Board 59: WIP: Lab Container: An environment to manage a student's time to complete programming labs while providing effective feedback from course staff

Mr. Yu Sheng Pan, University of Toronto

Yu Sheng Pan is a fourth year computer engineering student at University of Toronto who will be pursuing a MBA degree at the Rotman School of Management in fall of 2023.

Mr. Aniruddha Redkar, University of Toronto

I am a fourth-year computer engineering student at the University of Toronto. With my technical skills and creative mindset, I am determined to make a significant impact in the field of technology.

Sowrov Talukder, University of Toronto

Sowrov Talukder is a Computer Engineering student at the University of Toronto helping to improve programming labs in education.

Mr. Parth Sindhu, University of Toronto
Dr. Hamid S. Timorabadi, University of Toronto

Hamid Timorabadi received his B.Sc, M.A.Sc, and Ph.D. degrees in Electrical Engineering from the University of Toronto. He has worked as a project, design, and test engineer as well as a consultant to industry. His research interests include the applicati

WIP: Lab Container: An environment to manage a student's time to complete programming labs while providing effective feedback from course staff

Abstract

Lab Container is a web application aimed at enhancing the experience of students and instructors in programming courses. The tool provides a containerized and interactive environment for students to complete lab assignments and fosters collaboration between students and instructors. Lab Container includes a comprehensive inventory of resources, including compilers and libraries, and enables instructors to track students' progress and provide real-time feedback on their code. The frontend of the platform is built using React JS, while the backend employs microservice architecture, where the microservices communicate with each other to provide a seamless and integrated experience for all users. An empirical study was conducted to assess the usability and functionality of Lab Container, with results indicating that the tool effectively reduced the amount of time students spent on lab assignments, and improved collaboration between students and instructors. The design of Lab Container represents a step forward in computer science education and has the potential to transform the way students engage with lab assignments.

Background

The issue of students over-allocating time to complete programming lab assignments in a course and falling behind other subjects has been a persistent challenge in computer science education. This problem is exacerbated by the difficulty that instructors face in reading and providing feedback on student code, as well as the limited information they received from auto-graders. This leads to a situation where students are forced to study less in non-programming courses, skip more lectures, and fall behind on those courses as a result.

This project introduces a solution to this challenge in the form of Lab Container. Lab Container is a flexible software tool that provides students with a containerized environment to complete their lab assignments. It also includes all necessary lab resources, such as libraries and compilers, which are configured by the course staff, and allows for ongoing progress tracking by instructors. With all the necessary tools provided within each student's container, they are able to access these resources anywhere and be able to complete assignments gradually rather than in one sitting. This not only reduces the student's working time but also provides instructors with information about the student's work, such as time spent, version history, and contribution from each team member. The software was tested on two groups of current engineering students from the University of Toronto with a focus on usability and functionality. The first group was on a small group of upper-year students and then the second group was on a larger group with students of different years. The results of the project highlight how the project can be best used to improve education for both students and instructors in programming courses.

Challenges with Programming Courses

The current implementation of programming courses presents several challenges for students and instructors alike. One major challenge is the allocation of time for lab assignments, as the provision of starting code and an automated testing tool (auto-grader) often leads to students over-devoting their time to complete the assignments. This can result in students prioritising coding labs over other courses as research has indicated that students often experience difficulties in balancing the demands of multiple courses and may prioritise assignments based on deadlines and perceived difficulty, leading to reduced engagement in other subjects [1]. Studies have also shown that the use of automated grading systems in programming courses can limit the effectiveness of formative feedback and lead to a narrow focus on syntax, rather than conceptual understanding [2]. On the other hand, the variability of coding implementations and methods among students can make it very time-consuming for instructors to understand the existing code before providing feedback, further hindering the educational experience. The reliance on automated grading as the primary metric for assessment also ignores key information regarding a student's approach and understanding of the assignment, which is a crucial element of engineering and computer science education.

Related Works

The University of Toronto has implemented two solutions for programming labs - PCRS and MarkUs. PCRS, an application designed by a professor at the university, provides interactive programming exercises for Python, C, Java, and Relational Algebra and SQL. MarkUs is a platform that allows students to submit their code for feedback, testing, and grading. However, a significant drawback of both these solutions is that there is no way for course staff to monitor the time spent by students on their programming labs. This lack of monitoring allows students to allocate excessive time towards programming labs, adversely impacting their other coursework. To address this challenge, Lab Container offers a comprehensive platform for creating and completing programming labs while simultaneously enabling course staff to track student progress and time spent on labs to prevent over-investment of time in programming labs.

