The Cybersecurity Packet Control Simulator: CSPCS

Neil Loftus*, Cameron Green†, and Husnu S. Narman‡
Department of Computer Sciences and Electrical Engineering, Marshall University
Huntington, WV, United States of America
Email: *loftus6@marshall.edu †green320@marshall.edu ‡narman@marshall.edu

***Abstract*—Games have been used as a learning tool for centuries. Through game-based learning, students actively learn and practice the right way to do things. A project team at Marshall University created a web-based application to visually represent networking and cybersecurity concepts by using "gamification" techniques such as linear levels and visualization. The goal of this project is to assist the instruction of Computer Science students in internetworking and cybersecurity subjects through game-based learning and measure its effectiveness. The application has users control a simulated network environment and direct computer messages to explain internetworking and cybersecurity concepts. An experiment was conducted with a test group taking a pre-test and post-test after using the applications. These two sets of data were compared and analyzed to measure the effectiveness of the application. Testing results shown that the application improved the self-evaluated confidence of students by approximately 45%, regardless of the students' self-reported learning style.**

***Index Terms*—Cybersecurity, Internetworking, Simulation, Education, Computer Science**

I. Introduction

In the modern age, the field of cybersecurity is one of the most critical concepts for the sake of national security. However, due to its complexity, this field is often challenging for students to get into and understand [1], [2]. A major factor in cybersecurity being difficult to approach is inequity in computer science education. Research has shown that to broaden participation of underrepresented groups in computer science careers, there must be equity in computer science education [3]. In this circumstance equity is expanding access to computer science education to all students, regardless of race, gender, disability status, or socioeconomic status. One source of inequity the paper points out is specifically related to how comfortable any given teacher is in teaching students about computer science. Many teachers do not have much experience in computer science education, and this can lead to students from those schools having less understanding and exposure to computer science concepts, than a student from a different school [3].

One remedy to this issue is usage of "Open Education Resources" (OER). An OER tool can be used by educators and students without need for purchase or permission [4]. The OER movement can help combat education inequity, by providing students with educational material that they might have not been able to receive otherwise. Students learning about computer science in general, learning alone cybersecurity, often struggle to grasp the more abstract and have difficulties in visualizing ideas of the subject particularly, as traditional lecture formats can be ineffective for some students as they cannot connect the ideas without seeing the process in action. As such, a visual demonstration can often provide students with the piece they are missing to understand the topic. While visual ideas can be demonstrated in drawing or with videos, for computer science, hands-on simulations have several benefits over other visualization forms. Most importantly, they allow students to learn at their own pace and experiment with different elements of the topic. These simulations can also make ideal OER resources, as when done correctly, the only requirement for use is a device that can access the application.

Previous research has been done into the effectiveness of visual programs for Computer Science learning. In a project by Narman et al., an interactive augmented reality application was developed to teach students about data structures. Of the students tested, 54% preferred the AR program over the alternatives offered [5]. Research into gamification, which is the technique of taking elements for games and using them in another context, has shown it as both effective for teaching by Buckly et al. [6] and preferred by students, as showed by Kingsley et al. [7]. In an educational context, an example of gamification would be turning whatever concept you wish to teach into a game and having students use it.

There have been several other programs created to explain cybersecurity concepts to students. These range from general-purpose education websites, such as Khan Academy [8], to video games focused based around cybersecurity, such as the ones developed by Texas A & M and CISA [9], [10], and to more utilitarian simulations, such as CS4G Netsim and Visual Algo [11], [12]. CSPCS is most similar to the third category of programs, using elements from games but being a simulation to introduce the fundamentals [13].

