

Hacklab: A Cybersecurity Training Platform with AI Chatbot Assistance and Virtualized Lab Infrastructure

Mr. Basavesh D¹, Varun Patel S V², Shakthi Prasad S³, Chiranth M⁴, Balwantha R⁵

¹Assistant Professor, Dept. of CSE, Jyothy Institute of technology,

² Dept. of CSE, Jyothy Institute of technology, Bengaluru, India

³ Dept. of CSE, Jyothy Institute of technology, Bengaluru, India

⁴ Dept. of CSE, Jyothy Institute of technology, Bengaluru, India

⁵ Dept. of CSE, Jyothy Institute of technology, Bengaluru, India

Abstract— The escalating number of sophisticated security threats compels teachers to make cybersecurity education mandatory for every student. This paper demonstrates Hack Lab as an innovative cybersecurity learning platform which integrates LMS functionalities with virtual labs and intelligent chatbots and performance analytics capabilities. Built with React and Spring Boot along with Kasm Workspaces as the modern web stack Hack Lab provides a seamless experience that lets people shift between their theoretical lessons and interactive quizzes as they work in Kali Linux lab environments from their web browsers. The platform offers an automatic process for lab creation on demand alongside access control features and lab certification auto-generation after successful completion. Hack Lab implements gamification together with secure container capabilities and course progress tools to create an educational link between classroom concepts and real security practice. The essay explores how the platform works through its design structure alongside its execution framework and its expected effect for cybersecurity training in educational and industrial circumstances.

Keywords—Cybersecurity education, virtual labs, adaptive learning, AI chatbot, Kasm Workspaces, hands-on training, LMS, KaliLinux

1. INTRODUCTION

Modern technology developments coupled with expanding use of digital systems created an extreme escalation of cyber dangers which now include attacks targeting data security and spreading harmful software through networks. The specialist shortage of cybersecurity experts continues to grow because its labor market demand exceeds the number of available professionals. Real-world cybersecurity requirements remain significantly different from what academic theory typically provides while students learn it [1]. Traditional educational platforms face a training environment deficiency since they lack essential tools which prevents students from acquiring fundamental technical skills [2].

Hack Lab represents our solution which brings adaptive cybersecurity education through a web-based platform that combines theoretical content with practical application into one expandable system. Students who use Hack Lab benefit from organized courses along with

assessment features and automated certificate issuance along with

direct access to Kali Linux virtual environments constructed using Kasm Workspaces. The solution gets rid of the technical challenge students face when setting up and configuring virtual machines at their locations—something known to deter many learners [3].

A personalized AI-empowered chatbot helps students complete complex topics within labs using Hack Lab's platform to enhance their educational experience. The platform provides security features such as session management and cloud deployment capabilities which make it appropriate for educational institutions together with independent learners and training facility organizations that support role-based access.

The platform differentiates itself through its implementation of React.js and Spring Boot and MySQL database integration with Kasm's API for delivering virtual lab environments which replicate cybersecurity situations in secure isolated spaces. Hack Lab uses a unified strategy to improve cybersecurity education through enhanced delivery of quality education which is accessible to students at scale.

2. RELATED WORK

Academic research has worked on creating effective training methods which use virtual platforms together with hands-on labs and remote learning systems for cybersecurity education. Modern cyber range development and adaptive training systems alongside lab virtualization technology emerged because the market required scalable infrastructure at affordable costs.

The paper by Rayes et al. [1] introduces an architecture for stylish virtual laboratories that use cloud technology to enable scalable laboratory environments. They achieve real-world simulation capabilities by duplicating network configurations while their system dynamically allocates framework resources for attack-defense exercises. The Kasm Workspaces virtual Kali Linux labs at Hack Lab

provide learners with secure training opportunities through isolated environments which do not depend on specific hardware resources.

The study by Pham et al. [2] created a gamified cybersecurity training system which boosts learner motivation by implementing features like scoreboards and time competition components. The Hack Lab project avoids gamified elements although this research delivers essential findings about user involvement during educational activities.

