An Accessible Portal to Teach Computer Science Modules to Typical and Special Needs Children: A Prototype

Davis Ward
Computer Science and Software Engineering
Auburn University
Auburn, AL USA
e-mail: dzw0042@auburn.edu

Quinterious Hall
Computer Science and Software Engineering
Auburn University
Auburn, AL USA
e-mail: qdh0003@auburn.edu

Daniela Marghitu
Computer Science and Software Engineering
Auburn University
Auburn, AL USA
e-mail: marghda@auburn.edu

***Abstract*— Society's increased reliance on technology has simultaneously increased the demand for people who can develop and design these new advancements. This has led to an influx of students looking to learn how to code and gain the technological skill set that is currently among the most marketable. Learning to code is challenging; without the right tools, resources, and assistance, it can be tough to build the foundation needed to understand key computer science fundamentals. The existing web platforms focused on assisting K-12 learners are competitive from an educational and technical perspective. There is a huge lack of virtual educational platforms that can deliver resources to students with disabilities through innovative accessible features and provide guidance to K-12 teachers that are trying to support this area. This lack of guidance is especially evident when examining resources available to teachers about increasing access and engagement of struggling learners including students with disabilities. The motivation of this paper is to introduce a prototype of a centralized portal, Accessible Virtual Learning, that implements user experience strategies and accessible usability principles aiming to be accessible to any student and also educators who need guidance on finding suitable materials. The success of this portal relies heavily on its ability to allow teachers and self-directed learners to facilitate curriculums effectively while maximizing student engagement, ease of learning, and digital assistance for students at various ages with different learning abilities, both physical and cognitive.**

Keywords-teaching computer science; accessibility; students with disabilities; human computer interaction.

I. Introduction

Over the next decade, the U.S. will have to adapt to technological advances (AI, Big Data, and Cybersecurity) by creating structures and implementing coordination strategies that take full advantage of the opportunities they present. This situation will only become more urgent: by 2026, Science and Engineering (S&E) jobs are predicted to grow by 13% compared with 7% growth in the overall U.S. workforce [1]. Yet, even as Science Technology Engineering and Math (STEM) competencies have become more essential, U.S. K-12 mathematics and science scores are well below those of many other nations and have stagnated [2].

Women, underrepresented minorities, and people with disabilities remain inadequately represented in S&E relative to their proportions in the U.S. population. The rapid growth of S&E jobs and demographic changes have outpaced the progress that has been made in the participation of these groups in S&E. Increasing STEM skills and opportunities for all Americans requires local, state, and federal governments, public and private educational institutions, community organizations, and industry to step up their efforts. Earlier intervention is needed to advance STEM education and careers [3].

Society's increased reliance on technology has simultaneously increased the demand for people that can develop and design these new advancements. This has led to an influx of students looking to learn how to code and gain the technological skill set that is currently among the most marketable. Learning to code is challenging and without the right tools, resources, and assistance, it can be tough to build the foundation needed to understand key Computer Science (CS) fundamentals.

In the past decade, there has been increased awareness of the importance of teaching CS basics to students prior to college. Many high schools now offer the opportunity to at least experience the rudimentary principles in developing software. This has led to more students being prepared to tackle a college curriculum in STEM and being more successful in developing the skills necessary to pursue a career in computer CS. However, there seems to be a lack of resources that allow special needs students to achieve this knowledge as efficiently as typical students can [4]. Despite the commitment of the CS education field to increasing equity within CS education, there is still limited guidance for K-12 teachers on how to support a broad range of learners in CS education. This lack of guidance is especially evident when examining resources available to teachers about increasing access and engagement of struggling learners including students with disabilities.

The existing web platforms focused on assisting K-12 schoolers in their CS endeavors are competitive from an educational and technical perspective. The next step is to make these platforms effective for a wide range of learners (e.g., students with special needs). This requires compiling several accessibility features to cater to both special needs and typical students to ensure an equal opportunity to learn [5]. In this paper, we:

1. Evaluate the basic needs of both typical and special needs students
2. Discuss the design goals for making sure these needs are met, and
3. Explain our development process and how it improves CS learning modules for K-12 special needs students.

Learning needs of students differ, and the combination of accessibility needs for digital tools and the gap in digital literacy across socioeconomic and racial/ethnic lines create an inequitable environment in early CS education. These challenges are evident in the annual reports from the K-12 education community declaring the lack of diversity and equity in CS classrooms and a call for action [6]. The web portal will implement the Universal Design for Learning (UDL) framework toward providing a centralized location for learning by compiling several common coding platforms designed for high schoolers and beginner learners. The design of this portal will require the implementation of significant accessibility features such as voice navigation and other assistive elements to make these resources accessible for students with any form of disability or condition that could affect their learning.