The *objective* of this paper is to create a tool to teach internetworking and cybersecurity with hands-on activities and then measure the effectiveness of "gamification" for

internetworking and cybersecurity education. This was done by creating a visual application to teach internetworking and cybersecurity, called "CSPCS" or "the Cybersecurity Packet Control Simulator." The application was then made publicly available on the internet, with the link to the application itself provided in references [13]. Afterward, a group of students were given a pre-survey, asked to use the application, and then given a post-survey upon finishing the application. These surveys asked users to rate the understanding of the topics presented from 1 to 5, with 1 being the least and 5 being the most understanding. Additionally, students were given an assessment quiz to measure how well they retained information correlated with their preferred learning style.

The *results* indicate students, on average, had a percent change of 45%, or a linear percent increase of 20% (from 2.21/5 to 3.21/5) in their rating of understanding after using the program on their first try. Students, on average, achieved a score of approximately 90% on the assessment score, with no strong correlation between learning style and better performance. Finally, by evaluating qualitative feedback from students, the main changes that could be made to the application to improve its effectiveness are: to add more detailed explanations, and more straightforward instructions and to update the user interface.

With these results it is important to acknowledge that this application was designed with a certain amount of assumptions about the audience that would be using it. Such assumptions include that the user would be a high school graduate attending college, that they would be fluent in the English Language, that they would not be visually impaired, and they would have access to a computer and internet connection. While this tool can be beneficial to people who are in this audience, it is limited in scope and may not be effective for users outside this demographic. In future iterations of this project, we have the opportunity to expand the audience in a number of different ways, such as providing accessibility or language options. Another option to expand the audience of users would be to create a mobile device supported version of the application, to allow users who have a mobile device but not a traditional PC to use to program.

The rest of the paper is organized as follows: Section II. describes the development process of the application. Section III. explains the implementation of the application, as well as demonstrates the system of input and functionality. Section IV. compares the application with some of the previous works that teach the same subjects, and section V. explains the results in more detail. Finally, section VII. presents concluding remarks with future works.

II. Development

A. Tools Utilized

The primary application we used for development was Unity 3D [14]. Unity was primarily used due to its ability to easily export to most platforms. Since the primary target is students, web browsers were determined to be the most accessible platform. In this case, CSPCS was exported to HTML 5 using the Unity engine's WebGL support [15]. This tool bundled the program as a website without having to refactor the source code to support browsers. Unity does not have inbuilt support for hosting, so Heroku was chosen to host the website online [16].

B. Content Overview

The application was broken down into six distinct levels, each related to internetworking or cybersecurity. Each level displays a distinct concept, with many building upon ideas established in the previous levels. Ideas are progressively introduced to avoid overwhelming and confusing the user. The first level is the most simple, introducing the idea of the simulated network and the system of control. The second level provides the user with a much more extensive example network but keeps the concepts relatively simple. This level focuses on general graph theory, such as how to find the best path between two vertices [17]. The third level introduces more graph theory concepts, such as weighted graphs, which are graphs with numerical costs associated with them [18]. This level primarily serves to demonstrate how weighted graphs can affect routing and change what the most efficient path is. The fourth level focuses more on the network algorithms, demonstrating how the routing algorithm used in early versions of the internet predecessor Arpanet functioned [19]. The fifth level is about pinging, which is where the time it takes for two devices to communicate is determined by sending a message (or a ping) between the two devices and back [20]. Unlike the first five levels, the sixth level is entirely cybersecurity-based. This level demonstrates a man in the middle attack, where a message between two devices is intercepted by a malicious third party, viewed and/or modified, and sent to the original destination without detection [21].

