email, and password. Students may think they need to include the confirmed password for checking the validity, but that isn't required.	N/a - Work period	The frontend isn't connected to the backend yet, so the data shown should be fake dummy data until the backend is connected.
Materials & Resources			
Computers			

Week 5:		
Learning Objectives for the Week	Language Goals	Vocabulary - Route - Request

Students will be able to develop functional code to send data between the front and backend services. Students will be able to connect their authenticated user interfaces with their backend authenticated routes. Students will be able to get information from the backend using a GET request. Students will be able to process the information received from the GET request and show the user. Students will be able to implement the search filtering functionality.	- Students will be able to explain the difference between different types of REST requests.	- POST Request - GET Request	
Mon.	**Tues.**	**Wed/Thurs.**	**Fri.**
Enduring Understandings			

Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.
Essential Questions			
How does data transfer between a website and a server?	How does data transfer between a website and a server?	How does data transfer between a website and a server?	How does data transfer between a website and a server?
Learning Goals & Objectives			
Students will be able to develop functional code to send data between the front and backend services.	Students will be able to connect their authenticated user interfaces with their backend authenticated routes.	Students will be able to get information from the backend using a GET request. Students will be able to process the information received from the GET request and show the user.	Students will be able to implement the search filtering functionality.

WHERETO			
E: Equip students with the ability to send POST requests.	O: Connecting the user interface for the authenticated and the backend route will finalize the user authentication system.	O: Pulling and parsing the information from the backend through a GET request is necessary for the culminating project.	O: Searching and filtering for books are essential aspects of the system.
Activities			
Introduce Axios and work with sending post requests to the backend. Write a test POST request on the backend to process test form data from the front end. Create a form on the front end and send the data to the back end. Check for functional implementations by students, then have students create their Authentication pages.	Review using Axios and useEffect with a POST request from Monday. Introduce the formative where students will connect the Authentication pages with the backend endpoint route.	Write a test GET request on the backend to send dummy data to the front end. Show the data on the front end once the data has been received. Conditionally show the data depending on user input and user events. Introduce the formative where students will call the GET request for the library data and show the user the books.	Add a search bar to the user interface. This should be an input and a button. On pressing “search”, send a POST request to the backend, calling the search functionality. This route should respond with the partially searched books. Process the data received on the front end and show the

			books below the search bar.
Assessments			
	(Formative) Process POST Request	(Formative) Process GET Request	
Modifications			
Start by calling post requests with single values being sent. Once the data is sent, see what the backend sends back. Keep trying different routes until the student is more comfortable with the idea of using POST requests.	Use the Authentication routes that exist and are fully functional. If certain routes don't function correctly, use the routes that do function. Alternatively, to test out the connection between their frontend and backend, it is possible for the teacher to provide a working route for students to test using.	Either use a GET request that was created in the previous week or create a new request for this. If students are confused with receiving data from the GET request due to improper GET requests on their backend, the teacher can provide a functional GET request route for students to use.	Start by setting up the interface with the search bar. Link up the functionality to the backend after. This may avoid some confusion around the frontend searching data from the backend.
Potential Misunderstandings			
The Form must have a location to send the data,	In order to avoid over-rendering the page	A GET request sends data back based on	Without adding the functionality

otherwise the data will stay on the website and not go to the backend server.	and data, students should make sure the “useEffect” function has a second argument (dependency) of “[data !== undefined]”. This will ensure that the data will load and run the code in the useEffect until data is no longer undefined.	URL parameters or queried values. GET requests don’t have a request body, but use request parameters and queries.	connecting the frontend search button to the backend search route, students may be confused about how to use their search functionality. Double-check each students’ code and make sure their search button connects to their backend route.
Materials & Resources			
Computers			

Week 6:		
Learning Objectives for the Week	Language Goals	Vocabulary
Students will be able to design and create their user interface.	- Students will be able to justify their design decisions for their website interface.	- User Interface - Frontend Design - Interaction - Event

Students will be able to provide and receive criticism towards their design for the library site. Students will be able to reflect on their design decisions, reconsider decisions, and redesign as they see appropriate. Students will be able to finalize their projects. Students will be able to host their project to access it globally. Students will be able to share and present their projects in pairs.		

Mon.	**Tues.**	**Wed/Thurs.**	**Fri.**
Enduring Understandings			
Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which

communicate with each other to serve the user.	communicate with each other to serve the user.	communicate with each other to serve the user.	communicate with each other to serve the user.
Essential Questions			
What are some design decisions that affect the effectiveness of a website for a user?	What are some design decisions that affect the effectiveness of a website for a user?	What are some design decisions that affect the effectiveness of a website for a user?	What are some design decisions that affect the effectiveness of a website for a user?
Learning Goals & Objectives			
Students will be able to design and create their user interface.	Students will be able to provide and receive criticism towards their design for the library site. Students will be able to reflect on their design decisions, reconsider decisions, and redesign as they see appropriate.	Students will be able to finalize their projects. Students will be able to host their project to access it globally.	Students will be able to share and present their projects in pairs.
WHERETO			

O: Finishing the design is critical as it is the final portion of the user interface.	R: Students will provide and receive feedback on their designs. Students can revise and edit their designs based on the feedback. E2: Journal entry reflecting on the design process and feedback received.	O: Final steps to finish it. Steps to host the project will also be given.	E2: Reflect on the entire process of the project. O: Present in groups their projects.
Activities			
Students will plan out the rest of their design, considering other key features they may want to include and how those should look. This plan will start as a rough sketch, before transitioning into coding components of the design.	Students will pair up and share their designs, wherever they are at. Students will provide feedback to each other on their designs. What is great, what could be improved, and what could change? Students will be given time to revise and edit their design implementation.	Work period. Students will spend the period finishing the projects, asking questions or feedback from peers or the teacher, and finishing features. In the last ⅓ of the class, a step-by-step walkthrough of getting the projects hosted on a cloud provider will be given.	Students will share their projects in their groups. They will note what features they included, what features they didn't get to, and what features they wish they could've implemented. Students will ask questions at the end and share things they liked about each others' work. Students will add a new journal entry, reflecting on the entire process, thoughts

	At the end of the class, students will return to their journals and add a new entry. This entry will be around this process, including the feedback received.		they had, features they implemented, features they wish they could've implemented, and any notes they received from their group members after their presentations.
Assessments			
(Summative 3) Frontend Design			
Modifications			
Try drawing the design on paper before trying to convert it into the actual website.	If students don't have completed designs, they can use the designs they've come up with so far. Each student should have some design as they've had portions of the previous period to design something.	N/a - work period	Students will present what they had for the final project at the end of the unit. If they haven't completed the entire project, they can present what they have.
Potential Misunderstandings			

The design should simply be what students want it to look like. It can be as aesthetically pleasing as students want and as practical as they want it to be.	Student criticism should be constructive and support each other.	N/a - work period	N/a - student presentations
Materials & Resources			
Computers			

Assessment Plan

The Logic of Backwards Design with the Six Facets

Stage 1	Stage 2	
If the desired result is for learners to…	Then you need evidence of the student's ability to…	So the assessments need to require something like…
Understand that… ● Websites/Web Applications serve as accessible platforms providing services to users. ● Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise. ● Interactive systems consist of a user-facing front-end and a	Explain… ● Tradeoffs with certain algorithms on program effectiveness and efficiency. Interpret… ● Data on algorithm benchmarks. ● Data on usage between websites and mobile applications. Apply … ● Various algorithms. ● Pseudocode and convert it into functional code. See from the points of view of… ● Users who may use the product Empathize with…	- Implement various algorithms based on the pseudocode given and reviewed in class. Review the execution times for each algorithm to reflect on their implementation and understanding of the algorithms. - Create webpages for the frontend to show the user, based on the data received from the backend. - Implement functionality of the web server by creating generic functions callable in more specific cases, creating a user authentication system, and designing the web interface.

behind-the-scenes backend, which communicates with each other to serve the user. ● The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data. And thoughtfully consider… ● How can a website help serve a community? ● How can we use algorithms to solve non-computational problems?	● The frustrations for users who may be looking for a book but don't know about the availability Reflect on… ● The implementation and efficiency of their code.	

● How does data transfer between a website and a server? ● How is data stored and accessed by a user through a program? ● How do algorithms impact the efficiency of a program? ● What are some design decisions that affect the effectiveness of a website for a user?		