A Better Learning and Teaching Experience

To create a better experience for students and instructors in programming courses, several approaches can be taken. One approach is to incorporate active learning strategies, such as pair programming and group projects, which can help students engage with the material and develop a deeper understanding. This approach has been shown to significantly improve student learning outcomes, such as increased success rates in introductory courses, increased retention in the major, higher quality software, and higher student confidence in solutions [3]. Fostering a collaborative learning environment through pair programming and group projects can also help students develop teamwork and communication skills, creating a more enjoyable and supportive learning environment. Providing personalised feedback can address specific areas where improvement is needed and provide guidance on coding implementations and methodologies. Focusing on the problem-solving process, rather than just the final solution, can help students develop critical thinking and problem-solving skills [4]. Finally, utilising technology tools that track students' progress and provide instructors with information about their work can also improve the experience for both students and instructors.

Lab Container is an effective solution that builds on the above suggestions for creating a better experience for students and instructors in programming courses. It provides a flexible platform for students to complete programming lab assignments individually or in teams, while giving instructors the ability to easily monitor their progress and provide real-time feedback on their code. Most importantly, Lab Container encourages students to incrementally work on their labs instead of cramming right before the deadlines. This promotes better time management and reduces the amount of time needed to spend on assignments. Lab Container can enhance the learning experience for students and instructors in programming courses by facilitating collaboration, encouraging a focus on the problem-solving process, and making lab assignments less time-consuming for students.

Technical Design

System Level Overview

The design of Lab Container is a web application that consists of a reactive frontend and an efficient backend. The frontend is implemented using React JS, a widely adopted JavaScript library for building user interfaces. It features three main components: the Authentication UI, Student Working Environment, and Dashboard, each of which serves a specific purpose in enhancing the user experience.

The backend of Lab Container is designed with a microservice architecture, which allows for a scalable and flexible solution. The architecture consists of four independent microservices, including Authentication, Analytics, Feedback, and Student Manager Services, each of which is responsible for a specific aspect of the application. Furthermore, the backend includes three independent Postgres databases that provide reliable and secure data storage for the application. The design for Lab Container was chosen to provide a seamless and efficient experience for users and meet the demands of modern web applications.

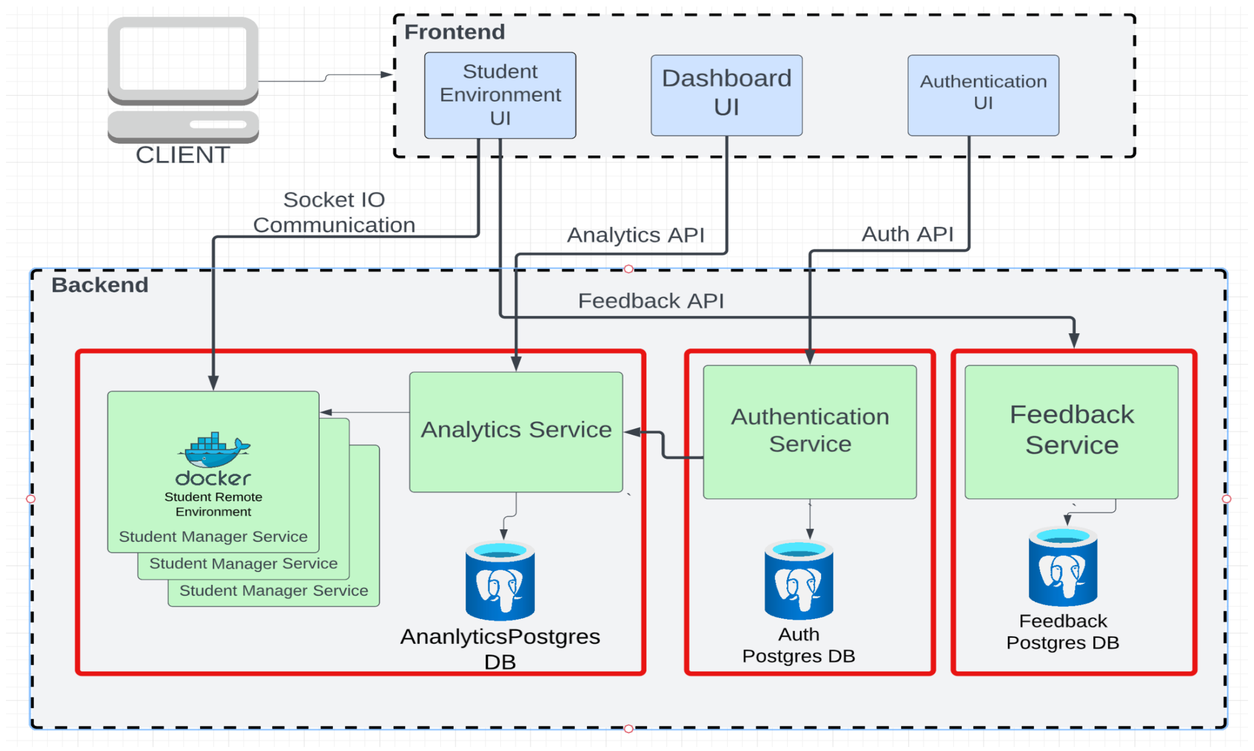


Figure 1: System Level Diagram

Module Level Overview

Student Environment UI

The Student Working Environment module of Lab Container refers to the frontend interface where students can complete their assignments on the web. This module integrates key features such as code editing, terminal access, file management, milestone tracking, and real-time feedback to enhance the overall learning experience. These subcomponents work together to provide a comprehensive and streamlined solution for students to work on their assignments.

