A Secure and Scalable Virtual Lab Platform for Computing Education

Richard Wing Cheung Lui[1,*], Aiden Wen Yi Zhang[1], and Philip Tin Yun Lee[2]

[1]Hong Kong Polytechnic University, Hung Hom, Hong Kong
[2]Hong Kong Shue Yan University, North Point, Hong Kong
Email: cswclui@polyu.edu.hk (R.W.C.L.); aiden-wy.zhang@polyu.edu.hk (A.W.Y.Z); tylee@hksyu.edu (P.T.Y.L)
[*]Corresponding author
Manuscript received June 12, 2023; revised July 31, 2023; accepted August 5, 2023; published January 16, 2024

***Abstract*—With the increasing importance of online teaching and learning, conducting hands-on labs online brings a number of challenges, such as difficulties in setting up and maintaining consistent lab environments across different operating systems and compatibility issues with students' local software and hardware environment. This paper aims to propose a secure and scalable virtual lab platform using container technology and open-source cloud-native technology. Instructors can configure and deploy customized lab environments in the cloud, and students may use web browsers to gain access to a sandboxed lab environment and live coding environment for online and self-paced learning. With the platform, students no longer need to install custom software on their computers and ensures the compatibility of the lab environment in online teaching and learning activities. A pilot study was conducted with 370 undergraduate and postgraduate students from the Hong Kong Polytechnic University. The platform demonstrated good scalability to support large classes of students to complete hands-on lab activities online. The survey results indicated that students perceived the platform as easy to use, helpful for learning online, and intended to continue using it beyond the course.**

***Keywords*—container-based platforms, interactive learning environment, cloud computing, online learning**

I. Introduction

In today's highly competitive environment, it is essential for students to keep up with the latest industry trends in information technology and keep their hands-on technical skills up to date. Traditionally, instructors create lab exercises with step-by-step instructions where students may complete hands-on lab exercises in pre-configured lab environments. With the increasing importance of online teaching and learning, conducting hands-on labs online brings a number of challenges [1, 2]. Students may fail to set up the lab environment properly on their own computers due to incompatibility with the operating system or other installed software. Students may also fail to complete the assigned tasks due to differences between the instructors' and students' lab environments. As the instructors have no physical access to the students' computers and terminals, it is challenging for the instructors to track students' participation and progress. Also, it is difficult for the instructors to provide personalized feedback to the students and assist the weaker students during the online lab sessions.

There is an increasing interest in platforms where learners can practice coding and other hands-on skills online. Some interactive learning platforms, such as repl.it [1] and Gitpod [2], provide a web-based Integrated Development Environment (IDE) to help learners practice programming skills and code in a real-time collaborative editor with team members for their group projects [2, 3]. Katacoda [3], an interactive Learning and Training Platform for Software Engineers, provides a number of free tutorials related to Git, Docker, Kubernetes, Linux, Machine Learning and programming languages where a pre-configured sandboxed environment is deployed for learners to develop coding and hands-on skills through step-by-step tutorials using a web browser. Despite the rich functionality and features for learning content customization, no persistent storage is provided for students to store their files and the lab session time is limited. The platform's performance is sometimes unstable, and the response time increases significantly when a group of students are accessing the lab environment at the same time.

This paper aims to propose a secure and scalable Virtual Lab Platform (VLP) for interactive skill-based learning and assessment for Computing education using container technology and open-source cloud-native applications. Instructors can create customized interactive step-by-step lab scenarios and deploy customized lab environments for various Computing related courses. Students may use web browsers to walk through the guided and contextual steps in the interactive lab exercises that run alongside a sandboxed lab environment and a live coding environment. Students may engage in self-paced learning, perform coding and conduct lab-based training at any time without installing any software in the local environment. This study contributes to the existing body of knowledge by presenting a case study on the design and implementation of a secure and scalable virtual lab platform using container and state-of-the-art cloud native technology. Also, it presents empirical evidence through a pilot study demonstrating the platform's effectiveness in enhancing the skill-based learning of Computing subjects.

This paper is organized as follows. In Section II, the literature related to interactive skill-based learning platforms is reviewed. The architecture and design of VLP are discussed in Section III and our pilot study is discussed in Section IV. In Section V, the summary and future research directions are discussed.