Formative Assessments

Assessment Type	**EQs Addressed**	**EUs Addressed**	**Evidence of Learning**	**Reflection & Revision**	**Timing & Frequency**
Sort: Given an object of values and a sort key, sort the	How do algorithms impact the	Algorithms provide the building blocks for creating systems	Given an unsorted object and given pseudocode,	Students have the opportunity to make the code more	This assessment would occur at the end of the first

dictionary. Implement two sort algorithms.	efficiency of a program?	and services that can help solve a wide range of problems, computational or otherwise.	students can implement two sorting algorithms to sort the objects. Students can interpret the pseudocode and use it as a guide to developing the two sorting algorithms.	efficient and update their code.	week. Leading up to this assessment, students would have a review of various algorithms covered already. A review of interpreting Pseudocode will also be done prior to this assessment.
Search: Given a search key, search for the value in an object or return None. Implement two search algorithms. Generic search.	How do algorithms impact the efficiency of a program?	Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.	Given an unsorted object and given pseudocode, students can implement two searching algorithms to find the values in the object. Students can interpret the pseudocode and use it as a guide to developing the two search algorithms.	Students have the opportunity to make the code more efficient and update their code.	After the Sort algorithm assessment, search algorithms will be reviewed. A review of how the algorithms function and the pseudocode for the sorting algorithms will be given. This formative will then be given to the students two lessons after the Sort algorithm

					formative. (Midway through Week 2).
Implement a Partial Search to filter items - generic partial search.	How do algorithms impact the efficiency of a program?	The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.	Given an object with data and a key to search, students can develop code to implement a partial search function. This partial search would take a search query and find values that contain that query but aren't exactly the same. Students can utilize prior code to assist with implementing this.	Students have the opportunity to make the code more efficient and update their code.	This assessment would happen at the end of the second week. We will discuss the idea of implementing a partial search algorithm. This will provide students the functionality of implementing an algorithm of searching for queries (books) based on parts of the title, author, and other aspects.
Serve at least 5 different pages to the frontend.	How can a website help serve a community? What are some design decisions that affect the	Websites/Web Applications serve as accessible platforms providing services to users.	Students can extend the front end to show multiple web pages based on the URL and location requested by the user.	Students have the opportunity to make the code more efficient and update their code.	After the fourth week, with the completion of the backend, the frontend will begin being worked on. This is to serve as a check-in and ensure

	effectiveness of a website for a user?				students are on track.
Using a Form, send a POST request and handle the updated UI after the POST.	How does data transfer between a website and a server?	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.	Students can add functionality to their website that takes information from a form and sends it to the backend.	Students have the opportunity to make the code more efficient and update their code.	Students will implement a POST request on the backend and send a Form response to the backend. This will happen right after the frontend design. (Start of week 5).
Send and handle a GET request from the backend	How does data transfer between a website and a server?	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with	Students can develop the functionality for the front-end website to request information from the back end, process the information, and	Students have the opportunity to make the code more efficient and update their code.	Students will implement a GET request on the backend and send the library book records to the frontend. This will serve as a check for

		each other to serve the user.	show it based on what was received.		understanding with students and how to work with the backend and frontend. (Middle of Week 5)

Formative Assessment Rationale

The first two formative assessments will serve as a check for understanding the algorithms previously covered. Students will demonstrate their understanding of interpreting pseudocode and applying the concepts to implement the algorithms. While these two could be combined into one formative assessment, there could also be used as two separate formative assessments. If these two are separate, the second formative assessment on Searching could serve as a reflective formative assessment since it would be similar enough to the first formative assessment. The partial search formative assessment would allow students to apply the concepts of searching for multiple different components rather than an exact search. This would allow students to demonstrate their application of foundational concepts such as strings, conditional statements, and loops. The fourth formative assessment moves towards the front-end component, where students would demonstrate their understanding in applying the design elements covered in the week leading up to the formative assessment. The fifth and sixth formative assessments may be combined since they are similar, but in keeping them separate, students can demonstrate their separate understandings of the two different types of requests for the backend. These would demonstrate their understanding of applying the concepts of communicating between the frontend and backend services.

Summative Assessments

Assessment Type	**EQs Addressed**	**EUs Addressed**	**Evidence**	**Previous Assignments**	**Follow-Up Assignments**
File Parser: Write multiple file parsing functions based on a generic file parser. The generic parser should take a file and rerun the contents. Each specific parser should use the generic parser and arrange the data into a usable format.	How is data stored and accessed by a user through a program?	Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.	Students can write code to generalize reading and parsing data from a file. Students can take the generalized function and extend its capabilities into more specified functions, to cater to more specific needs.	Sorting, Searching, and Partial Search formatives will happen before this.	After this, students will implement this code into their backend server being written and developed concurrently. This will lead up to the User Authentication summative.
User Authentication: Develop a functional user	How is data stored and accessed by a user through a program?	Interactive systems consist of a user-facing front-end and a	Students can develop a functional	Students will have multiple in-class exercises around encrypting data	Writing routes to use the backend

authentication system, registering and logging in a user. The system should store and keep track of the current user. Logging out writes to that user file. Registration creates the user file. The user file contains a username and hashed password.	How does data transfer between a website and a server?	behind-the-scenes backend, which communicate with each other to serve the user.	authentication system (logging in and registering a user) using the understanding and guidance from the classes leading up to this. Students can read and write the user information into a file to save the data between sessions.	using a hashing module.	functionality on the frontend side.
Frontend Design	What are some design decisions that affect the effectiveness of a website for a user?	Websites/Web Applications serve as accessible platforms providing services to users.	Students can design the front end using various design elements explained in class.	Upon completing the backend service, students will spend time designing the front end. While students will have time to discuss and reflect on their designs, the initial development will	Students will continue designing their interface and combining the frontend service with the backend service.

				serve as a summative.	

Summative Assessments Rationale:

There will be three main summative assessments, but all of them build upon developing the culminating assessment. The first two will develop the backend functionality and the last assessment will develop the frontend functionality. The first summative assessment will have students implement code to read, write, and interact with files stored on the computer. This will end up being the foundation for reading and writing data in the culminating backend summative assessment. The second summative will develop the user authentication system which will keep track of users, register and log in users, and track useful information throughout the program. The last summative will start with the implementation of the frontend interface to allow the user to interact with the service itself. This third summative assessment requires students to have completed the backend services already, therefore it must come after the prior two. In terms of understanding, the first summative assessment will demonstrate their ability to interpret data through files, apply various algorithms and programming concepts, and reflect on their understanding of the foundational concepts. The second summative assessment will allow students to demonstrate their ability to apply their understanding of hashing and working with modules, along with applying their knowledge from the file parsing implementation. The third summative will demonstrate their understanding to apply various design principles, explain where certain elements go, and interpret tradeoffs between the placing of various elements.

Culminating Assessments

Title	**Assessment Type**	**EQs Addressed**	**EUs Addressed**	**Evidence**	**Reflection & Revision**	**Timing**
Library Backend	This is the backend system for the Library website. For this component, students will develop the backend of the website. This means they will develop a server that will read the data from a file and create a simple structure to handle that data. This server will send the data to the endpoint upon being called. That is the base	How is data stored and accessed by a user through a program?	Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise. The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and	A set of requests will be sent to the backend, testing the received data from each endpoint. Student understanding can be seen based on the values returned and whether each endpoint returns the correct data.	Students will be given a benchmark score for certain endpoints, they will then be asked to reflect on their times and how they could potentially make the response time more efficient. Students will also have the opportunity to revise their code to accommodate	Both aspects of the project will be introduced in the first weeks of the unit, but this part will be focused on more in the beginning.

	requirement for this part. Students will also have to add additional features to the server.		the size of the data.		these efficiency changes.	
Library Frontend	This is the frontend system for the Library website. This assessment focuses on the website's appearance and functionality. The assessable aspect comes from the design and whether components are accessible and easy to use.	What are some design decisions that affect the effectiveness of a website for a user? How does data transfer between a website and a server?	Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.	Students can effectively create the front end and serve the pages to the user.	Students will be allowed to give each other feedback on their design and implement any changes.	This aspect will be introduced with the backend part, but it will be brought back into focus after the back end is completed.

Culminating Assessment(s) Rationale:

This culminating assessment has two main components. The first part is developing the backend and the second part is developing the front end. This is important as a website contains both a frontend and backend service communicating with each other. Without the backend, a frontend doesn't have data to serve and show the user. Without the front end, the user doesn't have an interface to interact with. In developing both of these, students demonstrate a strong understanding of how to apply the content they've learned. The algorithms used in the backend to assist with sorting and searching the data, filtering books, and working with user account information will demonstrate how they understand the Enduring Understanding of "Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise." In seeing the benchmark times for various endpoints, how students reflect and update their code to improve efficiency will demonstrate the Enduring Understanding of "the way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data." This is because they would demonstrate their understanding of factors that impact efficiency.