Authentication UI

The Authentication module encompasses the frontend components related to the creation and management of user accounts as well as the process of logging in to the Lab Container. This module comprises three key components: the Login page, the Signup page, and the Forgot Password functionality. The Login page enables users to securely enter their credentials. The Signup page provides a convenient and straightforward process for creating new user accounts.

The Forgot Password functionality allows users who have lost or forgotten their password to reset it in a secure and efficient manner.

Dashboard UI

The Dashboard UI module pertains to the frontend pages of the student and course staff dashboards within Lab Container. The student dashboard provides an interface for students to monitor their progress on laboratory assignments and manage their teams, while the course staff dashboard enables the creation and customization of lab assignments, as well as the monitoring of student progress.

Authentication Service

The Authentication Microservice is a component of the backend architecture in Lab Container. It provides a secure and centralised API for authentication and authorization management. The microservice implements JSON Web Token (JWT) authentication and refresh token mechanism, ensuring secure transmission of information between parties. The microservice is responsible for the creation and authentication of user accounts through JWT tokens.

Analytics Service

The Analytics Service refers to the Lab and Team Management microservice. This module serves as the central hub, providing API functionality for the creation, organisation, and management of lab assignments, milestones, and student teams. It is responsible for maintaining communication with the Container Runtime in order to initiate new instances of the Student Manager Service.

Student Manager Service

The Student Manager Service refers to the student container service that is assigned to each student for every lab assignment. The service is dynamically instantiated as students start each assignment. The microservice comprises three key components:

1. Socket IO API: It facilitates the connection between the frontend environment and the microservice, providing a secure and efficient interface for the web terminal.

2. File Manager API: It enables students to perform file operations such as saving, viewing, and editing within their individual containers.
3. Progress Tracking and Reporting Component: This component tracks and reports student progress and performance metrics to the analytics service in a timely manner.

Feedback Service

The Feedback Microservice offers APIs that facilitate communication and exchange of feedback between students and course staff regarding their progress and performance in the assigned labs. This microservice enables students to request assistance from the course staff during milestones, and for the course staff to provide constructive feedback to support student progress and learning.

Functionality Testing and Verification

The test plan for Lab Container is designed to gather feedback from student and instructor volunteers. The team created two surveys - a Consent Form And the Feedback Survey. The Consent Survey is used to inform potential volunteers about the testing process and to obtain their consent to participate. The survey asks if they are willing to complete a Feedback Questionnaire at the end of the testing period, and also informs them of their right to revoke their consent at any time.

The Feedback Survey is the main tool for collecting feedback from the volunteers. The survey consists of three sections: usability, functionality, and open-ended questions. The usability section includes 10 statements that are scored on a scale of Strongly Disagree to Strongly Agree. The functionality section includes 10 items that are scored on a scale of Very Unsatisfied to Very Satisfied. The open-ended sections include two questions that ask about the strengths and weaknesses of the application, with space for users to provide more detailed feedback.

The test plan has two stages, both of which start with the collection of volunteers through the Consent Survey. In stage one, the team recruited five engineering students as volunteers to use the Lab Container for a three-day period. Upon completion of the testing period, the volunteers completed the Feedback Survey to report their experience with the application and any bugs found. The results of the survey were analysed, and improvements were made based on the scores and open-ended answers. In stage two, the team recruited a total of 30 lower-year engineering students to use the improved version of the Lab Container for two weeks. At the end of this period, the Feedback Survey was given to each volunteer, and the overall usability and functionality scores were calculated.

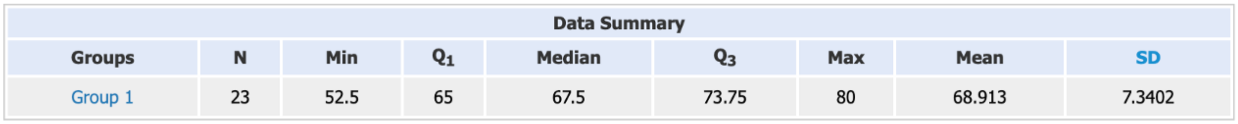
The usability and functionality scores were calculated using a method adapted from the System Usability Scale (SUS). Each response to each question is given a score of 0 to 4, with 0 being Strongly Disagree/Very Unsatisfied and 4 being Strongly Agree/Very Satisfied. The scores for all 10 questions are added together and multiplied by 2.5 to convert the range from 0-40 to 0-100. It is important to note that the final score is not a percentage even though the range is 0-100. The average SUS score is 68 [5].