[1] https://replit.com/

[2] https://www.gitpod.io/

[3] https://www.katacoda.com/

II. Literature Review

A Virtual Lab (VL) is a specific type of Virtual Learning Environment (VLE) to help learners practice problem solving and develop practical skills [4]. VLEs can be used to provide after-class learning opportunities to students with no geographical and time restrictions [5, 6]. For subjects making use of massive datasets and computational functionality, such as data science and artificial intelligence courses, the instructors may be concerned about the feasibility of equipping those facilities for a large number of students enrolled in the courses [4, 7, 8]. VLEs provide a possible solution to acquire large-sized datasets for students to complete assignments or extra studies [6]. VLEs offer a collaborative and interactive environment where students may get instant replies from the system [9]. CLaDS is a novel web-based VLE platform for data science courses [6], which provides hands-on experience for students, provides off-campus equipment, reduces equipment maintenance cost and time, scales for massive numbers, and includes large-scale datasets for analysis and evaluation. Green and Zhai [9] develop LiveDataLab on top of CLaDS, with the functionality to enable students to collaborate while playing with real-world datasets. Also, they have integrated the system with synchronous editing where students can edit their project codes at the same time. CvLabs [4] is a web-based learning platform where learners can sign up for labs that they have interest on. Lab guides are provided with assessments and detail procedures and students can choose their preferred learning pace for executing lab tasks. Cardoso *et al.* [10] incorporate their Virtual Programming Lab (VPL) into Moodle, a learning management system. Hu and his team [11, 12] developed a Collaborative Virtual Computer Lab (CVCL) to learn Unix and C programming. Iorio and Risso [13] introduced CrownLabs for remote computing laboratories, with also Kubernetes and Openstack to build the infrastructure. Katacoda is a commercial VLE for helping developers learn the latest IT technologies and skills. Katacoda allows the scenario authors to define the content for learning cloud technologies, such as Docker, Kubernetes and machine learning [4, 14], where students may complete lab-based training in a sandbox environment. Katacoda provides a markdown syntax for instructors to develop customized lab contents and annotate commands in the lab instructions such that learners may simply click the commands for execution in the terminals in the provisioned lab environment. Scenario creators may also provide scripts for automated checking of the completion of the lab tasks before proceeding to the next tasks. However, due to the increasing security concerns and resource demands required to keep the site running safely, O'Reilly Media decided to terminate public access to Katacoda.com in May 2022 and focused on integrating the learning platform within their paywall.

Open-source projects are also involved in VLE-based learning. Brunner and Kim [15] adopted Jupyter notebooks for an introductory data science course at the University of Illinois to introduce general programming concepts with Python. They believed that Jupyter Notebook, an open-source web application with a mix of code and text elements, is appropriate for interaction teaching. JupyterHub allows a group of users to gain access to computational environment such as Jupyter notebooks [15]. Shiny, a web-based framework for the R language, facilitates instructors to develop "interactive, dynamic, user-friendly, visually appealing and with similar functionality to Java/JavaScript applets" [16]. Stander and Valle [17] proposed a module "Understanding Big Data from Social Networks" for freshmen where R was chosen to be the primary language for the course due to the instructor's experience and its wide use on different operating systems. Li [18] deployed R Shiny applications with ShinyProxy for deploying container-based VLEs. Gitpod, an open-source and vendor-neutral online Web-IDE, was used to develop an online programming practice platform for support software engineering education with automatic feedback on solutions submitted by students [3]. Github classroom, an online platform developed by GitHub to help teachers manage and organize their programming assignments, was used to facilitate students' collaborative work using Git in a GIS programming course [19].

III. Design of the Virtual Lab Platform

A. Features and System Architecture

We aim to develop and evaluate a container-based Virtual Lab Platform (VLP) which can support shorter lab environment start-up time, better customization of the lab environment and provide a highly available persistent storage for students to complete their lab exercises in the cloud. Table 1 summarizes the differences between our virtual lab platform with other VLE platforms introduced in Section II.