Rubric

<table>
<tr><th colspan="2">User Authentication Rubric Overview</th></tr>
<tr><td colspan="2"><b>Assessment associated with rubric …</b>
The rubric is associated with the User Authentication summative assignment. Students will be asked to implement a simple authentication system, that takes in a username, email, and password. When taking in the password, the implementation should encrypt the password, securely storing the password. This rubric will be given during Week 4.</td></tr>
<tr><td colspan="2"><b>Learning Goals associated with rubric …</b>
Students will be able to implement a functional user authentication system through the backend service.</td></tr>
<tr><td><b>EQs associated with rubric …</b>
How is data stored and accessed by a user through a program?
How does data transfer between a website and a server?</td><td><b>EUs associated with rubric …</b>
Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.</td></tr>
</table>

User Authentication Summative	Advanced	Proficient	Nearing Proficient	Novice
Password Encryption Method	Password is encrypted using a built-in hash module. Encryption implementation is safe and secure, storing only the hashed password, not the plain-text password.	Password is encrypted using a manual encryption algorithm. Password is stored using the encrypted password and not the plain-text form.	Password is encrypted using a manual encryption algorithm. Password is stored in plain text.	Password is not encrypted using a hash or any encryption method. Password is stored in plain-text format.
Inputted Data into Requested Routes	Username, email, and password are the only inputs taken into the login and signup routes. No other data is taken, avoiding redundancy.	Username, email, and password are input into the login and signup routes correctly. Other additional data is taken in.	Username, email, and password are input into the login and signup routes incorrectly, either through the response or not through the request body. Other additional data is taken in.	Username, email, and password are not taken into the login and signup routes.
Tokenized Data	Only the authenticated user and the login credentials are tokenized and returned upon successful authentication. No additional data is present in the token.	The authenticated user and the login credentials are correctly tokenized upon successful authentication. Additional data is present in the token.	Either the authenticated user or the login credentials are correctly tokenized upon successful authentication. Additional data is present in the token.	User data is not tokenized and no token is returned upon successful authentication.

Error Handling	All cases of failing to log in or register a user return errors. These cases include: registering a pre-existing account, failing to log in the user based on the username, failing to log in the user based on the email, the email already being taken when registering, the username already being taken when registering, and a password that fails the requirements when registering an account.	Errors are returned when an attempt to register a pre-existing account occurs, a login fails due to the username or email being incorrect, or when the password fails the registration requirements.	Errors are returned in the case of either registering an account that already exists or a login failure. Other errors are not checked for.	Fails to return an error when the user fails to log in to their account or when registering an account that already exists.

ELL Inclusion Plan

ELL Content Standard:

- 11-12, 1, A, 1. Exchanging information and ideas with others through oral collaborative discussions on a range of social and academic topics.

CSTA Content Standard:

- 3A-AP-15: Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.

Standard Unpacked:

Key Nouns – control structures, tradeoffs, implementation, readability, program performance, benefits, drawbacks, choices

· **Big Ideas:** Share information and ideas through discussions, providing feedback around implementation,

Key Verbs – Justify, selection, explain

· **Key Performance Assessment:** Share their implementation of the various algorithms and exchange thought processes between students.

Knowledge and Skills called for by standard – Conceptually understand the algorithms, exchange and share individual implementations and knowledge around increasing efficiency and improving code.

Activities to support the development of the standard – Guided discussions around providing feedback to one another around their implementations, in favor of generating more efficient code.

Essential Questions – How do we generate effective solutions by discussing implementations of code and their efficiency?

What are your language goals for your unit? Why these?

- Students will be able to explain the benefits of developing a website to solve a problem.
- Students will be able to explain the steps of pseudocode.
- Students will be able to share their implementation of code and explain various steps of an algorithm.

- Students will be able to provide feedback to other students around their implementations, with efficiency and effectiveness in mind.
- Students will be able to explain the steps of reading files and writing out to files.
- Students will be able to describe their authentication implementation.
- Students will be able to explain the difference between different types of REST requests.
- Students will be able to justify their design decisions for their website interface.

What are some weekly language goals? Why are these goals and how are they tied to the other goals you have for the unit?

Week	Goals
1	- Students will be able to explain the benefits of developing a website to solve a problem. - Students will be able to explain the steps of pseudocode.
2	- Students will be able to share their implementation of code and explain various steps of an algorithm. - Students will be able to provide feedback to other students around their implementations, with efficiency and effectiveness in mind.
3	- Students will be able to explain the steps of reading files and writing out files.
4	- Students will be able to describe their authentication implementation.
5	- Students will be able to explain the difference between different types of REST requests.
6	- Students will be able to justify their design decisions for their website interface.

How will you assess these goals? What will you do to allow for reflection and revision?
These goals will be incorporated into their weekly journal entries, providing a guide for reflecting on their learning and allowing them to practice their English. I will assess these when I read their journal entries to check for understanding and how well students are meeting these goals.

How might you include student voice in the construction of language goals and their assessments?
By reading their journal entries, I will adjust my instruction to support their learning. This may look like making the information into more bite-sized pieces and keeping the language simple. I

would maintain the language, but make the definitions simpler, supporting their acquisition of the language.

How will you modify assessments (oral, written, or otherwise) for EL students?
As the assessments are around implementing code, if they get stuck and struggle to debug their code due to the language demands of understanding error messages, they could submit what they have. Their assessment would be graded based on their process and how close they got to a correct implementation, ignoring the errors.

How will you activate relevant student background knowledge? Will you use oral, written (a combination), or something else to do this? Why this strategy and when/how often do you plan to use it?
With the sorting and searching activities, I would bring up contexts that demonstrate both algorithms being used. Sorting and Searching through Cards may not be relevant enough to students, so I would bring up contexts that may be more relevant. For example, searching for a movie to watch on Netflix or a video on YouTube, sorting papers, or sorting and searching through songs on Spotify.

List key terms, words, idioms, and/or phrases (TWIPs) to be pre-taught. Include simple, student-friendly, definitions.

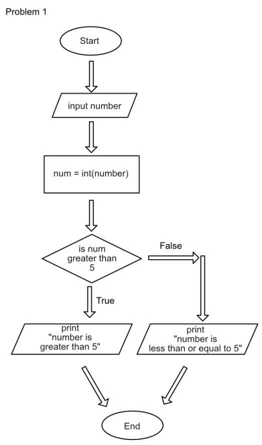
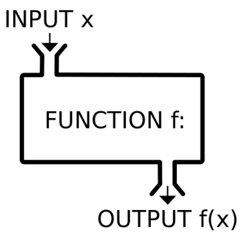
Pseudocode: A way of outlining and organizing the steps of code.
Function: A group of code callable by a name.
Algorithm: A process or set of codes followed one after another.
Collection: A group of items.
Sort: An algorithm that puts items in a collection in order.
Search: An algorithm that finds an item in a collection.
Recursion: A function that calls itself.
User Interface: The portion of a software program that the user interacts with.
Frontend/front-end/front end: The user interface portion of a program.
Backend/back end: The behind-the-scenes portion of a program that processes data.
Request: The method of communication between the front end and back end.

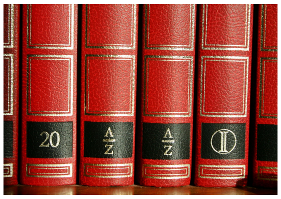
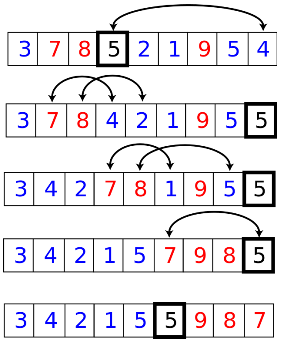
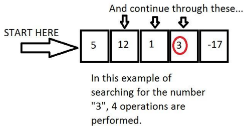
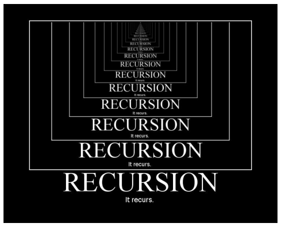
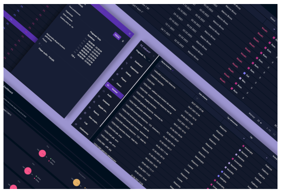
How will you modify texts for beginning, middle, or advanced English learners?
I would keep the language simple to accommodate the students who are beginning to learn English. To try to support the development of their ability to debug code, I would compile a list of common errors and try to break down these with students. Ideally, this would happen at various points in the unit as similar errors appear across the classroom.

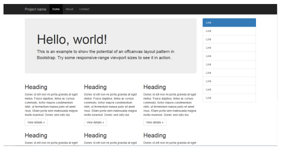
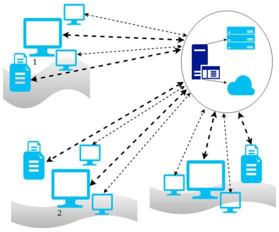
What homework or extended practice will you assign (if any)? How will this include EL support/practice?