Survey and Feedback Results

The preliminary evaluation of Lab Container was conducted with a small group of five current engineering students, who each provided valuable insights through the Feedback Survey. The results indicated a largely positive response with regard to the application's responsiveness and overall design of the user interface. On the other hand, some challenges were identified in the areas of interface complexity and team collaboration, which received comparatively lower scores. The feedback from the open-ended questions further emphasised the usefulness of the milestone-tracking feature and the attractive user interface. However, some areas of improvement were identified, including the creation of teams for labs and some backend connection issues. Overall, the average usability score was 67 and the average functionality score was 65, slightly below the team's desired target of 68.

The team took the feedback and comments from the first round of testing and made several improvements to the application before proceeding to the second round of testing. The team improved the usability of the application by simplifying the user interface for both students and instructors. The first round of testing showed concerns with usability. Buttons and tabs were made so that users know what actions will occur without any confusion. Text was also put in places where they don't clutter up space and are easy to read and close. Another major area of feedback from students was on the ease of team collaboration in the application due to many programming labs being group labs. These improvements were made in four months after the first round of testing. The second round of testing showed improvements in usability and more specific feedback was given about the team's page and bugs. The team section on the students page was made so that many processes were automated such as selecting the correct lab and joining a team that the user created. Finally, testing identified backend bugs that impacted usability and the functionality.

After compiling the survey results of 30 undergraduate engineering students, the results were shown to have noticeable improvements, with the average usability score increasing to 69.33 and the average functionality score improving to 68.23, meeting the team's desired target goal. These results were highly encouraging, demonstrating that the team's efforts to improve the application were successful. The data summaries of the second round of Usability and Functionality scores can be seen in the diagrams below.



Data Summary								
Groups	N	Min	Q_1	Median	Q_3	Max	Mean	SD
Group 1	23	52.5	65	67.5	73.75	80	68.913	7.3402

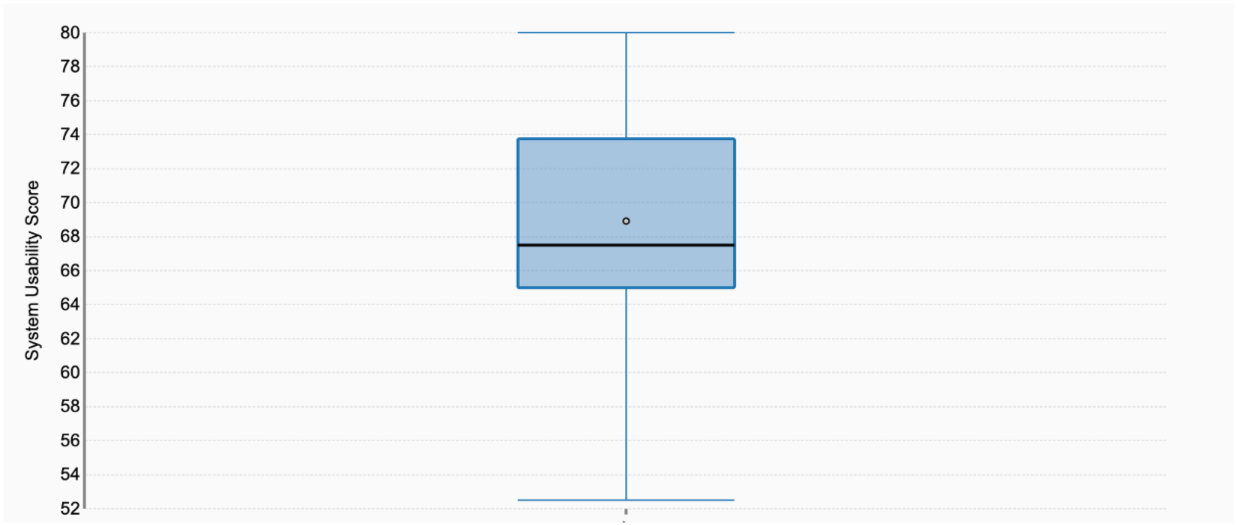
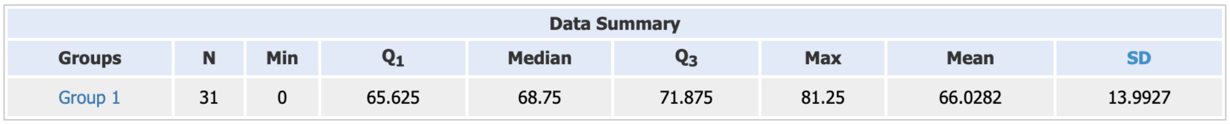


Figure 2: Data Summary - System Usability Score



Data Summary								
Groups	N	Min	Q_1	Median	Q_3	Max	Mean	SD
Group 1	31	0	65.625	68.75	71.875	81.25	66.0282	13.9927