C. Design Challenges

Since the program was designed to teach cybersecurity, internetworking, and routing, establishing a system for routing between devices was key. Before routing could be established, we first needed a way to represent a computer network. The simplest and most effective way we discovered this was to represent a network as a graph and apply graph theory concepts to it. A graph is a series of vertices connected together by edges [17]. To represent this in code, we used a linked list or node-based approach, where each "node" has a list of other "nodes" that had connections. Before the algorithm could be selected, some important factors about the networks had to be considered when translated to a graph. The first is that the networks were not binary; nodes could have any number of connections, from 0 to the total

number of other nodes [22]. The second is that the graph was undirected, meaning that each connection was two-way. Due to the approach taken, the connections were inherently directed, as a node could be connected to a different node, but that node did not have to be connected back. To fix this, the nodes were programmed so that if one ever connected to a node, that node would always connect back. Finally, the graph would have to support weighted connections, where nodes had numerical costs associated with them [18]. Because of this, the algorithm had to be one that would always take the path with the smallest total numerical value associated with it. While not every graph features weighted connections, the advantage of using an algorithm based on numerical costs is that it can also find the shortest path of an unweighted graph. For any unweighted graph, each connection is treated as "1", and since the algorithm will always take the path with the lowest numerical value, this will take the path with the least number of steps.

D. Routing Algorithm

The chosen algorithm for CSPCS was the original algorithm developed for Arpanet in 1969. Because of its simplicity and it is one of the oldest algorithms used for networking, it was an ideal choice for explaining the baseline concepts of internet routing. The basis of this algorithm is the routing table, where each node has a list of every other node it has discovered, the total cost it takes to reach that, and the next node it must travel to eventually reach its destination [19]. The algorithm begins by adding all adjacent nodes into its routing table. The algorithm then performs a message function, where it compares its routing table with the routing table of each adjacent node. If there is a destination that has not been discovered yet, it is added to the routing table. If that destination has been discovered, it compares the metric costs and uses the smaller of the two. This is repeated until every node is discovered. Once complete, the shortest path will be found for both weighted and unweighted graphs.

III. Implementation

A. Main Menu

Fig. 1 shows the main menu of CSPCS. The main menu only features the necessary buttons, with the primary function, the level select, in the center. Upon clicking on the level select, the user is given a dropdown menu where they can scroll through and choose the level they wish to go to [13].

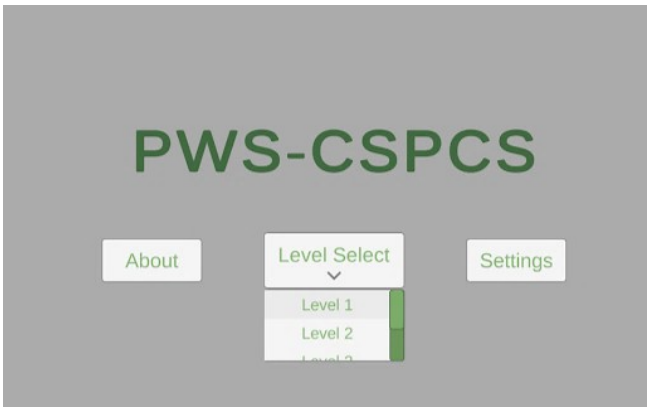

Fig. 1. Main Menu of CSPCS.

B. Simulated Environment

Fig. 2 shows an example of a level in CSPCS. On the left is the input panel, with several sections: control, info, objectives, questions, and help panels. On the right is the simulated network environment itself. The user primarily performs input on the left side of the screen, which affects the network on the right [13].

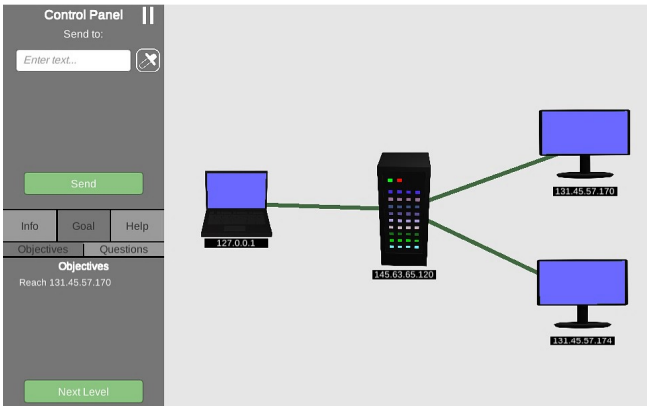

Fig. 2. Level 1 of CSPCS with a sample illustration.