We have adopted container technology to deploy our lab environments as it is more lightweight, less resource-consuming, and supports faster lab environment provisioning when compared with traditional virtual machines [20]. Fig. 1 shows the system architecture of our virtual lab platform. Our infrastructure is based on Kubernetes, an open-source container orchestration engine for managing containerized applications across multiple hosts [21]. We use a lightweight Kubernetes distribution K3s[4], which simplifies the deployment while preserving scalability and high availability. There are two types of roles for Kubernetes nodes: master and worker. Nodes with control plane components installed are master nodes and the others are worker nodes. Master nodes manage the state of the cluster and distribute the workload among the different nodes in the cluster. Meanwhile, worker nodes are used to deploy our lab infrastructure and students' lab environment. Pods are the smallest deployable units of computing that users can create and manage in Kubernetes, and a pod is a group of one or more containers sharing the pod's s resources. The students' lab environment is deployed as a Kubernetes pod running in one of the worker nodes. Currently, we have deployed 4 workers and 2 master nodes in our cluster. We can scale out easily by adding more worker and master nodes to handle more users and workload.

For each course, the course portal will be served by a

[4] https://github.com/k3s-io/k3s

"VLabController" pod in our Kubernetes cluster. The controller pod is also responsible for provisioning the lab environment. It acts as a proxy to forward the network traffic between the learners' browser and the pods running the lab environment, and it provides a set of APIs to manage the provisioning and management of the lab environments for teachers and students. We develop VLabController based on two open-source projects: ContainerProxy[5] and ShinyProxy[6], which supports a production-level deployment of multiple containerized applications with authentication. These projects are refactored to increase stability and scalability. We also add new features such as idle detection, session management, multi-container support and remote assistance (where teachers can connect to students' virtual lab environments to help them solve problems). Furthermore, we improve the event tracking and log components to monitor controller performances and collect usage data for further analysis.

Table 1. Comparison of virtual lab platforms' features

	Our Platform	CvLabs [4]	Katacoda
Virtualization	Container-based	Container-based	VM-based
Isolation	Process level	Process level	Guest OS level
Start-up time	Within seconds	Within seconds	Unpredictable
Lab environment	Customizable	Customizable	Limited Customization
Persistent storage	Highly-available	Supported	Not supported
Multiple containers	Supported	Not supported	N/A
Idle detection	Supported	Not supported	Supported

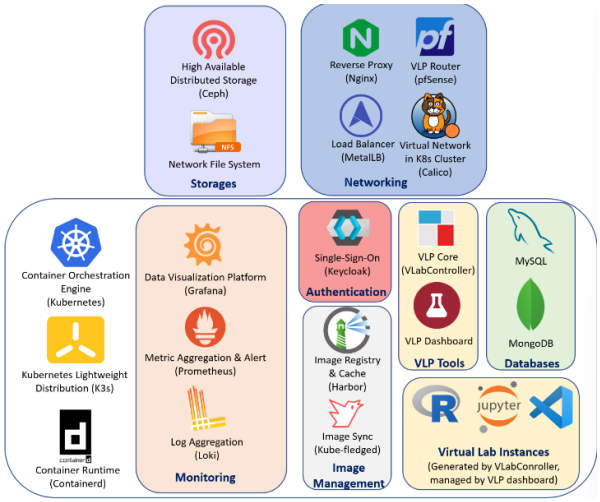

Fig. 1. System architecture of our virtual lab platform.

Fig. 2 shows how the lab environment for the learners is provisioned and managed. Each user in our system has a separate Kubernetes namespace to deploy pods for running his/her lab environment. To improve security, we have defined network policies in the namespace to isolate the network traffic between different users. By default, the files stored in the containers are ephemeral and will be lost when the lab environment and the corresponding pods are terminated. To support persistent storage across lab sessions, a Persistent Volume Claim (PVC) will be created in each user's namespace. When the lab environment is provisioned, the user's PVC will be mounted to the home folder of the user's pod. In this way, the files and software configuration stored under the root folder can persist across lab sessions. Each of our virtual lab environments is packaged as one or more Docker images. The images are uploaded to remote container registries (e.g., Docker Hub). During the provisioning of the lab environment, the image will be downloaded (pulled) from the repository (if it is not available in the worker node) to create the pod in one of the worker nodes in the cluster. However, Docker Hub imposes a rate limit years ago to restrict the number of image pulls based on individual IP addresses. To address the rate-limit issue and reduce the time and network bandwidth for pulling images from remote repositories, we deploy Harbor, an open-source container registry, for locally caching commonly used Docker images within the cluster.