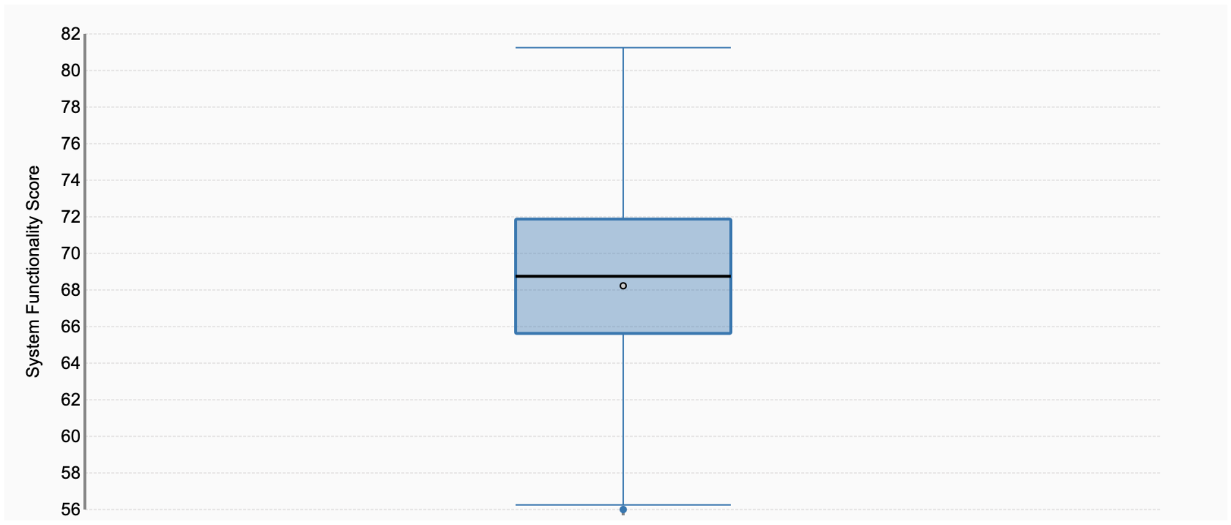


Figure 3: Data Summary - System Functionality Score

The team would like to emphasise the results of two specific questions from the testing survey [seen in figures 4 & 5], which asked volunteers to rate the statements: "I believe this application can help me save time when completing assignments" and "If the system is functional, I would use the system frequently". The results indicated that 20 out of 30 respondents believed that the application can help them save time when completing assignments, with only 4 respondents disagreeing with this statement. Additionally, 19 out of 30 respondents indicated that they would use the system frequently if it was functional, with only 3 respondents indicating that they would not. These results demonstrate that Lab Container has the potential to be a valuable tool for students in computer science education, providing them with an effective and efficient way to complete their assignments and manage their lab work.

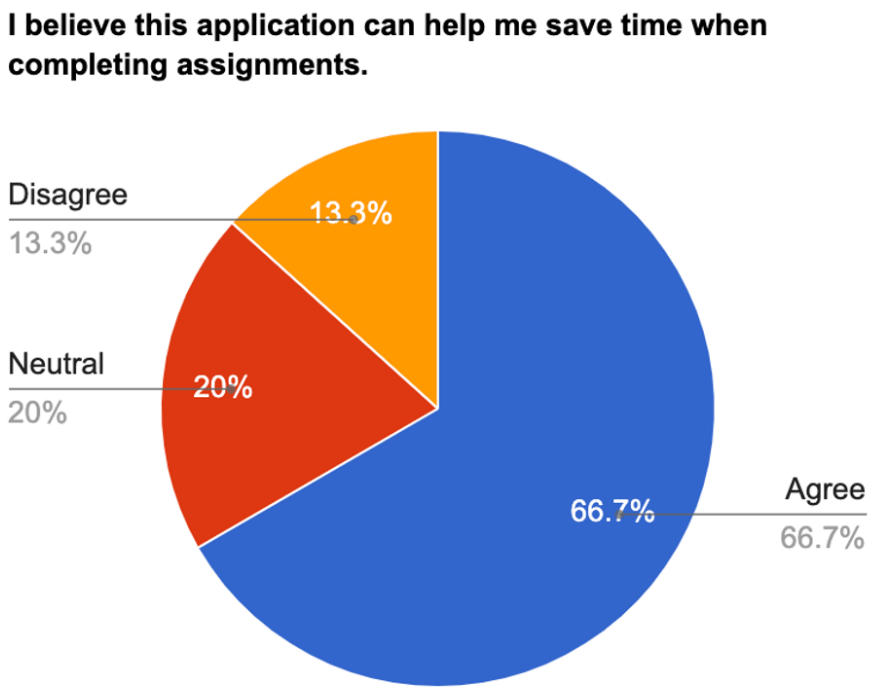


Figure 4: Survey Results - “I believe this application can help me save time when completing assignments”