C. Control Panel

While normally a computer network is controlled by a command-line interface, the program uses a simplified button interface so students can focus on the core concepts, as shown in Fig. 3. Fig. 3 illustrates element 1 (labeled in red), where the user can input the IP address of the device they are trying to reach. Similarly, element 2 is the input to specify what device is sending the message. For earlier examples, element

2 is excluded to avoid overwhelming the user with options. For later levels, more input fields are provided, such as to control how many packets the device sends. Element 3 is an eyedropper tool that allows the user to select an IP address by clicking on the device rather than text input. Element 4 is a button that allows the user to freeze the simulation so the user can inspect the packets in detail. Finally, element 5 is the send button. This begins the simulation with the data inputted [13].

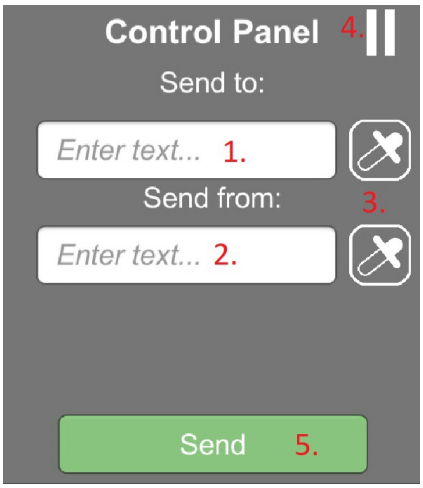

Fig. 3. Control Panel for CSPCS.

D. Selection Panel

Fig. 4 shows the selection panel. The bottom half of the input panel has options on what it can display based on the selection. The selection panel includes Info, Goal, and Help options to be selected. Each selected option has a sub options. As shown in Fig. 4, Goal selection has Objectives and Questions. Clicking any of the buttons will change the function of the bottom panel [13].

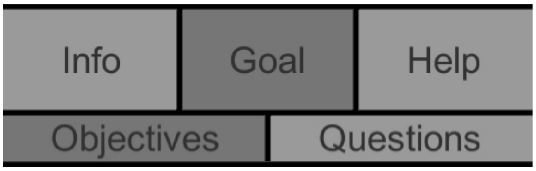

Fig. 4. Selection Panel for CSPCS.

E. Info Panel

The info panel has two primary functions, as shown in Fig. 5. The first is showing what a packet's information, such as its destination, sender, and message. Additionally, the user can select a device by clicking on it. Afterward, the dropdown menu will display the list of destinations available for that device in its routing table. When one of these IP addresses is clicked, the path from the selected device to its destination will be highlighted in the main application [13].

Fig. 5. Information Panel for CSPCS.

F. Goals: Objective Panel

The objective panel (as shown in Fig. 6) shows the goals that the user must meet to complete the level. This panel also features the button to move to the next level. The user can move to the next level without completing the level if they get stuck. During development, the ability to move to the next level was locked away until all objectives and questions had been correctly completed. This was changed to allow users to move at any time [13].

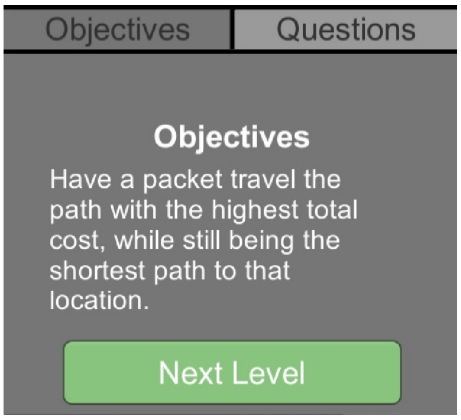

Fig. 6. Objective Panel for CSPCS.