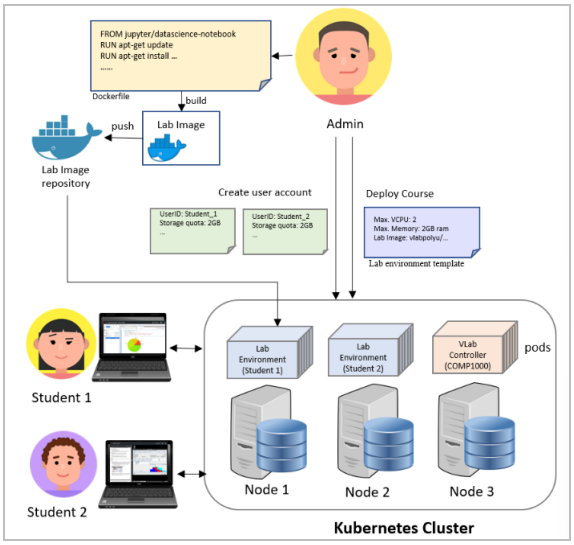

Fig. 2. Provisioning and managing the lab environment in a course.

Learners may provision different lab environments in our lab platform. To access the applications in the lab environment, we expose the different applications running at different ports in the lab environment through a URL. Nginx, an open-source software for web serving, is deployed as the reverse proxy for mapping the external URL to the internal IP addresses and ports of the applications within the user's lab environment. Our virtual lab platform also adopts other open-source cloud-native tools. The user accounts of students and teachers are managed by Keycloak, an open-source software that allows applications to use single-sign-on for authentication and authorization. For monitoring the health of our cluster, we make use of Prometheus, a time-series monitoring database, to scrape, query and monitor metrics from the nodes and the various pods in our clusters. We also develop interactive dashboards using Grafana, a popular interactive data visualization platform, to aggregate the logs metrics from Prometheus, Loki and other data sources to monitor the provisioned lab environment and collect usage data. Ceph [22] is a popular distributed object, block, and file storage platform, which provides excellent performance and reliability while promising unprecedented scalability. In our lab platform, Ceph is used to store the data from different applications running in our cluster (e.g., Keycloak and Prometheus) as well as files persisted in users' workspace. Network File System (NFS), a distributed file system, is also

[5] https://containerproxy.io/
[6] https://shinyproxy.io/

deployed in our cluster. As NFS folders can be mounted on pods in different namespaces with proper read/write permissions, teachers may store and distribute lab materials to students in their courses within our platform.

B. Defining and Provisioning the Lab Environment

Our lab environment is instantiated from one or more container images, which package the operating system and applications within the lab environment. The administrators can build and push custom Docker images to private or public image registries or reuse existing docker images. To define the lab environment, the admin should first create a lab environment template with one or more container definitions. Each container definition should reference a container image for the lab environment, the computational resource limits (e.g., the CPU and memory quota available to the pods) and the listening ports of the applications within the deployed lab environment.

Instructors can use the instructor portal to create one or more courses and deploy the course portal in our cluster. For each course, the instructors should specify the lab environment available to the learners by selecting among the lab environment templates (created by the administrators) and customize the course options, such as the tutorial page and the default storage quota of the students. After course creation, instructors can import their learners from a Comma-Separated Values (CSV) file. Instructors can also store learning materials like sample codes and datasets in NFS. These files will be synchronized to learners' workspace during the lab environment deployment process.

Learners can access our VLab platform by using browsers. Learners should first login and Keycloak will be used for user authentication. After successful login, learners will be redirected to the course portal and they may select the lab environment to be provisioned. Once a lab environment is selected, a pod will be spawned in our Kubernetes cluster to initialize the lab environment and the learners will be presented with a web-based editor or desktop graphical interface with an optional tutorial page to complete the learning tasks or assignments.

C. Development Issues and Challenges

To develop the production environment of our VLP, we first set up a K3s Kubernetes cluster deployed and deployed the VlabController, instructor portal, and the various cloud-native technology such as Ceph for persistent storage management, Harbor for local caching of lab images, Keycloak for authentication/authorization, and Prometheus/Grafana for lab environment monitoring.