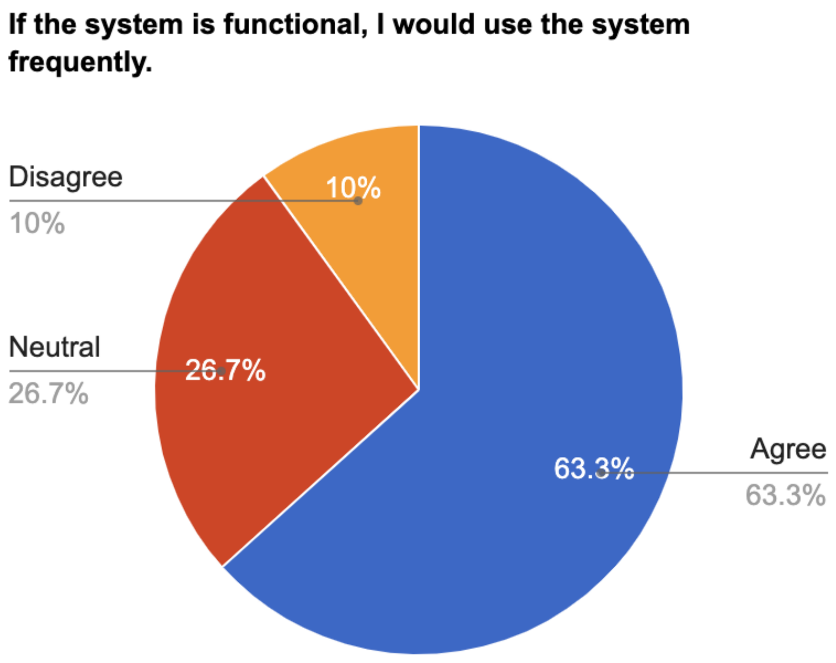


Figure 5: Survey Results - “If the system is functional, I would use the system frequently.”

Future Work

The preliminary testing of Lab Container has provided helpful insights into areas of improvement for Lab Container. The results of the surveys will be used to inform further developments to Lab Container before it is tested on a larger group of volunteers.

In order to improve the overall user experience, the team will work on streamlining the interface to make it more simple and intuitive. The team will also work on enhancing the team collaboration features to make it easier for students to work on lab assignments. Furthermore, the team will address the backend connection issues to ensure a more consistent experience for all users.

In the future, the team plans to conduct a larger-scale study with a larger group of volunteers to further validate the effectiveness of Lab Container in improving the overall quality of programming courses. The team will also consider expanding the scope of the application to support other programming platforms to make Lab Container a more versatile tool.

Conclusion

In conclusion, the team has developed a web application called Lab Container that provides an interactive environment for programming students to complete lab assignments. The tool provides all necessary resources for students and allows instructors to track students' progress. The results of the preliminary testing showed that the most favourable responses were for the categories *responsiveness* and *look and feel*, while *interface complexity* and *team collaboration* had the least favourable responses. These findings provide valuable insights into the usability and functionality of the application and serve as a guide for future developments. The study highlights the potential of Lab Container to revolutionise computer science education by making programming assignments much less tedious and providing a foundation for future research in this area. The team is committed to continue improving the application and making it more user-friendly for all users.

References

[1] Person, P. Paul, and R. Ramsden, “Learning to teach in higher education: Paul Ramsden, Paul Ramsden: T,” *Taylor & Francis*, 05-Dec-1991. [Online]. Available: https://www.taylorfrancis.com/books/mono/10.4324/9780203507711/learning-teach-higher-education-paul-ramsden-paul-ramsden. [Accessed: 05-Feb-2023].

[2] A. Leite and S. A. Blanco, “Effects of Human vs. Automatic Feedback on Students’ Understanding of AI Concepts and Programming Style.” [Online]. Available: https://arxiv.org/pdf/2011.10653.pdf. [Accessed: 06-Feb-2023].

[3] B. Hanks, S. Fitzgerald, R. McCauley, L. Murphy, and C. Zander, “Pair programming in education: A literature review.” [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/08993408.2011.579808. [Accessed: 05-Feb-2023].

[4] R. E. Mayer, “Teaching and learning computer programming: Multiple research perspec,” 30-Sep-2013. [Online]. Available: https://www.taylorfrancis.com/books/mono/10.4324/9781315044347/teaching-learning-computer-programming-richard-mayer. [Accessed: 05-Feb-2023].

[5] R. I. N. T. U. BISWAS and B. Deans, “Here's what we learned about Page Speed,” Backlinko, 08-Oct-2019. [Online]. Available: https://backlinko.com/page-speed-stats. [Accessed: 24-Dec-2022].