The success of this portal relies heavily on its ability to allow teachers and self-directed learners to facilitate curriculums effectively while maximizing student engagement, ease of learning, and digital assistance for students at various ages with different learning abilities, both physical and cognitive. The development of this website has always focused on the user first, before design or development. The Accessible Virtual Learning (AVL) portal will be able to deliver collections of resources in a visually pleasing, accessible, and engaging way.

In Section II we will discuss related platforms and accessibility tools. In Section III we will discuss the study of Human-Computer Interaction. In Section IV we will discuss the design approach, architecture of the platform, and accessibility features. Lastly, In Section V we will discuss the conclusion of our work and task moving forward to improve the prototype.

II. BACKGROUND AND RELATED WORK

There are several disabilities that can be a challenge to effectively use computers and other technologies. Many accessibility features have been introduced since the commercialization of the personal computers to help people with disabilities use technology more easily.

Closed captioning is the display of text on a screen from the audio portion of a video. This allows a user to read any spoken dialogue, music, or even register sound effects and has been an instrumental accessibility feature to ensure material is available to individuals who are deaf or have impaired hearing. Closed captioning differs from subtitles as it provides greater accuracy and includes dialogue, an explanation of sound effects, and identification for who or what is currently speaking.

Keyboard shortcuts were introduced as an accessibility feature to allow users to access a site in its entirety using only typed commands. Many users with disabilities are not able to use a mouse or pointer to navigate the interface of a computer. Keyboard shortcuts have also become common among typical users, who usually only use keyboard shortcuts for certain tasks. An ongoing issue with keyboard shortcuts is that computer interfaces are normally designed to work best with the combination of both a keyboard and mouse being used. Navigating a site using the keyboard exclusively can easily become more cumbersome than using a mouse, seemingly creating an entirely separate user experience for users with disabilities. The task of creating a system that has seamless integration of navigation accessibility features will have to overcome the challenge of making sure navigating a site via keyboard exclusively has comparable utility as using a mouse and keyboard combination.

In the mid-twentieth century, barrier-free design and accessible design terms were introduced to illustrate efforts to remove physical barriers to people with disabilities [7]. Over the years with the technology advancements, there have been many improvements on how information is being presented to students with vision-impairment, Screen Readers being the dominant mechanism.

A Screen Reader is a software application that converts text and/or images from a screen to the speech format that visually impaired people can understand and interact with. Many screen readers are also compatible with the websites developed under accessibility standards. The main disadvantage of Screen Readers is that blind users need to go through an abundance of irrelevant content before they find what they were looking for [8]. This problem can be resolved by using an interactive JavaScript speech recognition library that gives the speech control of the application to the user. This allows an application to use the device's microphone and receive speech. The speech is then converted to text that is subsequently matched against a list of commands that would initiate a corresponding action for the user navigating the site.

Text-to-speech is a commonly used feature that ultimately allows text from your mobile device or your computer to be read to you aloud. Text-to-speech has drastically improved the access of information for the visually impaired specifically. An issue with text-to-speech, since its introduction as an accessibility feature. has been that the text is usually read by a computerized voice that can frequently mispronounce or distort the natural phonics of a word, making it difficult at times for someone who is using the feature to accurately interpret what is being said and affecting their overall literacy over time. Over the years, this issue has been addressed by developing more natural sounding text-to-speech systems that are almost indistinguishable from humans.

Moodle is a platform that has made it is to be fully accessible and be able to accommodate all users regardless of what their learning needs may be [18]. Their interface is tested with a range of screen reader software and is developed to comply with most accessibility standards. Totara's corporate e-learning platform provides the same accessible learning modules for business and organizations to perform training needs and employee onboarding [19]. E-learning platforms need to be available to a wide audience; and implementing as many accessible tools as possible only increases the potential audience that the platform could reach.

III. LITERATURE STUDY METHODOLOGY

The way technology has revolutionized the world socially, economically, and politically has been seismic and is clearly only scratching the surface. In a matter of a few years, the Internet has become one of the widely used technologies that has changed the way we communicate, learn, or do business. A 2019 report by Internet World Stats shows that the number of internet users has increased by almost 1150% since 2000 and 4.39 billion active internet users in 2019 [9].

Human-Computer Interaction (HCI) is a study of design, implementation, and evaluation of an interactive computing system for human use and for studying the major phenomena surrounding them. The accelerating growth of the Internet and the technology boom has led a number of schools and universities to provide courses and degree programs via distance education. HCI research has made it possible for students with disabilities to have the necessary accommodations for an equal opportunity to gain an education online. In most STEM fields, it is imperative for students to at least be moderately proficient at math. Students with disabilities are often at a disadvantage when it comes to understanding complex formulas and interpreting important visualizations. As HCI has evolved, students with vision-impairments have been able to close the gap with MyA+ Math, an accessible learning platform that has interactive resources to help the visually impaired learn key math concepts [17]. With the evolution of HCI, developers have also been able to explore new ways to make the interaction between humans and computer easier [10].