G. Goals: Question Panel

The question panel provides the user with a series of questions about the content presented in CSPCS, as shown in Fig. 7. Similar to the control panel, this panel uses a text input system with an eyedropper button for convenience. Users have infinite attempts to complete a question, and the program will give either a red X or check-mark to denote if the answer was incorrect or correct. When a question is completed, a new one will be given until they have all been completed [13].

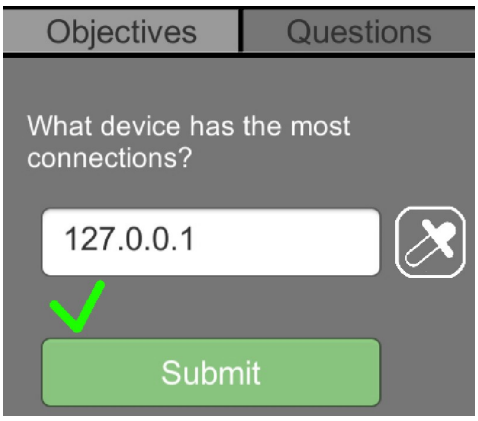

Fig. 7. Question Panel for CSPCS.

H. Help Panel

The final user interface element is the help panel (Fig. 8). At the start of the level, users are shown a brief pop-up panel with information about the level. Users can click on the help button to reopen this panel. The reset button resets all progress on the current level and returns it to its original state. This can be necessary to complete later levels. Finally, the main menu button can be selected to return users to the title screen, allowing them to select a different level [13].

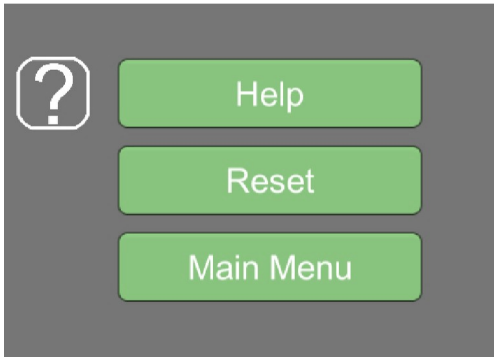

Fig. 8. Help Panel for CSPCS.

IV. Related Works and Distinction of the Developed Tool

A. Khan Academy

Khan Academy's lessons are divided into many smaller subjects. Each subject has a few articles and/or videos that teach people the information, as well as quizzes that test users' knowledge. Users can take each subject in any order they like, and certain subjects can even be skipped if users know the material. Khan Academy separates the visual aspects of its lessons from its interactive aspects, while CSPCS integrates them into each other. Our program has users answer questions while a visual representation of the topic is seen in the background. The program will also use visuals to show the user exactly how their answer is right or wrong. Another difference between the two programs is the topics that are being taught. Khan Academy teaches elementary computer science topics, while CSPCS focuses explicitly on cybersecurity and networking. CSPCS also goes more in-depth than Khan Academy on the material being taught to the users [8].

B. Texas A & M and CISA

Both Texas A & M and CISA have created a collection of different games on cybersecurity topics. Regarding Texas A & M, their games are primarily designed for a younger audience, with the content being more focused on identifying possible cybersecurity threats, such as phishing links in emails [9]. While Texas A & M developed their games for browsers, CISA made all of their games on the app store and google play store. Like Texas A & M, these programs are entertaining games first and aim to educate students on cybersecurity [10]. These games share some topics with CSPCS but are overall very different.

C. Visual Algo

Visual Algo is a website that provides users with several different simulations on computer science topics, ranging from sorting arrays to path-finding algorithms. These in-depth simulations provide users with accurate representations of the concepts presented. Unlike the other programs mentioned, Visual Algo primarily focuses on data structure concepts, not internetworking or cybersecurity [12]. This is still relevant to CSPCS since it aims to teach students about data structures concepts, especially graph theory, as a preliminary to the internetworking content. Visual Algo features more in-depth explanations than CSPCS, but it does not use any gamification, with it being a completely utilitarian simulation.