[6] Gulati, N., & Higgy, A., & Timorabadi, H. (2022, August), *Work In Progress: CodeCapture: A Tool to Attain Insight into the Programming Development Process* Paper presented at 2022 ASEE Annual Conference & Exposition, Minneapolis, MN. https://peer.asee.org/40663

Appendix

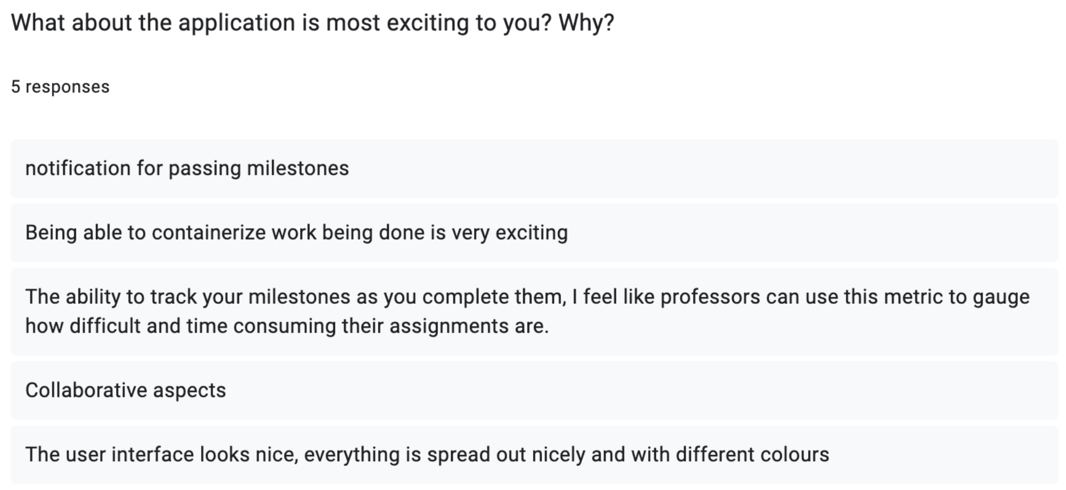
Appendix A: Preliminary Test Results

Scaled usability and functionality scores based on the survey results. The scores are scaled to be in the range of 0-100, and the averages are calculated and shown in the last row of the table. The first round of testing was done with 5 participants.

Participant	Scaled Usability Score	Scaled Functionality Score
1	47.5	56.25
2	80	71.875
3	80	59.375
4	62.5	68.75
5	65	68.75
Average	67	65

Appendix B: Preliminary Test Open-ended Question Feedback

The feedback from the preliminary test participants regarding the two open-ended questions in the Feedback Survey



What about the application is most exciting to you? Why?

5 responses

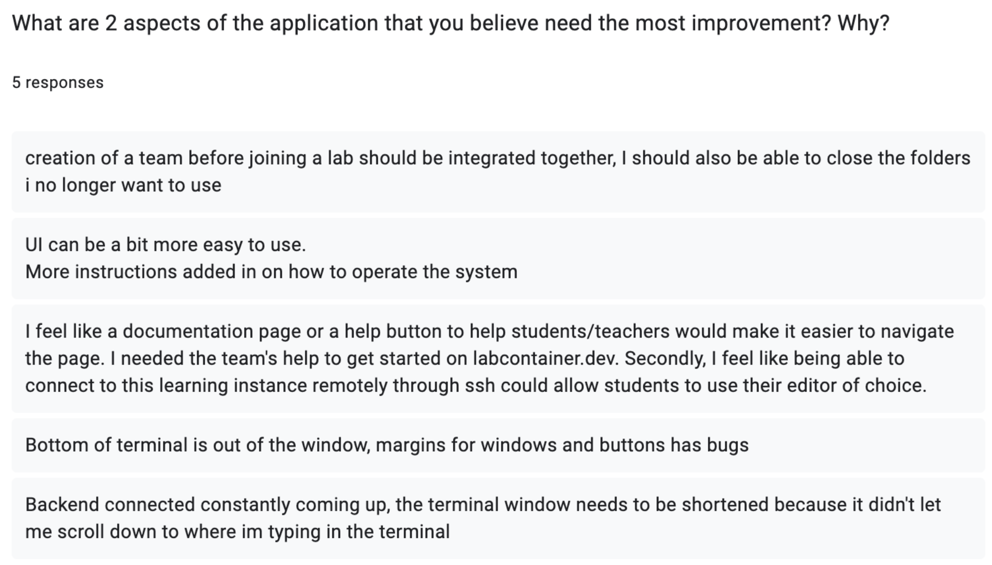
notification for passing milestones

Being able to containerize work being done is very exciting

The ability to track your milestones as you complete them, I feel like professors can use this metric to gauge how difficult and time consuming their assignments are.

Collaborative aspects

The user interface looks nice, everything is spread out nicely and with different colours



What are 2 aspects of the application that you believe need the most improvement? Why?