Software engineers have very quickly risen to the top of the totem pole in job outlook, and technology companies can only hope that the supply of skilled developers will one day match the demand. Developing software is a strenuous task. Learning to develop software is even more difficult and compounds the challenge of knowing what code to write on top of knowing how exactly to write it. Therefore, it is important to identify and alleviate any additional challenges that are not inherent in the process of learning CS. Making CS easier to learn is not the objective. The objective is to ensure we are not making it more difficult to learn than it already is.

When analyzing the challenges presented to students learning to code, it is clear the learning curve gets steeper for students with special needs. This is simply due to the fact that learning resources have not catered their curriculums or platforms for this specific demographic and lack even basic components necessary to ensure special needs students can learn just as efficiently as typical students. Coding is extremely visual and intellectual. If there are students with visual impairments or cognitive disabilities, it presents several obstacles that may make it difficult for these students to even begin their learning process.

Special needs students are frequently provided the opportunity for accommodations for face-to-face and traditional instruction methods. It is imperative to activate the same policies for online learning platforms. The W3C Web Content Accessibility Guidelines(WCAG) [11] provide a framework for ensuring basic accessibility needs are met; and all platforms should be complying with these to meet the needs of students and reach a larger demographic of learners with their resources.

IV. DESIGN AND IMPLEMENTATION

We designed the AVL portal with the following objectives:

- Provide a clean and easy to understand user interface for the user to create an account and get the wanted resources.
- Adhere to W3C Accessible standards that allows the use of Screen Readers and other accessibility tools.
- Provide a clean color scheme and font sizes that are accessible to visual disabilities.

A. Technology Used

Accessible Virtual Learning is implemented using Hypertext Markup Language 5 (HTML), Cascading Style Sheets (CSS), Embedded JavaScript Templates (EJS), and JavaScript for the front-end. For the back end, Node.js runtime engine [12] along with Express framework is used. A MySQL database is used for storing user's information such as name, email, encrypted password, user role such as 'educator' or 'student', and foreign keys for module ownership. There also exist tables that store module information for educator resource allocation. The blog portion of AVL uses ghost.io, an online publishing platform that makes content administration tasks secure and straightforward. It also uses a RSS feed to add articles to the AVL blog that are related to our content space, along with the articles that our content creators publish. The backend also uses the following open-source JavaScript packages and middle-wares [13]:

- Sequelize is a promise-based Node.js object-relational mapper that is used for the MySQL database models and querying.
- Bcrypt.js is a JavaScript package that allows proper password hashing for privacy and security of user profiles.
- Passport.js is a Node.js authentication middleware that facilitates the AVL login system.
- Annyang.js is a JavaScript speech recognition library used for voice navigation on web apps. Custom voice commands and actions can be created that allows AVL to be more accessible to users with

visual disabilities. It is especially useful for user navigation purposes.

- Connect-flash is a JavaScript package that is used in AVL to create robust user feedback related to form interaction. This facilitates all the back-end form validation messages to the user.
- Express-validator is a JavaScript package that assists in the back-end form validation and sanitation logic.

B. AVL Portal Architecture

The AVL portal follows a model-view-controller architecture [14]. There are models that are representations of data that are being posted and manipulated by the controllers. The program logic and database manipulation are done in the controllers that pass on data to the view, in the form of EJS templates that serve HTML/CSS/JS pages to the user. The architecture is facilitated through Express routes that are used for knowing what the user wants to see or interact with and calls the appropriate controllers to interact with the data, and then sends the appropriate view with that data. In the following, we detail the implementation of different components implemented in AVL

1) Dashboard Component

The dashboard delivers content and functionality to both educator and student accounts and is the first thing the user sees after logging into AVL. The dashboard features a list of resources, called modules, that educators can add – such as an article, resource, or course along with an URL to the resource. Educators can create, edit, and delete modules. Modules are stored in the MySQL database within their own table, with a foreign key connecting it to the educator who created it. Students can then view and sort through the modules that all educators have created. Modules are contained by a card user-interface that is in a list that can be navigated by keyboard, which is especially important for screen readers. Users can also consume modules by the author, through the educator page. This is important if a student is using the website to get resources specifically from their educator.