D. CS4G NetSim

On the surface, NetSim seems very similar to CSPCS, but the most noticeable difference is how information is displayed. CSPCS displays all the relevant information on the screen at once, while NetSim only displays a little at a time.

For example, both programs display a computer's IP address differently. With CSPCS, the IP is right under the computer, but NetSim requires the user to click on the computer to display its IP and other relevant information. NetSim will also obstruct other information like the objectives and the info of other devices when the computer's info is displayed. There is also a major difference in the complexity of users' inputs. CSPCS has a simple set of choices that are predetermined based on what lesson is being taught to the user. NetSim has the user open up a mini-menu with several different blanks the user needs to type their answers into, as well as a dropdown menu for all the IP addresses. NetSim's menu stays mostly the same for all its lessons, even if some of the blanks are not needed in certain cases. The two programs also have a difference in what they teach. NetSim only focuses on cybersecurity, while CSPCS teaches other subjects on top of cybersecurity [11].

Fig. 9. Graph of students' self-assessed understanding of cybersecurity, internetworking, and routing before and after. The left side shows before, and the right side shows after.

V. Results

The application was deployed in a university classroom, with many of them not having taken any classes on the topics. Students were given a pre-survey, asked to use the application, and then given a post-survey after finishing. The pre-survey asked students to list their grade level and what type of learner they considered themselves to be (such as Kinetic, Visual, etc.) Additionally, students were asked to rate themselves from 1-5 on how well they understood internetworking, routing, and cybersecurity [23]. The post-survey [24] had students take a quiz to gauge their retention of the topics presented. Students were asked to rerate their understanding scores after using the program [13]. Finally, students were asked qualitative questions about their opinion on CSPCS and what they felt could be improved. The sample size for the data was n = 30, with 8 Sophomores, 8 Juniors, 2 Seniors, and 12 Graduate Students. All data was gathered via a google forms survey [25].

A. Average Understanding of the Topics

Fig. 9 shows the subject understanding of students. After using the application, students had a noticeable increase in their self-evaluated understanding of all three topics presented. The average score for all three topics was 2.21/5 before and 3.21/5 afterward. This shows an increase of one point across the board on average and a percent change of 45.21% in students' self-evaluated confidence [23], [24].

B. Correlation Between Learning Style and Retention

Due to the visual nature of the program, it was hypothesized that users who identified most with the visual learning style would benefit the most from CSPCS. The majority of users identified as either visual (n = 16) or kinetic (n = 12), with only a fraction identifying as auditory learners (n = 2). As shown in Fig. 10, there was no significant difference in evaluation scores among different learning styles, with averages of 90% for visual, 88.33% for kinetic, and 90% for auditory learners. This has a few possible implications. The first is that the program had similar effectiveness in information retention among users, regardless of their learning style. However, recent research has called into question the importance of learning styles as a concept [26]. Regardless, the average grade on the retention quiz ignoring learning styles was around 90%. It is possible that this result was skewed by having questions that were too easy. Out of the five questions, two of them had all 30 correspondence answers correctly. After removing these questions from the average, the average score drops to 83.33% for visual, 80.56% for kinetic, and 83.88% for auditory learners. This data still demonstrates an overall understanding of the content, as well as little difference between students of different learning styles [23], [24].

C. User Feedback

Part of the post-survey focused on asking qualitative questions. This was done to see what users did and did not like about CSPCS and what could be improved. The data showed an overall positive reception, as shown in Fig. 11, with students on average rating the education benefits of the program at 3.97/5 and interest in using the program if improved at 4.03/5. Additionally, there was a section where users were encouraged to type out any specific feedback that they thought could improve CSPCS. The leading suggestions we received were to provide more in-depth explanations of the content, clarify the instructions, and improve the user interface. Regarding the last suggestion, a few users had issues that caused the content not to be displayed properly, possibly due to the browser they were using and its plugins. Overall, the user-friendliness rating was only 3.17/5, which