To support students with debugging and developing a stronger understanding of finding and moving bugs in code, there will be a couple of worksheets throughout the unit that will include rewording common errors into more student-friendly language. These worksheets will include an error with a description of what caused the error and what a possible solution is. Students will have to write the error in their own words. This should help students develop a stronger understanding of where these errors come from and how to resolve them. Additionally, these worksheets should help students who are developing their English proficiency with understanding and interpret the error messages. Interpreting error messages can be difficult, but they are key to resolving these errors. That is because, to resolve these errors, students must understand what the error messages mean. Writing these error messages in their own words should help facilitate their understanding of what these messages mean and allow students to understand how to fix them when they come across these errors while programming.

Language Study Guide*:

Word	Definition	Example	Picture
Pseudocode	A way of outlining and organizing the steps of code using shapes.	The *pseudocode* of a simple program is to the right.	 Problem 1 Start input number num = int(number) is num greater than 5 False True print "number is greater than 5" print "number is less than or equal to 5" End
Function	A group of code callable by a name.	A large portion of the callable code written has been in the form of *functions*.	

Algorithm	A process or set of codes followed one after another.	Most of the code written in previous units is *algorithms*.	
Collection	A group of items.	A list is a *collection* of items. A set is a *collection* of unique items.	
Sort	An algorithm that puts items in a collection in order.	When we want to organize cards based on their suit or value, we *sort* these cards.	 3 7 8 5 2 1 9 5 4 3 7 8 4 2 1 9 5 5 3 4 2 7 8 1 9 5 5 3 4 2 1 5 7 9 8 5 3 4 2 1 5 5 9 8 7
Search	A way of looking for information.	When we look for information, we are also *searching* for that information.	 And continue through these... START HERE 5 12 1 3 -17 In this example of searching for the number "3", 4 operations are performed.
Recursion	A function that calls itself.	The Fibonacci function is a common function written by using *recursion* as it calls itself.	
User Interface	The portion of a software program that the user interacts with.	A lot of the programs we've written don't include a *user interface*, but instead work off of inputting commands.	

Frontend/front end	The user interface portion of a program.	The *frontend* of Google Classroom could look better.	 Project name Home About Contact Link Link Link Link Link Link Link Link Link Link View details » View details » View details » Hello, world! This is an example to show the potential of an offcanvas layout pattern in Bootstrap. Try some responsive-range viewport sizes to see it in action. Heading Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui. Heading Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui. Heading Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui. Heading Donec id elit non mi porta gravida at eget Heading Donec id elit non mi porta gravida at eget Heading Donec id elit non mi porta gravida at eget
Backend/back end	The behind-the-scenes portion of a program that processes data.	Handling the user data happens on the *backend* of a website instead of the frontend.	
Request	The method of communication between the front end and back end.	When we log into a website, we send a *request* with our data to the backend portion of the website.	

*** See sources listed below for image and definition sources.**

Detailed Lesson Plans 1 & 2

Detailed Lesson Plan Overview

Desired Goals and Accomplishments:

- Students will be able to compare the efficiency of three different sorting algorithms.
- Students will be able to interpret the pseudocode of two search algorithms.
- Students will be able to implement two search algorithms.

Context Within Greater Unit:

- Search and Sort algorithms are key features of the Cumulative project, where the user will be able to search, filter, and sort books from the library website.
- The ability to read pseudocode is also important as it allows for a guide towards how algorithms are written, providing a foundation for visualizing the algorithmic process.

Standard:

Computer Science Teachers Association:
3A-AP-13: Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

Standard Unpacked:

Nouns Used – Prototype, computational problems, prior student knowledge, personal interests

Verbs Used – Use algorithms, by leveraging prior student knowledge, connecting ideas to personal interests

What students need to know – Use algorithms and code to solve computational problems.
What students are able to do – Create a prototype of a product that develops a personal interest or supports their community.

Non-Content Goals:

- This lesson revolves around implementing sorting algorithms. However, I want students to consider the tradeoffs of each sorting algorithm as they implement them. In thinking about the tradeoffs, the hope is that they can develop their critical thinking skills even more, considering when it is better to use one algorithm over another. Outside of the coding aspect, I also want students to start thinking about the decisions they make and how there are consequences and downsides for any decision made.

Stage 1 - Identify Desired Results	
Unit EQs: - How can we use algorithms to solve non-computational problems? - How do algorithms impact the efficiency of a program? .	Unit EUs: - Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise. - The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.
Detailed Lesson EQ: - How can Merge Sort improve sorting performance? - What are some situations where Sequential Search is a better choice over Binary Search?	Detailed Lesson EU: - Merge Sort is a more efficient sorting algorithm due to the algorithm splitting the data into halves, remerging in a linear manner. - Sequential Search and Binary Search have their use cases, with Binary Search having a base requirement of the data being sorted already. Sequential Search functions as expected regardless of whether the data is sorted or not.
Key knowledge: Students will know … - The use cases for Merge Sort. - The tradeoffs with Insertion Sort and Bubble Sort versus Merge Sort.	Skills: Students will be able to … - Read and Implement algorithms from Pseudocode. - Implement Merge Sort, Sequential Search, and Binary Search algorithms.

- The use cases between Sequential and Binary Search. - How to interpret runtimes.	- Analyze the efficiency of various algorithms based on their respective runtimes.
Real-world applications: - Searching and Sorting algorithms are two of the most common algorithms when working with data. Binary Search is one of the most efficient searching algorithms, given data is pre-sorted. Merge Sort is one of the most efficient sorting algorithms. Both of these are found in abundance in various projects within the real world. Merge Sort is also one of the two most commonly implemented sorting algorithms built into programming language libraries. -	

Stage 2 - Determine Acceptable Evidence	
Learning Goals: - Students will be able to compare the efficiency of three different sorting algorithms. - Students will be able to interpret the pseudocode of two search algorithms. - Students will be able to implement Sequential Search and Binary Search algorithms.	Evidence of Learning: - Students can elaborate on the differences between Insertion Sort, Bubble Sort, and Merge Sort both procedurally and conceptually. - Students can write the code for both search algorithms based on the pseudocode, demonstrating their conceptual understanding of the pseudocode. - Students can justify the benefits and use cases of both search algorithms demonstrating a conceptual understanding of both algorithms.
Opportunities for Student Self-Assessment and Reflection: - Upon developing and running the code, students will be given a range of expected run times. They will have the opportunity to reflect on the process and revise their implementations if it falls outside of the expected ranges.	

What understandings or goals will be assessed through these tasks?

- Given Day 2 will include the Formative Assessment on implementing Binary and Sequential Search, student understanding of implementing pseudocode will be assessed.
- Additionally, students' self-reflection on their implementations of the algorithms based on the provided pseudocode will demonstrate their understanding of code efficiency and how to make code execute more or less efficiently.

What products and performances will provide evidence of desired understandings?

- The production of functional code, through functional implementations of Sequential and Binary Search will provide evidence of the desired understanding.
- Students' self-reflection journal entries will also provide evidence of their understanding of developing efficient code, through seeing more efficient implementations.

By what criteria will student products and performances be evaluated? If you are using a rubric, what are the categories you will be using?

Considering the bulk of the student work in these two lessons revolve around implementing certain sorting and searching, the criteria for evaluation will be around how functional their implementations are. If the code doesn't behave as intended, I will direct students to return to their code and try to fix or resolve the problems causing the code to function unexpectedly or unintentionally. Since these criteria are based on how effective/functional the code is, the criteria will be: "Implementation is functional without errors and behaves as intended, Implementation is functional but with errors or behaves as unintended, or Implementation is not functional." These three statements will provide the standards for measuring and evaluating the code produced by students.

Stage 3 - WHERETO	
W = Help the students know **W**here the unit is going and **W**hat is expected? Help the teacher know **W**here the students are coming from (prior knowledge, interests)?	Searching and Sorting algorithms are critical early components of the Library Cumulative project. By implementing these algorithms now, they serve as a connection to the prior knowledge and segue into seeing these algorithms used in a proper and functional application.

H = **H**ook the students and **H**old their interest?	Day 1: Start with a review of Insertion Sort and Bubble Sort from Friday's lesson. Day 2: Start with reviewing recursion and writing recursive functions.
E = **E**quip students, help them **E**xperience the key ideas and **E**xplore the issues?	Day 1: Merge Sort review. Day 2: Review pseudocode for both search algorithms. Review writing recursive functions.
R = Provide opportunities to **R**ethink and **R**evise their understandings and work?	Reflect on the run time difference between Insertion Sort, Bubble Sort, and Merge Sort. Revise their algorithms and implement Merge Sort based on the Pseudocode.
E = Allow students to **E**valuate their work and its implications?	N/a
T = Be **T**ailored (personalized) to the different needs, interests, and abilities of learners?	Break down the algorithms into their subcomponents, providing students with the opportunities to approach the steps in a manner that makes sense to them.
O = Be **O**rganized to maximize initial and sustained engagement as well as effective learning?	Both search algorithms will be used in their backend service.