2) Blog

The blog allows educators to create blog posts that surround the topics of accessibility and CS education. These blog posts are meant to be read by both educators and students. Educators can create, edit, and delete blog posts. The posts are then displayed in an accessible way. The blog also generates content from news feeds on relevant technological topics in order to maintain a fresh collection of articles to read whenever a user logs in. Students can view the entire archive of blog posts and articles but are not allowed to post, edit, or delete any content. Blog content is strictly informative and should act as an extension of the learning modules within AVL to facilitate extracurricular learning not directly related to coursework. The blog was implemented using ghost.io [14] to facilitate the type of content but also the content authors. Since this is a public facing portal and ultimately anyone can create an educator account, we decided to have the ability to choose which educators can create content for the AVL blog.

3) Resources Component

The resources page is a collection of resources that the site creators collected. These are resources that are notable in usability and popularity in the education and computer science space. The resources are also displayed with a card user interface.

4) Voice Navigation

The voice navigation feature allows users to explore the different features of the portal non visually. When a user logs in, a large voice icon on the bottom right of the screen is presented, which will also be accessible to a screen reader. This button displays a banner over the whole web app that explains how to use the voice navigation. The only purpose of this button is to display those instructions. When a user is on any given page, they can speak any of the following commands to be redirected to the desired page. This component was implemented with Annyang.js, where all of the voice commands and their desired actions were added. With this library, more custom voice commands can be added in the future to extend the scope of the portal. The voice commands can be seen below in Table 1.

TABLE I. VOICE COMMANDS

Voice Commands	*Action*
‘Home’	Redirect to the index page
‘Dashboard’	Redirect to the dashboard page
‘Educators’	Redirect to the educators’ page
‘Blog’	Redirect to the AVL blog
‘Resources’	Redirect to the resources page
‘Log out’	Log the user out of the portal

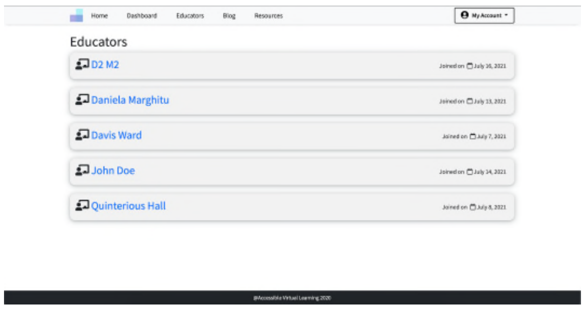
C. User Interface

This section will contain screenshots of the pages, features, and functionality of the portal that has been developed. As this is still a prototype, the look and scope of the portal may change in the future. They may be changed based on feedback from students and educators.

The user interface uses a CSS framework, Bootstrap 5, to aid in the development of the views. This framework is especially helpful when creating mobile first applications. The use of Bootstrap 5 components does speed up the development process of user interfaces, but developers must be careful to add extra html attributes and hidden text, as not all bootstraps are natively accessible to W3C standards.

1) User’s View

Figure 1, Figure 2, and Figure 3. are screenshots of the educators’ page, resources page, and AVL blog. These are pages that all users can access, and do not change based on the user type.

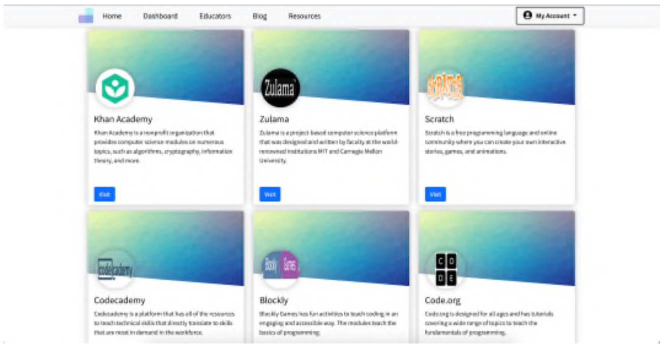


Educators

D2 M2	Joined on July 16, 2021
Daniela Marghitu	Joined on July 13, 2021
Davis Ward	Joined on July 7, 2021
John Doe	Joined on July 14, 2021
Quinterious Hall	Joined on July 8, 2021

@Accessible Virtual Learning 2020

Figure 1. Educators' page.



Khan Academy

Khan Academy is a nonprofit organization that provides computer science modules on numerous topics, such as algorithms, cryptography, information theory, and more.

Zulama

Zulama is a project-based computer science platform that was designed and written by faculty at the world-renowned institutions MIT and Carnegie Mellon University.

Scratch

Scratch is a free programming language and online community where you can create your own interactive stories, games, and animations.

Codecademy

Codecademy is a platform that has all of the resources to teach technical skills that directly translate to skills that are most in demand in the workforce.