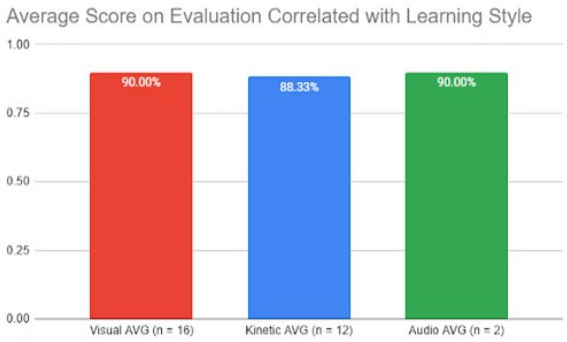

Fig. 10. Graph of quiz results correlated with learning style. Very little difference was found in scores between the three styles.

was lower than we had hoped. After implementing the proposed suggestion, we believe that our overall experience score will rise from 3.67/5 to 4/5 or higher in a future test [23], [24].

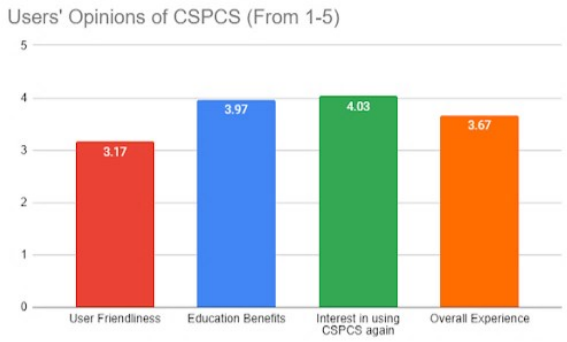

Fig. 11. User opinions of the user friendliness, education value, interest in using again, and overall experience of CSPCS.

VI. Conclusion and Future Works

In conclusion, students who used the application reported a positive effect on their understanding of all three topics addressed in the application, with an average percentage change of 45% from before to after. Additionally, students responded positively to learning about internetworking and cybersecurity with a visual application. No strong correlation was found between learning style and retention of information from this program.

The immediate changes planned for this program are based on the user suggestions. We plan to streamline the user interface and add more detail about the presented algorithms. The current version is somewhat abstracted not to overwhelm the user, but we plan to make the simulation much more in-depth for additional levels. A significant feature we plan to add is a simulated command-line interface (CLI) to teach users how they would perform these operations on an actual computer.

In addition to the changes based on user feedback, there are several changes that could be made to broaden the audience of the application and to expand it for an Information and Communication for Development (ICT4D) context. The current version of this application is designed to assist students in continuing their computer science education, but currently that scope only includes an audience that is English speaking, not visually impaired, and has access to a computer. We believe that by addressing the limited scope of the current application and expanding it, the application would be more effective in narrowing the digital divide. Possible changes to expand the audience of the application include adding language options, accessibility options, including a text to speech mode, and exporting the application to mobile platforms. This last change would address the fact that the current application requires the user to have a traditional computer by bringing it to the more common mobile platform.

Acknowledgements

We thank the NSF: Project Works Studio, SURE: Summer Undergraduate Research Experience, and Creative fellowship programs for this project.

References

[1] X. Chen, "Stem attrition: College students' paths into and out of stem fields. statistical analysis report. nces 2014-001." *National Center for Education Statistics*, 2013.

[2] D. B. Silva, R. de Lima Aguiar, D. S. Dvconlo, and C. N. Silla, "Recent studies about teaching algorithms (cs1) and data structures (cs2) for computer science students," in *IEEE Frontiers in Education Conference (FIE)*, 2019, pp. 1–8.

[3] R. Santo, L. A. DeLyser, J. Ahn, A. Pellicone, J. Aguiar, and S. Wortel-London, "Equity in the who, how and what of computer science education: K12 school district conceptualizations of equity in 'cs for all' initiatives," in *2019 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*, 2019, pp. 1–8.