Stage 4 – Lesson Map Day 1	
Teacher Moves	**Student Moves**
Hook/Activating Prior Knowledge Cards will be placed on a table out of order. The pseudocode for both Insertion Sort and Bubble Sort will be posted on the board. The instructions for students will be on the board, where they will be asked to sort it with both algorithms, reshuffling the cards before each sort.	Students will sort the cards based on the steps from both algorithms. Students will first sort it with Insertion Sort. After finishing, they'll shuffle the cards and sort them again with Bubble Sort. After sorting the cards using both algorithms, students will write down in their journals which seemed faster. After they finish writing down

	their entry, they will open their computers and pull up their code from before.
Review the code from Friday’s lesson on implementing Insertion Sort and Bubble Sort. The teacher will ask students to bring up their code and look through their code.	Students pull up their code and review how they implemented the algorithms while looking at the pseudocode to recall what they did before.
The teacher will provide a range of expected execution times. These expected times will be based on running the code on different systems and finding the average times.	Students will run their code and compare the execution times with that of the expected ranges.
The teacher will provide students with 5 minutes to reflect on their code implementation. The teacher will post guiding prompts to help guide students through the reflection. These prompts will include, “My run times were… This was because …”, “I used _ loops because …”, “I have _ variables and _ lists because …”, and “I can improve my code by …”.	Students have 5 minutes to reflect on their code implementation and how they could improve it. Students will use the guiding prompts to support their reflection. Students aren’t restricted to using the prompts as they can also freely reflect on their code. The prompts are only there to guide student thinking.
Introducing the Activity The teacher will reintroduce Recursion and post some pseudocode for writing Recursive functions. The teacher will implement Factorial iteratively and recursively.	Students will take notes on concepts around Recursion, such as Base cases, Recursive cases, and backtracking. Students will follow along with the implementations of Factorial.
The teacher will provide students with some exercises to practice writing recursive functions.	Students will work through the exercises, implementing the recursive functions for 15 minutes.
Activity The teacher will introduce Merge Sort and demonstrate the code running. The teacher will also show the console at each split and each merging point.	Students will take notes after seeing the code running and seeing the pseudocode. Students will implement Merge Sort in pairs to think

Reflection The teacher will provide execution/run times and allow students to compare their run times with the expected run times. *Closing* The teacher will review Searching algorithms at the end, reminding students that the next lesson will be an implementation of search algorithms.	through and collaborate through the process. They will be given 15 minutes for this. Students will compare their execution times with the expected times. Reflecting on the process and what went right/wrong. Students will take notes on search algorithms and how they work. These notes will revolve around the steps done by the algorithm and how the algorithm checks if the current value is the queried value.
Materials & Resources*	**Differentiation and Modifications**
Computers	If the recursive functions appear to be too complicated or difficult, students can approach them iteratively - using loops - instead. This may help students with thinking through the code first before moving into writing it recursively. With Merge Sort, students could focus on finding the recursive cases and the base cases first, considering when the function should recurse and when it should end.
Stage 4 – Lesson Map Day 2	
Teacher Moves	**Student Moves**
Hook/Activating Prior Knowledge The teacher will post the code of four different recursive algorithms. The teacher will ask students to identify the base case, recursive case, and what each recursive function does. The teacher will also ask students to identify the algorithm with a clear bug that will cause the code to crash.	As students enter the class, four different recursive algorithms will be on the board. Students will identify the base case, recursive case, and supposed functionality of each recursive function. One of the algorithms will have an error with the base case, causing the program to crash if it was run. This is because that specific algorithm would never hit the base

The teacher will then debrief these recursive algorithms, asking students to explain their thinking. After identifying the key information of each recursive algorithm, the teacher will take a poll from the students asking which algorithm will crash and cause a Stack Overflow. After students identify and choose the problematic algorithm, the teacher will reveal which one is the problem algorithm. The teacher will then ask students to explain the error and provide a solution to the algorithm. The teacher will review the recursive algorithms from the prior lesson, providing students with some example implementations. *Introducing the Activity* The teacher will introduce Sequential Search and Binary Search. This will be done through a Card search activity. The teacher will give cards to each table group. The teacher will then post both algorithms on the board, walking through the steps done in Sequential Search. *Activity* The teacher will provide students with time to use both search algorithms to find a specified card. Upon using the Sequential Search method, the teacher will direct students to use the Binary Search method. The teacher will introduce the pseudocode	case. Students would have to find the algorithm with this bug. After the teacher debriefs the information from each recursive algorithm, students will raise their hands on which algorithm they think the error is in. Students will turn and talk about their reasons and come to a conclusion about which algorithm is problematic. Once the problem algorithm is revealed, students will explain the error and pitch a solution to the problem, resolving the Recursive Error. Students will compare their implementations with the example implementations. Students can tweak and change their implementations, but they will explain their reasoning for how they implemented the algorithms from the exercises. Students will be given a set of unsorted cards and asked to search for a given card in that set. As the teacher goes through both algorithms, students will take notes on how Sequential Search works. Using the steps from Sequential Search, they will try to search for that card. Students will then be given the pseudocode for Binary Search and asked to take notes on how the algorithm works. Students will then sort their cards. Once they sort their cards, they will be tasked with searching for a specified card using the Binary Search method.

for Binary Search and walk through how it works. After giving students time to work with using a Binary Search approach to finding a card, the teacher will check in with students, clearing up any misconceptions that may have come up. Once all questions have been answered, the teacher will introduce the Formative Assessment, tasking students to implement both algorithms using code.	After practicing using Binary Search to find a card, students will check in with the teacher and ask any questions that they may have. Students will be given the remainder of the class period to implement the search algorithms in code.
Materials & Resources*	**Differentiation and Modifications**
Cards Computers Writing Utensils	With the card activities, students can start with smaller hands to try to practice. As they understand the procedures better, they can increase the size of hands/number of cards they're working with. This may help provide a stronger and deeper understanding for each student.

Reflective Log 3: Unit Self-Assessment

1. What do you think the overall strengths of your unit are?

I think the unit has two main strengths. The first is the cumulative assessment which is relevant and interesting for students, and the second is the scaffolding of activities throughout the unit. The culminating assessment has students developing their first major application-driven project, creating a website for a library, loaded with basic features and student-driven features. Students have the opportunity to add features they care about and are interested in adding. This opportunity allows students to make it more personal and more relevant to them. With the second point, the activities and assessments are set up in a manner that helps students succeed throughout the unit, building up to this cumulative assessment. For example, reading and working with pseudocode at the beginning of the unit helps students visualize the code and see how the code behaves without writing the code. Additionally, each assessment has students develop and build out a key component of the culminating assessment allowing students to have a strong start toward creating the library website.

2. Where do you think you will need to work to improve?

I felt that the ELL Inclusion Plan could've been improved and more effective. Computer Science itself is not the most inclusive or supportive for English Language Learners. I've complained about this and I struggled with thinking of a plan that could support students with both developing their English language skills and writing effective code/debugging their code. This is an issue due to error messages found in code requiring a degree of understanding of English. While translating the error messages is possible, the meaning of the message can get lost in translation and make little sense. Therefore, I tried to create a plan that helps students translate these messages into words and language that makes sense to them. However, I'm not sure how effective this is in practice as I haven't had an opportunity to try it out in the classroom. Because of the inherent lack of support in the Computer Science space for English language learners, I felt like the ELL Inclusion Plan could be better, but I also couldn't find resources that were actually helpful in creating an effective plan.

3. What skills, knowledge, tools, and understandings from this project will you carry with you to your future teaching?

From this course and project, there are three main ideas that I think I will carry with me going forward. The first is backward planning in its entirety. When I was in my Undergraduate fieldwork course, Kory O'Rourke taught us how to work with UbD and work with backward planning. However, I hadn't gone so in-depth with the process. This process definitely helped a lot with thinking through and planning out the unit plan. I know I will continue to use this in the future regardless of what setting I am teaching in. The second main idea is creating effective Enduring Understanding and Essential Questions. My practice in developing these EUs and EQs was quite limited before and after our exercise in making them more open-ended, I felt more confident and comfortable creating them. Lastly, my third main idea is the difference between understanding and knowing. When I was in my C&I course, we talked about Conceptual versus Procedural Understanding and that changed how I thought about what I knew. However, this course made me reconsider that even further with what I understood and what I knew. I know I will leave thinking more about how to develop student understanding and not just student knowledge.

4. What do you hope your students come away with once you teach it?
Considering this unit plan revolves around applying foundational concepts and knowledge, and seeing where the concepts are used in regular day-to-day programs, I would hope my students would take away an understanding of this application in the real world. A lot of the programs developed in a normal programming course don't show the application of these concepts in real-world contexts, or the contexts are so far removed from students that it doesn't seem relevant. However, I hope that after developing this library website, students would be able to recognize where the algorithms, loops, and various other concepts are used within the real world. Additionally, I would hope students can be proud of what they have learned and show their learning to others through this project. Since at the end of the unit, students are taught how to host their project, I would hope they would show it to others as it demonstrates their learning.