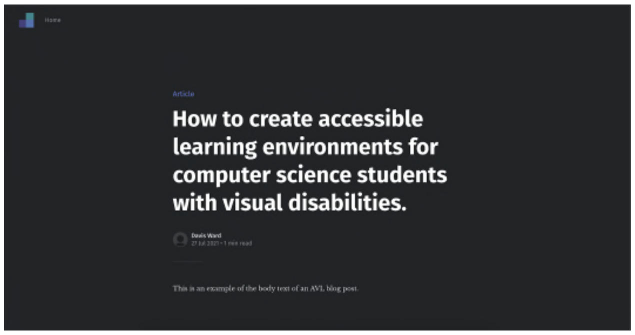
Blockly

Blockly Games has fun activities to teach coding in an engaging and accessible way. The modules teach the basics of programming.

Code.org

Code.org is designed for all ages and has tutorials covering a wide range of topics to teach the fundamentals of programming.

Figure 2. Resources page.



Article

How to create accessible learning environments for computer science students with visual disabilities.

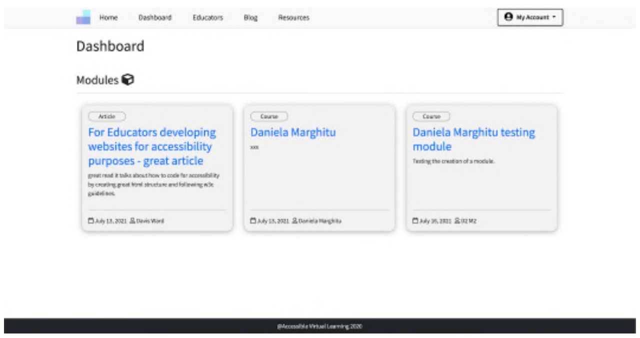
Davis Ward
27 Jul 2021 · 1 min read

This is an example of the body text of an AVL blog post.

Figure 3. AVL Blog page.

2) Student's View

Students can interact and explore resource modules but cannot create modules. The cards are keyboard focusable and navigable. The view also dynamically sizes the card based on the width of the viewing screen, the number of modules, and the amount of content within each module. Figure 4. shows the card user interface for the student's dashboard page.



Dashboard

Modules

Article
For Educators developing websites for accessibility purposes - great article
great read it talks about how to code for accessibility by creating great html structure and following w3c guidelines.
July 13, 2021 Davis Ward

Course
Daniela Marghitu
xxx
July 13, 2021 Daniela Marghitu

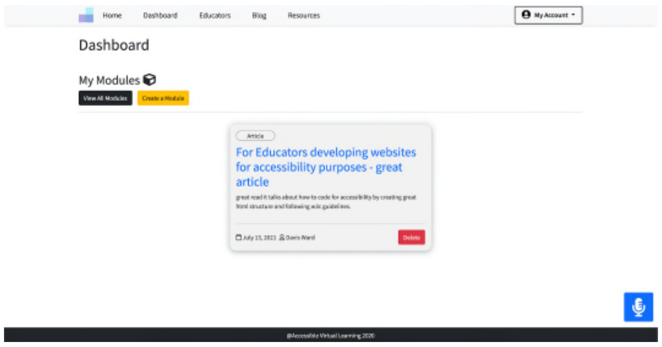
Course
Daniela Marghitu testing module
Testing the creation of a module.
July 16, 2021 D2 M2

@Accessible Virtual Learning 2020

Figure 4. Student's view.

3) Educator's View

Educators can consume the resource modules, but also view them as the student would. Figure 5 shows the card interface showing the educators own modules, and Figure 6 the form for creating a module. All of the educator's views are accessible in the same way that the student's views are. This is important because it demonstrates the opportunity for students with disabilities to become educators with disabilities.



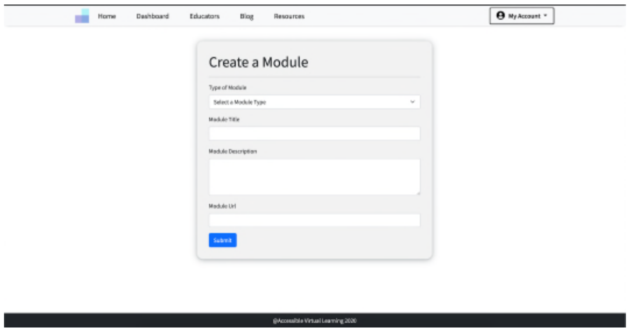
Dashboard

My Modules

Article
For Educators developing websites for accessibility purposes - great article
great read it talks about how to code for accessibility by creating great html structure and following w3c guidelines.
July 15, 2021 Davis Ward

@Accessible Virtual Learning 2020

Figure 5. Educators' view.



Create a Module

Type of Module
Select a Module Type
Module Title
Module Description
Module Url

@Accessible Virtual Learning 2020

Figure 6. Educators' form for creating a module.