It is a common scenario to run multiple applications within a single lab environment. For example, a database lab environment may be running an OpenVSCode Server, a database client pgAdmin and a database backend PostgreSQL. We can build a single Docker image that starts the various applications in a single container. However, running multiple applications in a single container may violate the single responsibility principle. We may need to rebuild the whole image if there is an update on any of the applications. This increases the complexity of the container images and the maintenance effort. To address the problems of the single-container approach, we have extended ShinyProxy to leverage the Kubernetes's capability to run multiple containers with a pod to compose multiple container images to create a lab environment. Different applications are launched from their own docker images and run in their corresponding containers. On the other hand, we create a Docker lab environment that allows students to run docker within the provisioned lab environment. This environment is developed based on a feature called "Docker-in-Docker (DinD)" which allows users to run Docker inside Docker containers. DinD requires us to enable the "privileged" mode to allow the Docker engine running inside the container to access the devices on the host [23], such as hard drives and network adapters. However, running a container in privileged mode may pose security risks, where the malicious users may gain root access to the host [24]. With our multi-container feature, we can run the Docker backend and user workspace in separate containers. The users are only allowed to access the non-privileged containers for improving the system security.

During the initial deployment of the platform, we observed a high latency for lab provisioning and high disk utilization of the disk when a large number of users are starting the lab at the same time. For many container images for databases (such as MySQL and Oracle), the database is initialized when the container is first launched and the process is disk-write intensive. When many users try to start the database containers at the same time, the file I/O will be very intensive, and this will result in disk overloading and errors in the cluster. To address the issues, we added some high-performance Solid-State Drives (SSDs) to our physical servers. Also, we have developed custom database images with pre-initialized databases to enhance the user experience by reducing the container startup time.

IV. Platform Evaluation

We invited undergraduate and postgraduate students from the Hong Kong Polytechnic University to participate in our pilot study. The students were enrolled in courses related to business analytics, database mining, database systems, and software engineering in the Faculty of Engineering and School of Business in the Spring semester of the academic year 2021–22. The lectures and lab sessions were mostly conducted online due to COVID-19. We prepared lab environments with software such as OpenVSCode, Docker, JupyterLab, Wekas and Orange Data mining tools. The students completed the hands-on labs, assignments and projects in our Virtual Lab Platform (VLP). Fig. 3 shows a deployed lab environment running the Orange data mining application inside our Kubernetes cluster.

At the start of the semester, students in the various courses were provided with an overview of the virtual lab platform. In subsequent weeks, students would use the virtual lab platform to complete the lab exercises. For some courses, students will also use the platform to complete their assignments and projects. During the semester, 370 users logged in to our lab platform and started 5523 lab sessions. The average time for provisioning the lab environments was 11.9 seconds. The peak workload mainly occurred during online lab sessions, where 97 students were running 148 lab sessions

simultaneously in our virtual lab platform. On average, there were 39.2 people logged in to the platform each day. Each user spent an average of 4.8 hours on the platform during the semester to complete the lab activities, assignments and projects. The most active user launched 165 lab sessions on the platform and spent 84.1 hours in the lab environment.

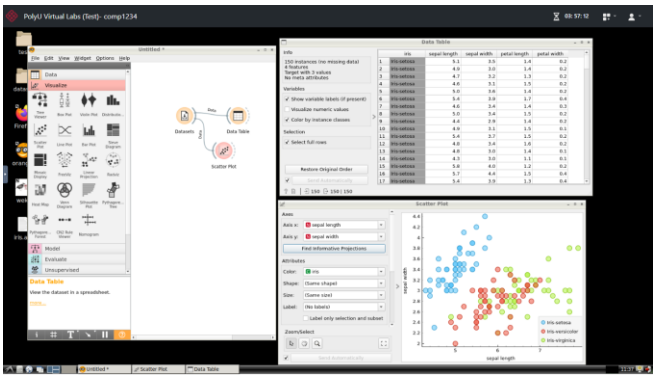

Fig. 3. A screenshot showing the deployed lab environment.

Towards the end of the semester, students were invited to participate in a survey to measure their expected perceived ease of use, perceived usefulness and intention to use our VLP. Eighty students (34 undergraduate and 56 postgraduate students) responded to our survey. The survey questions (constructed with a 7-point Likert scale) and the results are summarized in Table 2. For each construct, we performed a two-tailed t-test to compare the mean scores of undergraduate and postgraduate students. The results are summarized in Table 3. Overall, the perceived usefulness and intention to use the platform for postgraduate students were higher than that of the undergraduate students.