5. What are some aspects of the unit that you would want to share with other teachers?
Despite the ELL Inclusion Plan being a weaker point of my Unit Plan, I think it's something I would want to share with other teachers, especially in Computer Science. I say this because I know how big of a challenge it is for students who are learning English as another language. Additionally, I know how difficult it is to plan and develop effective lessons as an instructor to both support their acquisition of English and apply their understanding in coding - an exercise that is already so English-heavy. I would also share it with other teachers to get their feedback and hear what they would do, what they have done, and what they would change. I feel like it's the part that could be improved the most and to do so, it would be most productive to be improved by other teachers who are regularly teaching Computer Science already.

6. Anything else that you think would help explain your thought process?
While this unit plan shifted from my original plan at the start of this course, I am really happy with how this plan has turned out. I think I would still want to develop a unit plan around my original idea of students choosing their projects, but this unit plan seems like a really good foundation for students to build on. Ultimately, I've really enjoyed this experience and I'm quite proud of how this has turned out.

Appendix 1: Resources Bibliography

Bart, A. C., Whitcomb, R., Riddle, J., Saleem, O., Tilevich, E., Shaffer, C. A., & Kafura, D. (2022). JSON datasets. CORGIS Datasets Project. Retrieved April 6, 2023, from https://corgis-edu.github.io/corgis/json/

- This resource is a collection of datasets representing different types of information. This was used to provide the foundation for the data being used for the library website. While this doesn't provide all of the books for the library, this provides the basic information and some examples to use for the books of the library.

Veersamy, A. (2014, February). Teaching English-based programming courses to English learners/non-native speakers of English. Retrieved February 05, 2023, from https://www.researchgate.net/publication/304024941_Teaching_English_based_programming_courses_to_English_learnersnon-native_speakers_of_English

- This resource is a conference paper on how individuals require a certain level of English competency in order to effectively learn how to program. This gave me some foundation for the ELL Inclusion Plan and helped me consider strategies to develop students' competency in English. This is all in order for students to understand how to use the various keywords found in English-based programming languages. Additionally, this conference paper discusses the various other programming languages created in other spoken languages, such as "Phoneix", "HPL", and "SAKO". This gave me the idea that

teaching programming in a non-English-based language may not be as effective due to the lack of popularity of alternative programming languages.

Weigend, M. (2015, October). Making computer science education relevant. Information and Communication Technology, 53–63. https://doi.org/10.1007/978-3-319-24315-3_6

- This journal article describes the importance of teaching computer science within the context and making the projects relevant for young individuals. While this article didn't explicitly give me an idea of what this culminating project should look like, it did give me the motivation around creating a practical project for students to apply their understanding.

Appendix 2: Reflective Logs 1 & 2

Reflective Log 1 - Unit Topic and Theme

An Introduction to Computer Science or an Introduction to Programming course tends to only be interesting to those who have a direct interest in programming or computing. This is based on what I've seen as I've taught and tutored programming at various levels and toward different age groups throughout the last three years. However, in my experience, there tends to be a changing point in people's approach to programming once they see the application of what they've learned and once they're able to create something personal or something they can be proud of. While it's essential to teach the foundations of programming, I believe once the foundations have been taught, it's essential to look at these applications and begin developing a project that individuals can be proud of, demonstrating their understanding and proficiency, and showing their interests as individuals. These projects should be chosen by students and allow students to explore deeper topics that interest them and go beyond the foundations. Upon planning and starting these projects, students often become more engaged with the content and begin applying themselves more, demonstrating their curiosity and their passion for their projects. Knowing this, for this unit plan project, my topic will revolve around applying the foundational concepts of programming that my students will have learned and reviewed already, and guiding them to create a project they can be proud of.

This guide will span multiple weeks and help students conceive and develop a project that they can be proud of, while also demonstrating their understanding of the content and potentially giving back to their community.

Students will go through the Design Thinking process, thinking of a problem in their communities that they would like to solve, researching solutions that already exist, proposing a potential solution of their own, designing this solution, and potentially implementing this solution. In working through this process, students will develop their critical thinking skills, research skills, presentation skills, and problem-solving skills, and demonstrate the programming skills they've acquired leading up to this. The critical thinking skills will help them think of a problem that their community has. The research skills will help them find solutions that already exist and help them find flaws or issues with pre-existing solutions. The presentation skills will help them explain the problem to others and pre-existing solutions to others. Additionally, students will be able to take in some feedback and learn to provide useful advice to others about their problems. The problem-solving skills will help them consider more solutions and think through possible solutions for themselves. Lastly, their programming skills will help them develop and implement the solutions.

Ultimately, this unit will help students develop skills that go beyond the skills of coding and programming. Students will develop skills that will help them as they continue growing, regardless of what professions they choose to go down. The hope is that this unit may also help students become more interested in STEM as they'll be able to have a project that's personal to them and possibly see the impact on a community.

This unit reflects my educational philosophy as I strongly believe in emphasizing the reasons for the content being taught, why it's important, and where it's applied in the real world. As a Computer Science and Mathematics teacher, I've always been concerned with keeping students interested. My answer to that concern has always been trying to explain the reasons for the content and where it's being applied. As a Mathematics teacher, I've struggled with that a little more. However, with Computer Science, I've always found that part quite easy to do since all of the foundations end up being applied somehow. With this unit focusing on a student-developed project, I think it helps students see for themselves where the content applies in situations that are personal and impactful for themselves. I can always tell them, but I think it's significantly more powerful and meaningful for them to also see and find those applications for themselves–granted with some guidance and support too. Therefore, this unit reflects my philosophical commitment to ensure students have a more conceptual understanding of programming beyond the procedural elements of how to code and find some of the meaning and applications for themselves too. Additionally, I think this unit allows me to also gain a better understanding of my students, where they come from, the communities that they're in, and what's important to them. The element of developing and deepening relationships has always been really important to me. In having students come up with projects that are meaningful to them and give back to their community while being supported and provided guidance throughout, I think I can deepen my relationships with my students and understand them better as individuals too.

Reflective Log 2

For my Introduction to Computer Science unit plan, my Enduring Understandings are the following:

- Websites/Web Applications serve as accessible platforms providing services to users.
- Algorithms provide the building blocks for creating systems and services that can help solve a wide range of problems, computational or otherwise.
- Interactive systems consist of a user-facing front-end and a behind-the-scenes backend, which communicate with each other to serve the user.

- The way algorithms are programmed affects the efficiency of a program based on factors such as the number of iterations, variables, and the size of the data.

My Essential Questions are:

- How can a website help serve a community?
- How can we use algorithms to solve non-computational problems?
- How does data transfer between a website and a server?
- How is data stored and accessed by a user through a program?
- How do algorithms impact the efficiency of a program?
- What are some design decisions that affect the effectiveness of a website for a user?

My culminating assessment will revolve around the final project, where students will create a web application containing an interface and a backend service, allowing users to search the catalog of a fictional library. Before continuing, a breakdown of the project is required for the remainder of this to make sense. When we access a website, we interact with the front-end portion of the site. This front end is everything that one can interact with. This includes the menu, buttons, forms, text inputs, images, the text itself, etc. As this portion is strictly the portion that gets interacted with, this doesn't include the process of logging in a user, handling user data and information, or persisting the data between each period the user accesses the site. Instead all of those processes – logging in, handling and processing user data, persisting data between sessions – get handled by what's considered the backend. Without this backend, a website can't do a lot and simply can only show basic data. This backend service provides the front end with the data to show, it handles logging in and maintaining the user data, and communicating with a database, to name a few uses. When data is stored, it often isn't stored directly on this backend service, but stored within a database system. This database system can vary in complexity. It can be a full-fledged database or a simple text file containing the information. Returning to the backend, this backend would read, update, and write to this database system – regardless of which format it uses – to maintain that data and keep it consistent between users accessing the system. These three systems work in tandem and provide the foundation of a web application and serve as a simplified version of many web services that we often use. Students wouldn't be expected to create a complex project. Instead, they would be expected to create a simplified version of a library catalog system, where users can view what books are in the library, which books are available or reserved, and provide a means of searching, sorting, filtering, and reserving a book. Additionally, students will build a simple login system into the backend, which would handle maintaining the user's information, book reservations, and history of reservations. Students will build the backend first, reading a file and storing the contents of the file in the backend, which will then be used to communicate the data to the front end. Students will then build the front end to read the data and show it in a way that's useful and easy to read. While this project may seem quite extensive, it can be simplified into a couple of main steps that students will explore and work through.