D. Testing and Accessibility

1) Design for Accessibility

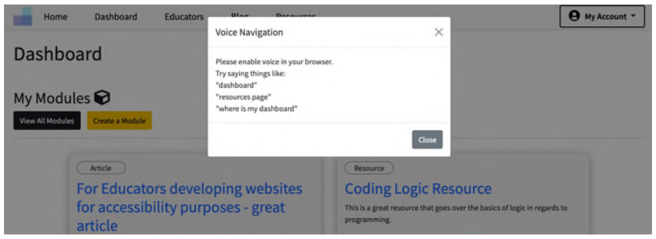
According to the Web Content Accessibility Standard 2.0 (WCAG 2.0), the following are important requirements for making web apps accessible:

- Text alternatives that serve equivalent purpose for all non-text content
- Text can be resized up to 200% without losing content functionality
- Users can operate the site using keyboard-based navigation options
- Users can access content with the use of assistive tools like screen readers
- Text to background contrast must be a 4.5:1 ratio at a minimum

An accessible portal that adheres to W3C standards must be designed with strong, semantic, and structural HTML that closely follows the guidelines. When designing the user interfaces, the HTML is the first thing that was focused on, as styles can be added after to create a better-looking view. Many Accessible Rich Internet Application (ARIA) [15] attributes were introduced natively to HTML 5. For example, many of the buttons on the site are instead used as anchor tags, but with a role attribute of the button. This functionality means to screen readers that it is not a link, but a button that a user clicks.

The use of icons is also used heavily on the portal, both semantically and decoratively. For decorative icons, the aria attribute of aria-hidden should be set to true, so that a screen reader will simply skip over the icon tag. Since it does not display any meaning, it is not necessary for visually impaired users to digest. However, for semantic icons, is it extremely important to use accessible html because these icons display important meanings for the content that is next to, or below them. A span tag must be added after the icon that contains the textual meaning of the icon and is hidden to visual users, but not to screen readers. This way a visually impaired user can have the same experience as a visual user. Content images can also enhance visual user experience, but they must have appropriate alternative text for visually impaired users to receive the same experience.

Adding native voice navigation to a web portal is a huge advantage. Screen readers are advanced enough to make navigating a website using auditory and physical sense inputs and responses easy but having the ability to navigate pages instantly through speech makes it even more accessible for these users. With a feature like this, it is paramount that the instructions to use the voice navigation are easily consumed by the screen reader, or the feature itself would be unusable without third party assistance. When a student clicks on the voice navigation symbol, instructions pop up that also dim the rest of the page. The voice navigation feature can be shown below in Figure 7.



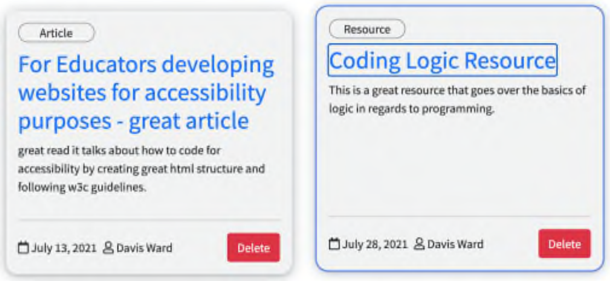
Voice Navigation

Please enable voice in your browser.
Try saying things like:
"dashboard"
"resources page"
"where is my dashboard"

This is a great resource that goes over the basics of logic in regards to programming.

Figure 7. Voice navigation feature.

Additionally, making sure the web portal was navigable by keyboard was an important standard. It should not only be navigable by keyboard, but when a user is focused on a certain user interface element, the element should show a visual cue to let the user know where they are on the structure. For example, when a user is on the AVL dashboard, and navigates to focus on one of the module cards through keyboard action, the card is moved in an upright position and a colored border appears. The user can then click enter to navigate to the URL of the module they are focused on. The unfocused and focused states can be seen in Figure 8. The hover effect is also the same as the focused effect for users navigating by mouse.

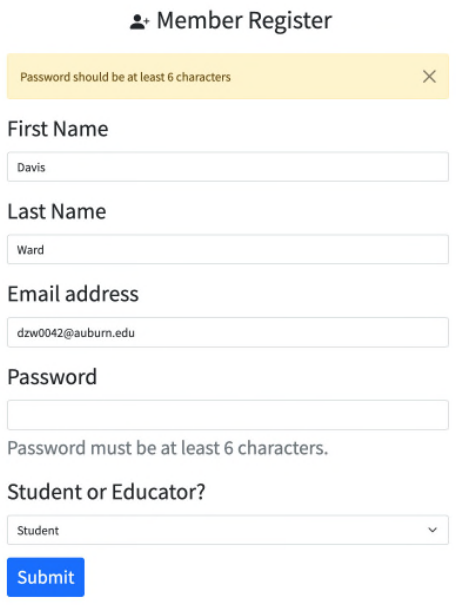


great read it talks about how to code for accessibility by creating great html structure and following w3c guidelines.