Table 2. Survey questions and results

Construct	Survey Questions	Mean Score (out of 7)	Standard Deviation
Perceived Ease of Use	My interaction with VLP interactive lab environment is clear and understandable.	5.48	1.16
	Interacting with VLP environment does not require a lot of my mental effort.	5.19	1.23
	I find VLP to be easy to use.	5.34	1.21
	I find it easy to get VLP to do what I want it to do.	5.39	1.13
Perceived Usefulness	Using VLP improves my performance in my course.	5.41	1.15
	Using VLP in my learning improves my productivity.	5.28	1.22
	Using VLP enhances my effectiveness in my learning.	5.45	1.07
	I find VLP to be useful in my learning.	5.58	1.08
Intention to Use	I plan to use VLP after the course.	5.29	1.29
	I intend to continue using VLP after completion of the course.	5.41	1.14
	I expect my use of VLP in the future after the course.	5.46	1.18
	Assuming I had access to VLP, I intend to use it.	5.46	1.15
	Given that I had access to VLP, I predict that I would use it.	5.45	1.17

Table 3. Comparison of the survey results for undergraduate and postgraduate students

	Undergraduate		Postgraduate		Significance (2-tailed)
Construct	Mean	Standard Deviation	Mean	Standard Deviation	
Perceived Ease of Use (EU)	5.32	1.33	5.41	1.32	0.11
Perceived Usefulness (PU)	5.21	1.11	5.62	1.12	0.021**
Intention to Use (IU)	5.25	1.09	5.61	1.08	0.0255**

**$p < 0.05$.

We also conducted focus groups to collect the students' qualitative feedback regarding their learning experience of our VLP. Most of the students perceived that the platform is easy to use, useful for learning online and intended to continue to use the system after the course. From the students' qualitative feedback, most students found the platform easy to use and greatly enhanced their online learning experience. By using web browsers to complete the labs, assignments and projects, the platform provides a consistent environment to complete the lab online across different operating systems. They could concentrate on learning the knowledge in the course without paying attention to how to install the application and set up the lab environment. In our initial platform design, the maximum duration of each lab session is 4 hours. Some students commented that some lab tasks (e.g., training deep learning models) may take more time to complete. In response to the students' comments, we have extended the maximum lab session duration to 8 hours.

V. Conclusion

This paper proposes a secure and scalable Kubernetes-based virtual lab platform for interactive skill-based learning. We have illustrated how container technology and open-source cloud-native applications can be used to develop and operate the virtual lab platform. Students may engage in self-paced learning, perform coding and conduct lab-based training at any time without installing any software in the local environment. The issues and challenges during the development and operation of the virtual lab platforms are also discussed. In our current infrastructure, a single NFS server is deployed for sharing resources and files between instructors and learners. We can't keep the storage service running during maintenance and it does not support high availability. For future research, we will develop a highly available NFS with a Ceph plugin called NFS-Genesha. The Ceph-based NFS may increase the stability, availability and scalability of our platform. On the other hand, our lab environments are based on Docker containers and they do not support some Linux kernel features to allow students to explore the operating system architecture. The adoption of Kata Container, based on Linux KVM technology, can help us build lab environments of more variety for courses related to operating systems and system programming.

Conflict of Interest

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Richard Lui was the leader of the project and he led the system development, evaluation and paper writing; Aiden Zhang developed the learning platform and wrote the technical implementation of the system; Philip lee perform literature review; All authors had approved the final version.

REFERENCES

[1] M. Ponticiello, M. Simmons, and J.-S. Lee, "The effects of the sudden switch to remote learning due to COVID-19 on HBCU students and faculty," *Learning and Collaboration Technologies: New Challenges and Learning Experiences*, Springer International Publishing, pp. 488–506, 2021.

[2] K. Kusumaningtyas, E. D. Nugroho, and A. Priadana, "Online Integrated Development Environment (IDE) in supporting computer programming learning process during COVID-19 pandemic: A comparative analysis," *IJID (International Journal on Informatics for Development)*, vol. 9, no. 2, pp. 66–71, 2020.

[3] T. Hinrichs, H. Burau, J. H. von Pilgrim, and A. Schmolitzky, "A scaleable online programming platform for software engineering education," *Workshop on Software Engineering for e-Learning Systems 2021*, vol. 2814, 2021.