5 responses

creation of a team before joining a lab should be integrated together, I should also be able to close the folders i no longer want to use

UI can be a bit more easy to use.
More instructions added in on how to operate the system

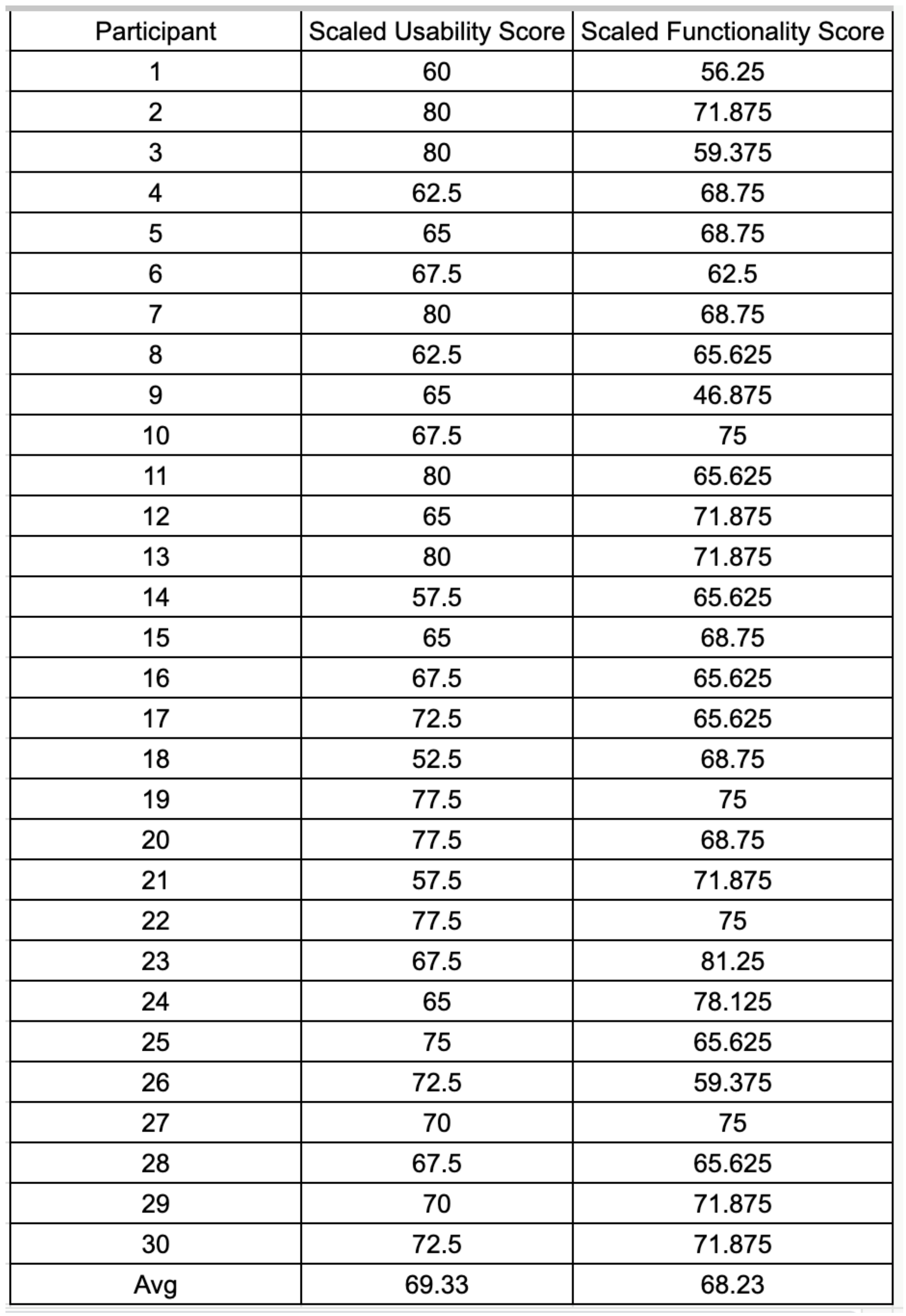
I feel like a documentation page or a help button to help students/teachers would make it easier to navigate the page. I needed the team's help to get started on labcontainer.dev. Secondly, I feel like being able to connect to this learning instance remotely through ssh could allow students to use their editor of choice.

Bottom of terminal is out of the window, margins for windows and buttons has bugs

Backend connected constantly coming up, the terminal window needs to be shortened because it didn't let me scroll down to where im typing in the terminal

Appendix C: Second-Round Testing Results

Scaled usability and functionality scores based on the second round of survey results. The scores are scaled to be in the range of 0-100, and the averages are calculated and shown in the last row of the table. The first round of testing was done with 30 participants.



Participant	Scaled Usability Score	Scaled Functionality Score
1	60	56.25
2	80	71.875
3	80	59.375
4	62.5	68.75
5	65	68.75
6	67.5	62.5
7	80	68.75
8	62.5	65.625
9	65	46.875
10	67.5	75
11	80	65.625
12	65	71.875
13	80	71.875
14	57.5	65.625
15	65	68.75
16	67.5	65.625
17	72.5	65.625
18	52.5	68.75
19	77.5	75
20	77.5	68.75
21	57.5	71.875
22	77.5	75
23	67.5	81.25
24	65	78.125
25	75	65.625
26	72.5	59.375
27	70	75
28	67.5	65.625
29	70	71.875
30	72.5	71.875
Avg	69.33	68.23