This is a great resource that goes over the basics of logic in regards to programming.

Figure 8. Unfocused and focused states of modules.

Forms are also a very important part of accessible web apps. They must have clear labels that correspond to each input area so that the user knows what each input is for, and also so that screen readers can correctly convey the form. AVL forms use server-side validation and user notifications to display success, warning, and error messages regarding the submission of the form. The notification should clearly state which label-input field was not sufficient to let the user know exactly how to fix it to successfully submit the form. Figure 9 shows the user interface of the member registration form, and what happens when the password field input is not sufficient. It states what label was incorrect, and what was incorrect with it.


Member Register

Password should be at least 6 characters

First Name

Davis

Last Name

Ward

Email address

dzw0042@auburn.edu

Password

Password must be at least 6 characters.

Student or Educator?

Student

Figure 9. Validation of a member registration form.

2) *Testing for Accessibility*

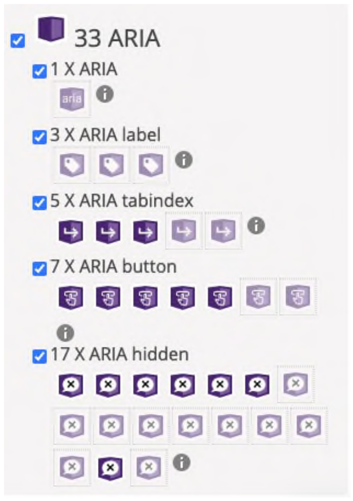
Testing for accessibility is just as important as designing for it. The testing process for the portal includes the following main steps:

1. Code review of the HTML structure and correct attributes, alt tags, and hidden ARIA text.
2. Testing of the live site by interacting with all features and assuring expected behavior.
3. Testing with the Web Accessibility Evaluation Tool (WAVE) [16], a chrome extension that displays any errors against the W3C guidelines by parsing the HTML structure of each page.
4. Comparing the web portal against the WCAG 2.0 Web Accessibility Checklist to assure all requirements were met.

Going through this type of testing is much more robust then only doing the first step. It assures that the site is indeed accessible, as it is very easy to skip over critical accessibility problems.

Out of the 47 guidelines that are specified in the WCAG 2.0 Web Accessibility checklist, all were met except 4. These are planned to be fixed in the near future before testing with actual student subjects. Specifically, the readability of the site can be improved, with alternate text for information that is past a lower-secondary reading level. These areas will be highlighted with the actual test subjects and alternate text will be provided. Additionally, text-based help needs to be added for the module functionality. We do think it is intuitive enough to be used without discrete instruction, but this may prove untrue in subject testing.

The test results using WAVE were very promising as we used it concurrently throughout the development phase. Each page was updated to ensure a result of 0 errors in WAVE. It also shows all of the aria attributes and ensures color contrast is acceptable. Some results from WAVE can be seen below in Figure 10.


33 ARIA

1 X ARIA

3 X ARIA label

5 X ARIA tabindex

7 X ARIA button

17 X ARIA hidden

Figure 10. WAVE ARIA attribute results.

V. CONCLUSION AND FUTURE WORK

The AVL platform meets the original objectives and design goals. As a prototype, it needs to be tested and expanded on. In any online platform, accessibility should be a requirement, not a design goal. If platforms continue to strive to implement key features that allow inclusion for all, the online education space will make great progress. We hope we made a difference by examining accessibility on the web and turning our findings into a working prototype.

In the future, we plan on testing the portal with a range of different students to ensure that the user interface is easy to follow and understand. We would also like to see different students using the site with screen readers and testing our voice navigation. The input from this type of testing would be insightful and lead us to making smart changes. Testing would be done by selecting a range of students with different disabilities ranging from visual to developmental. We would not be allowed to assist the student at all with creating an account and interacting with learning modules. If a student ever became stuck or confused, then we would note the point of frustration and acknowledge a change needed for that functionality or content. This would ensure that not only is our site accessible, but usable by a large range of students and educators with different needs.

Technology continues changing the way we learn. If the goal is to maximize the potential that technology has as a resource for knowledge, it is imperative to ensure that this resource is available to everyone and can be used by any demographic to gain new skills, talents, and abilities.

A different approach to the need of more accessible resources for learning platforms relating to students with disabilities could have been a tool, rather than a portal. A tool that could be used for multiple purposes might have reached more learners, but after the initial research of the idea, creating a portal seemed to be a more direct approach. One of the biggest challenges for this project was deciding what specific needs the leaners would require, and how to implement them into our portal through functionality. We believe we did a good job of this in respect to the time and knowledge of our work, but after subject testing, the portal could be vastly improved with more specific accessible functionality that scopes past WCAG requirements.