[4] ISKME, "About oer commons," Online, 2022, accessed: 21-June-2022. [Online]. Available: https://www.oercommons.org/about

[5] H. S. Narman, C. Berry, A. Canfield, L. Carpenter, J. Giese, N. Loftus, and I. Schrader, "Augmented reality for teaching data structures in computer science," in *IEEE Global Humanitarian Technology Conference (GHTC)*, 2020, pp. 1–7.

[6] P. Buckley and E. Doyle, "Gamification and student motivation," *Interactive learning environments*, vol. 24, no. 6, pp. 1162–1175, 2016.

[7] T. L. Kingsley and M. M. Grabner-Hagen, "Gamification: Questing to integrate content knowledge, literacy, and 21st-century learning," *Journal of adolescent & adult literacy*, vol. 59, no. 1, pp. 51–61, 2015.

[8] "Khan academy," Online, accessed: 19-Apr-2022. [Online]. Available: https://www.khanacademy.org/

[9] "Cybersecurity games," Online, accessed: 19-Apr-2022. [Online]. Available: https://it.tamu.edu/security/cybersecurity-games/index.php

[10] "Cybersecurity games," Online, accessed: 19-Apr-2022. [Online]. Available: https://www.cisa.gov/cybergames

[11] E. Atwater and C. Bocovich, "CS4G netsim," Online, June 2017, accessed: 19-Apr-2022. [Online]. Available: https://netsim.erinn.io/
[12] S. Halim, "Visualgo," Online, accessed: 19-Apr-2022. [Online]. Available: https://visualgo.net/en
[13] "PWS CSPCS," Online, accessed: 19-Apr-2022. [Online]. Available: https://pws-cspcs.herokuapp.com/
[14] Unity, "Unity user manual 2021.3 (LTS)," Online, 2021, accessed: 19-Apr-2022. [Online]. Available: https://docs.unity3d.com/Manual/UnityManual.html
[15] "WebGL 2.0 api quick reference guide," Online, 2019, accessed: 19-Apr-2022. [Online]. Available: https://www.khronos.org/files/webgl20-reference-guide.pdf
[16] "Heroku," Online, accessed: 19-Apr-2022. [Online]. Available: https://www.heroku.com/about
[17] S. C. Carlson, "Graph theory," Online, Nov 2020, accessed: 19-Apr-2022. [Online]. Available: https://www.britannica.com/topic/graph-theory
[18] E. W. Weisstein, "Weighted graph," Online, accessed: 19-Apr-2022. [Online]. Available: https://mathworld.wolfram.com/WeightedGraph.html
[19] A. Khanna and J. Zinky, "The revised arpanet routing metric," *ACM SIGCOMM Computer Communication Review*, vol. 19, no. 4, pp. 45–56, 1989.
[20] A. Trivedi, "What is ping?" Online, Sep 2021, accessed: 19-Apr-2022. [Online]. Available: https://www.geeksforgeeks.org/what-is-ping/
[21] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "Digital identity guidelines," Online, June 2017, accessed: 19-Apr-2022. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-3.pdf
[22] N. Parlante, "Binary trees," Online, 2000, accessed: 19-Apr-2022. [Online]. Available: http://cslibrary.stanford.edu/110/BinaryTrees.html
[23] "CSPCS pretest," Online, accessed: 19-Apr-2022. [Online]. Available: https://bit.ly/3EDOT94
[24] "CSPCS posttest," Online, accessed: 19-Apr-2022. [Online]. Available: https://bit.ly/3Oul66W
[25] "Google forms," Online, accessed: 19-Apr-2022. [Online]. Available: https://www.google.com/forms/about/
[26] P. R. Husmann and V. D. O'Loughlin, "Another nail in the coffin for learning styles? disparities among undergraduate anatomy students' study strategies, class performance, and reported vark learning styles," *Anatomical sciences education*, vol. 12, no. 1, pp. 6–19, 2019.