Now that the context and background of the project have been explained, let's connect this back to the Enduring Understandings and Essential Questions. This project demonstrates my first Enduring Understanding of Web applications providing services to users as the students will ultimately develop a full-fledged application that other users can interact with. Students will utilize various algorithms to help construct the system by incorporating searching and sorting algorithms to streamline the user experience. Students will construct both a front and a backend, creating a system that can store data and retrieve data between user sessions. After incorporating the algorithms, students will update the algorithms used to evaluate the impact of certain algorithms on the efficiency of the program. This efficiency will be quantified by how long it takes for the program to load or handle the request.

In terms of the facets of understanding, students will explain how the foundations of programming can be applied to other areas beyond the examples we had seen in class. Students will explain when certain algorithms are used and when some algorithms aren't, additionally explaining the impact on the efficiency of these algorithms. Students will interpret various datasets showcasing different information, such as algorithms and their efficiency, solutions that exist for libraries and their effectiveness, and design decisions that impact the user interface's effectiveness and ease of use. Students will apply the fundamentals of programming and use algorithms to develop a website that helps solve the problem of book availability, book reservations, and tracking user orders. Students will consider the users' perspectives and how they may interact with the website. This perspective will help inform them about how they're designing the interface and how useful certain design elements are. Before creating the interface, students will use websites that have difficult interfaces and empathize with poor design decisions, allowing them to develop more meaningful and practical interfaces for the user. Throughout the project, students will reflect on what knowledge they already have to implement certain features. At the end of the project, there will be an opportunity for students to implement additional features. During this period, students will reflect on what features they want to add, considering what concepts they understand, and how they can approach implementing these features. In their journal entries – which will be further explained later – students will reflect on the process and how they've worked through the various components of the project. This will deepen their understanding and demonstrate their ability to reflect on their knowledge.

With the essential questions, students will be given a problem around a library struggling with cataloging their books and book availability. Students will then be given time to research how a website could support this library and ease the process for librarians and customers alike. This will help students answer the first EQ about how a website can help serve a community. This will also help answer the question of community problems being solved with a technological solution. Students will then develop a website for this library with data coming from a file. As students develop this, they will be asked to consider how libraries sort their books and how books are found within the library. Students will then implement a searching and sorting algorithm into their website to help find a queried book. As students implement their backend service, they will read and write data to a file, using that as the database. Students will learn about various methods to store data, but for the context of the project, they will be required to read and write to a file. This is to keep the content within the scope of their understanding and prevent overcomplicating the material.

Throughout the project, students will include a journal of what they've learned and what they've researched. This will also help serve as an informal assessment, beyond the major project, informing the teacher about where students are at with the process. Additionally, the journal will help show students' understanding of the content and the essential understandings. There will be some journal entry prompts that will help guide student thinking during these journal entries, which will further demonstrate their understanding. The culminating project will demonstrate students' ability to contextualize and apply what they've learned into a system, which further demonstrates their ability to "do" programming. As they work through the project, they will also encounter bugs and errors, which they will learn to fix and resolve. These problems replicate common problems that appear in the real world of programming and solving these problems are important skills for them to develop. Additionally, students will have to consider when and how to use certain concepts they've learned previously when approaching various problems. Lastly, the project will allow students to practice what they've learned, get feedback, and refine the system to be more efficient and effective. Students will demonstrate their understanding of refining the system effectively by how they design the interface, where they place inputs, buttons, and other parts of information. The efficiency will come with considering load times and how to make the system faster or easier to work with. The effectiveness will come from the thought process behind the front end, while the efficiency will come from the motivation behind the back end.

TEC 674 Curriculum Plan Final Checklist S2023

- ☑ ~~**Title Page (**https://owl.english.purdue.edu/owl/resource/560/01/**)** but do not include a running head. Make sure that there are page numbers on all pages after the title page. Include your last name as a header in the right-hand corner of each page.~~

- ☑ ~~**Table of contents with Bookmarks**~~
- ☑ ~~**Unit Introduction (Overview & Rationale)**~~
  - ☑ ~~Brief introduction to your unit and your aims for student learning~~
  - ☑ ~~Explanation of major decisions you made~~
- ☑ ~~**Unit Calendar**~~
  - ☑ ~~Follows guidelines and is complete~~
  - ☑ ~~Along with your EQs and EUs, includes key vocabulary for the unit~~
  - ☑ ~~Includes resources~~
  - ☑ ~~Complete~~
- ☑ ~~**Assessment Plan**~~
  - ☑ ~~Follows guidelines and is complete~~
  - ☑ ~~Includes EUs and EQs~~
  - ☑ ~~Includes Formative Assessments and Rationale~~
  - ☑ ~~Includes Summative Assessments and Rationale~~
  - ☑ ~~Includes description of Culminating Assessment~~
  - ☑ ~~Includes Stage 1 and 2 (see the Assessment Plan guidelines)~~
- ☑ ~~**Rubric**~~
  - ☑ ~~Follows guidelines and is complete~~
  - ☑ ~~Criteria and Performance-level Descriptions~~
  - ☑ ~~Indicates for which assessment it will be used~~
- ☑ ~~**ELL Inclusion Plan**~~
  - ☑ ~~Follows guidelines and is complete~~
  - ☑ ~~Connects to unit calendar~~
- ☑ ~~**Lesson Plan 1**~~
  - ☑ ~~Follows guidelines and is complete~~
  - ☑ ~~Content Standard(s) (unpacked)~~
  - ☑ ~~Language goals~~
  - ☑ ~~Stages 1, 2a, 2b and 3 (WHERETO)~~
  - ☑ ~~Detailed Lesson Map that builds on prior knowledge and puts student engagement and active learning at the center~~

- ☑ ~~Handouts or materials/links to resources~~

☑ **~~Lesson Plan 2~~**

- ☑ ~~Follows guidelines and is complete~~
- ☑ ~~Content Standard(s)(unpacked)~~
- ☑ ~~Language goals~~
- ☑ ~~Stages 1, 2a, 2b and 3 (WHERETO)~~
- ☑ ~~Detailed Lesson Map that builds on prior knowledge and puts student engagement and active learning at the center~~
- ☑ ~~Handouts or materials/links to resources~~

☑ **~~Log 3/Reflection on Unit~~**

- ☑ ~~Follows guidelines~~
- ☑ ~~Revised and edited if necessary~~

☑ **~~Appendices (Resources Bibliography and Process Portfolio)~~**

- ☑ ~~Resources Bibliography with citations for unit plan resources~~
- ☑ ~~Reflective logs 1 and 2~~
- □ In-class Reflections and Feedback optional

Self Assessment

Designer: Edward Rees

Unit Title: Library Website

Unit Overview	**Yes**	**Needs Work**	**No**
Does it discuss the importance of the unit topic/theme in regard to your students and educational aims?	x		
Does it discuss how the unit fits with your curriculum philosophy?	x		
Does it provide a clear overview and reasons for the goals, activities, and assessments you included?	x		

Assessment Plan	**yes**	**needs work**	**no**
Does it clearly describe a culminating assessment (performance or product) that provides evidence of students' development of enduring understandings?	x		
Does the plan include a rubric with clearly indicated criteria for student work that targets understanding and skills?	x		
Does the plan include formative and summative assessments that lead up to the culminating assessment?	x		
Does the assessment plan demonstrate how you will use a variety of approaches to assess	x		

students' diverse abilities and learning processes in fair and valid ways?			
Comments: **With the Assessment plan, I felt that my assessments build up to the culminating assessment. For example, each formative and summative create a component for the major project, providing a snapshot of students' understanding. Additionally, the assessments serve as a means of understanding how to better support students, what topics need additional time to cover, and how to help students more.**			

Resources Bibliography	Yes	Needs Work	No
Does your bibliography provide an APA citation for each of the instructional resources you used for your unit?	x		
Comments: **Since most of the resources were created from memory or projects I've done before, I didn't have a lot of sources to cite.** However, the couple of sources I did use were cited.			