This virtual learning platform is created to take the first step into providing a universal learning experience. As technology continues to evolve, the resources available to enhance learning will also advance. In the future, the following additions could be included to improve the site:

- Providing new and updated learning material is key for maintaining interest in a skill area. We are planning to add an RSS feed to the blog portion of the site in order to keep a constant flow of new content to keep learners attracted. Articles will be relevant to learning modules on the platform.
- The platform will give students the opportunity to be content creators on the site's blog after they have displayed a certain level of proficiency in their learning. Their content will be moderated by their respective educators.
- The platform will allow students to create a profile based on their interests and learning objectives. This would then be used to recommend public other relevant resources.
- The current version of the platform uses voice navigation and has an established list of commands a user is allowed to use. A future version of the site will have custom voice commands added to expand the utility of this feature and improve the overall user experience.
- The site will be updated once extensive user subject testing trials are completed, to ensure usability and to make sure the site meets all the needs of the target audience.

ACKNOWLEDGMENT

This work was made possible due to NSF Grant # 1842092.

REFERENCES

[1] National Science Board, National Science Foundation. 2020. Science and Engineering Indicators 2020: The State of U.S. Science and Engineering. NSB-2020-1. Alexandria, VA. [Online]. Available from: https://ncses.nsf.gov/pubs/nsb20201/ 20021.07.22

[2] NSB, "Elementary and Secondary Mathematics and Science Education," Science & Engineering Indicators 2020.

[3] S. Grover, S. Cooper, and R. Pea, "Assessing Computational Learning in K-12." ITiCSE '14 (p. 5). Uppsala, Sweden, June 2014.

[4] R. Ladner, M. Israel, "For All in Computer Science for All." Communications of the ACM 59, no. 9 pp. 26-28, 2016.

[5] M. Ray, M. Israel, C. Lee, and V. Do, "A Cross Case Analysis of Instructional Strategies to Support Participation of K-8 Students with Disabilities in CS for All." In Proceedings of the Association for Computing Machinery (ACM) Technical Symposium on Computer Science Education. (SIGCSE), pp. 900-905, 2018.

[6] J. Wang, H. Hong, J. Ravitz, and S. Hejazi Moghadam, "Landscape of K-12 Computer Science Education in The US: Perceptions, Access, and Barriers." Paper presented at the Proceedings of the 47th ACM Technical Symposium on Computing Science Education, pp. 645-650, 2016.

[7] E. Ostro,: Universal design: an evolving paradigm. Universal design handbook 2,34{42 2011

[8] I. Ramakrishnan, V. Ashok, and S. M.Billah: Non-visual web browsing: Beyond web accessibility. In: International Conference on Universal Access in Human-ComputerInteraction. pp. 322{334. Springer 2017.

[9] Miniwatts Marketing Group. Internet Usage Statistics: The Internet Big Picture World Internet Users and Population Stats 2019. [Online]. Available from: https://www.internetworldstats.com/stats.htm. 2021.07.22

[10] Lawrence, D.O., Ashleigh, M.: Impact of human-computer interaction (hci) onusers in higher educational system: Southampton university as a case study. International Journal of Management Technology 6(3), 1{12 2019.

[11] Web Content Accessibility Guidelines (WCAG) [Online]. Available from: https://www.w3.org/WAI/standards-guidelines/wcag 20021.07.22

[12] EDUCBA [Online]. Available from: https://www.educba.com/javascript-vs-node-js/ 20021.07.22.

[13] JavaScripting [Online]. Available from: https://www.javascripting.com/, 20021.07.22.

[14] Codecacademy [Online]. Available from: https://www.codecademy.com/articles/mvc 2021/07/22.

[15] Accessible Rich Internet Application (ARIA) [Online]. Available from: https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA , 20021.07.22.

[16] WAVE Web Accessibility Evaluation Tool [Online]. Available from:

https://wave.webaim.org/, 20021.07.22.

[17] Jariwala, A., Marghitu, D., Chapman, R.: A multimodal platform to teach mathematics to students with vision-impairment. In: Antona, M., Stephanidis, C. (eds.) Universal Access in Human-Computer Interaction. Access to Media, Learning and Assistive Environments. pp. 109–117. Springer International Publishing, Cham (2021).

[18] Moodle [Online]. Available from: https://docs.moodle.org/311/en/Accessibility 2021.07.22

[19] Totara [Online]. Available from: https://help.totaralearning.com/display/TPD/Accessibility+at+Totara 2021.07.22