[4] G. Chen, "CvLabs: A container based interactive virtual lab for IT education," Georgia Institute of Technology, 2020.

[5] P. Li, "Developing information technology labs on google cloud platform," *2020 ASEE Virtual Annual Conference Content Access Proceedings*, 2020.

[6] C. Geigle, I. Lourentzou, H. Sundaram, and C. Zhai, "CLaDS: A cloud-based virtual lab for the delivery of scalable hands-on assignments for practical data science education," in *Proc. Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE, pp. 176–181, 2018.

[7] D. R. Cruz and D. M. M. Mendoza, "Design and development of virtual laboratory: a solution to the problem of laboratory setup and management of pneumatic courses in Bulacan State University College of Engineering," in *Proc. 2018 IEEE Games, Entertainment, Media Conference (GEM)*, pp. 1–23, 2018.

[8] D. Joyner and C. Isbell, "Master's at scale," in *Proc. the 6th 2019 ACM Conference on Learning at Scale, L@S 2019*, pp. 1–10, 2019.

[9] A. Green and C. Zhai, "LiveDataLab," in *Proc. the 6th 2019 ACM Conference on Learning at Scale, L@S 2019*, pp. 1–2, 2019.

[10] M. Cardoso, R. Barroso, A. V. Castro, and Á. Rocha, "Virtual programming labs in the computer programming learning process, preparing a case study," *EDULEARN17 Proceedings*, pp. 7146–7155, 2020.

[11] X. Hu, H. Le, Y. Long, A. Bourgeois, and Y. Pan, "Board 88: Support remote collaboration in virtual computer labs," *2019 ASEE Annual Conference & Exposition Proceedings*, 2019.

[12] X. Hu, H. Le, A. G. Bourgeois, and Y. Pan, "Collaborative learning in cloud-based virtual computer labs," in *Proc. Frontiers in Education Conference, FIE*, vol. 2018, 2019.

[13] M. Iorio, A. Palesandro, and F. Risso, "CrownLabs—a collaborative environment to deliver remote computing laboratories," *IEEE Access*, vol. 8, pp. 126428–126442, 2020.

[14] F. Trinta, P. A. L. Rego, and W. Viana, "Teaching development of distributed software during COVID-19: An experience report in Brazil," *ACM International Conference Proceeding Series*, pp. 616–625, 2020.

[15] R. J. Brunner and E. J. Kim, "Teaching data science," *Procedia Computer Science*, vol. 80, pp. 1947–1956, 2016.

[16] J. Doi, G. Potter, J. Wong, I. Alcaraz, and P. Chi, "Web application teaching tools for statistics using R and shiny," *Technology Innovations in Statistics Education*, vol. 9, no. 1, 2016.

[17] J. Stander and L. Valle, "On enthusing students about big data and social media visualization and analysis using R, RStudio, and RMarkdown," *Journal of statistics education*, vol. 25, no. 2, pp. 60–67, 2017.

[18] Y. Li, "Towards fast prototyping of cloud-based environmental decision support systems for environmental scientists using R Shiny and Docker," *Environmental modelling & software : with environment data news*, vol. 132, p. 104797, 2020.

[19] B. Anbaroğlu, "A collaborative GIS programming course using GitHub Classroom," *Transactions in GIS*, vol. 25, no. 6, pp. 3132–3158, 2021.

[20] A. M. Potdar, N. D G, S. Kengond, and M. M. Mulla, "Performance Evaluation of Docker Container and Virtual Machine," *Procedia Computer Science*, vol. 171, pp. 1419–1428, 2020.

[21] G. Sayfan, *Mastering Kubernetes*, Packt Publishing, 2018, ch. 1, pp. 7–17.

[22] S. Weil, S. Brandt, E. Miller, D. Long, and C. Maltzahn, "Ceph," in *Proc. the 7th Symposium on Operating Systems Design and Implementation*, 2006, pp. 307–320.

[23] T. Bui, "Analysis of docker security," arXiv, 2015.

[24] T. Combe, A. Martin, and R. D. Pietro, "To docker or not to docker: A security perspective," *IEEE cloud computing*, vol. 3, no. 5, pp. 54–62, 2016.

Copyright © 2024 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (CC BY 4.0).