Unit Calendar	Yes	Needs Work	No
Does the calendar include the enduring understandings and essential questions for each week as well as learning targets, activities, assessments, and materials/resources for each day of the unit?	x		
Does the calendar provide a sound instructional sequence that fosters developmental learning?	x		
Comments: **Each element required in the calendar is there. The assessments are spread out in a manner that aligns with where students should be to build toward the culminating assessment.**			

Stage 1:

1. Are the designated understandings “enduring”? Do they represent big ideas at the heart of the discipline and/or important understandings that have enduring value beyond the classroom?	[✓ YES]	Needs Work	Not Yet
2. Do they enable us to make vital and informative connections in our learning?	[✓ YES]	Needs Work	Not Yet
3. Do they represent complex concepts that students will develop through inquiry and construction of meaning?	[✓ YES]	Needs Work	Not Yet
4. Do they offer opportunities for authentic work?	[✓ YES]	Needs Work	Not Yet
5. Do students confront misconceptions, stereotypes, or other misunderstandings of these concepts during the unit?	[✓ YES]	Needs Work	Not Yet
6. Are provocative and guiding essential questions framing and providing perspective for the work?	[✓ YES]	Needs Work	Not Yet
7. Do the questions have multiple answers?	[✓ YES]	Needs Work	Not Yet
8. Are they a doorway to discussion, inquiry, and research? Do they raise other important questions?	[✓ YES]	Needs Work	Not Yet

9. Are important and unit-relevant knowledge and skills *specifically* identified so they provide direction for assessment?	[✓ **YES**]	**Needs Work**	**Not Yet**
10. Do the knowledge and skills state explicitly *what* students will know and do and not just "that" they will know about a particular topic?	[✓ **YES**]	**Needs Work**	**Not Yet**

Stage 1 *Comments/Explanation:*

The enduring understandings represent the big ideas behind the unit. Each understanding represents a key focal point in the unit around websites, algorithms, implementation, and considerations when developing a functional product. Students are able to take their understanding and apply it to the real world, making connections with their learning in the websites and web services they interact with and use. Based on the scheduling of the calendar, students would be able to explore the understanding and develop a stronger and clearer understanding beyond what they have already seen. While the culminating project is scaffolded and built off of an outline, students have the ability to add more features and make it their own product. This provides them the opportunity for authentic and meaningful work that goes beyond writing code and developing code for the sake of it. Throughout the various weeks, the misconceptions and misunderstandings are addressed through various activities or through notes to the instructor to clarify. With each Essential Question effectively matching a week, along with each question being open-ended, students may have different answers that lead to deeper discussion and considerations. Additionally, the rubrics students are graded on are clear and showcase the expectations for students.

11. Do(es) the performance task(s) reflect the key elements of GRASPS; Goal, Role, Audience, Situation, Performance, Standard?	**Extensively**	[✓ **Somewhat**]	**Minimally**
12. Are there appropriate opportunities for	[✓ **Extensively**]	**Somewhat**	**Minimally**

students to exhibit their understandings through authentic performance tasks?			
13. To what extent do the assessments provide: fair, helpful, valid, reliable, and sufficient measures of the desired results?	[✓ **Extensively**]	**Somewhat**	**Minimally**
14. Are there appropriate opportunities for student self-adjustment/revision, based on feedback?	[✓ **Extensively**]	**Somewhat**	**Minimally**

Stage 2 *Comments/Explanation* (Provide evidence here):

I think the performance tasks could've been clearer when using GRASPS. I felt this way as I feel like the role and audience are unclear. The goal, situation, and performance are more clear as they are toward developing a library website or components of the website to solve a problem. Additionally, I felt that the standard portion may not be really clear. However, I feel like it is there and students are able to demonstrate their learning through the assessments and performance tasks. While the majority of the performance tasks are code related, there are some that are around demonstrating their learning through explanations or peer feedback. Therefore, I believe students are able to exhibit their understanding in multiple ways with appropriate opportunities. The assessments themselves are around implementing code, therefore the assessments are naturally helpful, valid, reliable, and measurable. As students have already developed their foundation prior to this unit, the assessments are also fair for students. Lastly, many components of the culminating project have feedback attached to them. Students will give each other feedback, the teacher will give them feedback based on their implementations, and students have opportunities to self-reflect on their implementation and process. Based on their feedback and reflections, students are able to revise their code implementation and make it more efficient. Due to this, I would say there are appropriate opportunities for students to self-adjust their code and revise what they have.

Stage 3:			
15. To what extent is the lesson sequence developmentally appropriate and engaging? Does it build on students' prior knowledge?	[✓ **Extensively**]	**Somewhat**	**Minimally**
16. Will the students know where they're going and why, what they already	[✓ **Extensively**]	**Somewhat**	**Minimally**

know, where that might go astray, and what is required of them? (**W**HERETO)			
17. Will the students be **h**ooked – **e**ngaged in digging into the big ideas? (W**HE**RETO)	[✓ Extensively]	Somewhat	Minimally
18. Throughout the unit, will the students have adequate opportunities to explore/experience key ideas and receive instruction to **e**quip them for the desired performance? (WH**E**RETO)	[✓ Extensively]	Somewhat	Minimally
19. Will the students have sufficient opportunities to rethink naïve or prior ideas; and **r**ehearse, **r**evise and/or **r**efine their work based on timely feedback? (WHE**R**ETO)	[✓ Extensively]	Somewhat	Minimally
20. Will the students have opportunities to **e**xhibit their understanding through final performances and products? (WHER**E**TO)	[✓ Extensively]	Somewhat	Minimally
21. Is the learning plan **t**ailored and flexible to ensure optimal performance and interest for all students? (WHERE**T**O)	Extensively	[✓ Somewhat]	Minimally
22. Is the learning plan **o**rganized and sequenced to maximize engagement and effectiveness? (WHERET**O**)	[✓ Extensively]	Somewhat	Minimally
23. Are the needed materials identified?	Extensively	[✓ Somewhat]	Minimally

Stage 3 ***Comments/Explanation*** **(Provide evidence here):**

Throughout the unit plan, different hooks are used to increase student engagement and get students involved with the content. There are warmups having students manipulate cards, strong starts with coding exercises, and manipulating pseudocode to show the procedures. The lessons are sequenced in a way building upon student learning goals, with each lesson using concepts from prior lessons. Additionally, each algorithm and implementation clearly is connected to the culminating project. This would be explained to students either before the implementation or after as a debrief. Students also have opportunities to explore their own ways of implementing various algorithms. While there are better ways of writing these implementations, students can explore and experiment with the ideas, writing their own code to reflect their thinking. Throughout the unit, students will journal and reflect on the process, revising their code and refining their work. Throughout this process, students are able to update their work and consider what changes they are making, along with why they are making the changes. With the nature of the project building upon their understanding, students naturally have opportunities to exhibit their understanding. The differentiation point is a little weaker due to not being fleshed out enough. It could've been clearer as to how students who fall behind can still achieve the desired results, along with how students who are ahead can add to their product, demonstrating their learning still. However, the organization and sequencing are logical and flow with the development of each concept. Materials are identified but not all manipulatives are in the plan. For example, the pseudocode symbols aren't in the plan, along with a link to a functional version of the project. However, the materials are listed.

24. To what extent is the entire unit coherent, with the elements of all three stages aligned?	[✓ **Extensively**]	**Somewhat**	**Minimally**
25. To what extent does the design present a wise balance between a focus on big ideas and the development of basic knowledge and skills?	**Extensively**	[✓ **Somewhat**]	**Minimally**
26. Does the time period of the unit allow the identified content standards/benchmarks to be adequately addressed?	[✓ **Extensively**]	**Somewhat**	**Minimally**
27. To what extent does the design present a wise balance between an in-depth approach and feasibility?	**Extensively**	[✓ **Somewhat**]	**Minimally**

28. To what extent does the design present a wise balance between developmental appropriateness and genuine challenge?	[✓ **Extensively**]	**Somewhat**	**Minimally**

Overall Unit Design
Notes/Reflections (provide evidence here):

I think the entire unit is coherent as each element is present and makes logical sense. The organization is logical and builds upon prior knowledge. The assessments also help develop the culminating project while also providing feedback for students and the teacher as to what areas require slowing down and what areas could speed up. In terms of the balance between big ideas and basic knowledge and skills, I think the balance could be better. I feel like the big ideas are focused on too much and the basic skills aren't enforced enough. This would lead to an issue with students developing a stronger or deeper conceptual understanding of the ideas. However, there is a decent amount of time spent on reinforcing some basic knowledge or reviewing concepts again, such as reading pseudocode, using recursion, and creating websites. These would've been covered before, but I felt like the plan doesn't include enough time on these topics as the expectations for students to already know these are there. I think the time period of the unit allows the standards to be addressed, as this unit is primarily focused on applying their understanding to create a functional product. This unit isn't focused on building the foundations but on applying their learning. In terms of balancing between an in-depth approach and feasibility, I feel like it's not the most feasible lesson plan due to a lot being approached each week. Going into this, I had originally considered each lesson to be an hour long. However, as I was working through the calendar, I felt that an hour wouldn't be enough time to cover everything. Additionally, some of the topics would only develop a more surface-level or more procedural understanding of the concepts. If I could extend this to 8 weeks and spend more time on each concept, I would then probably say this would be more feasible. However, due to the time spent on each topic, I don't think this is really feasible. It's possible but it wouldn't develop a strong enough conceptual understanding for students to confidently move forward. Addressing this would likely entail the next unit working similarly, with students developing a web service for others to use. Likely it would follow my original plan for this unit, where students would research and create a product that they personally are invested in. Lastly, I would say there's a wise balance between appropriateness and genuine challenge as the exercises have a decent amount of scaffolding, but students are still pushed the apply their thinking, thinking critically throughout the unit.