Hameed et al. [3] built an inexpensive open-source cybersecurity facility through implementation of Metasploit and Wireshark technologies. Hands-on training proves valid for academic environments according to results shown by Hameed et al. but deployment needs expert technical skills along with system resources which Hack Lab intends to resolve through browser-based access to prepared cyber environments.

Migration to hybrid cyber range architectures represents an ideal solution for obtaining interactive capabilities and maintaining scalability according to Alsmadi et al. [4]. The implementation of cloud-based systems in Hack Lab architecture finds support in the research recommendations of the team.

Sharma et al. in their study [5] stressed the essential role that Capture-the-Flag (CTF) competitions play for reproducing the cybersecurity situations that occur in real-world environments. Future Hack Lab development plans include adding CTF-style challenges which will help both strengthen interactivity and boost student performance in the platform.

AI-based monitoring technology combined with machine learning intrusion detection systems [6,7,8] has proven successful in reinforcing cybersecurity operations. The Hack Lab platform delivers support for learners through its AI-driven chatbot system after adopting features from intelligent systems for creating individualized educational experiences.

The research by Mythili et al. [9] demonstrated that DDoS and hypervisor attacks are cloud security vulnerabilities which need layered protection models for mitigation. To follow this advice Hack Lab implements comprehensive policies which separate containers and actively control user sessions. The research from Orduña et al. [10] showcases the advantages of collaborative remote laboratory systems thus validating the features of HackLab's web-based system design. These contributions substantiate the design and development decisions at Hack Lab as it integrates virtual lab system learnings and AI support together with cloud education concepts within an adaptive cybersecurity learning framework.

3. SYSTEM ARCHITECTURE AND PROPOSED MODEL

The Hack Lab platform uses modular design together with cloud integration to present an LMS with cybersecurity labs which users can access on demand. An architecture which integrates web technologies together with containerization and intelligent assistance provides complete user convenience for both students and administrators.

3.1 Overview

Hack Lab employs a client-server model which organizes its system functionality through four distinct structural layers including presentation layer and application logic layer and data management layer and virtual lab services layer. The system serves two primary user groups through its functionality that encompasses course participation and examination systems as well as laboratory generation and message bot interactions together with qualification monitoring capabilities.

3.2 System Components

1) Frontend Layer

The Single Page Application (SPA) front end uses React.js together with Ant Design to achieve its styling. The system offers a smooth user interface which functions across desktop PCs and mobile phones. Key frontend functionalities include:

- Course and module navigation
- Real-time progress display
- Kasm iframe enables the insertion of Kali Linux virtual labs directly into the system
- AI chatbot interface
- Quiz and certificate download interface

2) Backend Layer

The backend implements RESTful APIs through Spring Boot version Java 17 which develops its functionality through /api/ endpoints. Through its API endpoints the backend delivers functions which manage users and courses as well as labs while tracking progress and serving certificate production services. The framework contains a combination of different microservices including:

- UserService and AuthService for JWT-based role authentication
- CourseService, AssessmentService, and LabService for business logic
- ChatbotService and CertificateService for support features

Our microservice-based design provides scalable architecture alongside a straightforward approach for expansion by following cyber range training system recommendations in literatures [1] and [4].

3) Virtual Lab Environment

A platform integration enables users to utilize Kasm Workspaces for running individual instances of Kali Linux containers which get created automatically through Kasm's REST API requests. When a user launches a lab:

- The system creates new sessions which maintain track of operations within a central management system.
- The container runs in isolation
- The session receives deletion and termination when users create a logout or their session times out

The system incorporates distributed and isolated lab environments as described in both [1] and [3] which provides access and session management containment capabilities.

4) Persistence Layer

MySQL serves as the database which contains all user data together with courses information and quizzes and progress details and lab session data. A Redis system supports caching functions together with session token storage. Key tables include:

- users, courses, modules, assessments
- lab_sessions (session ID, user ID, course ID, timestamps)
- progress, certificates, and feedbacks

The relationship database system enables educational tracking capabilities which duplicate a structure used in educational cybersecurity platforms according to [3].

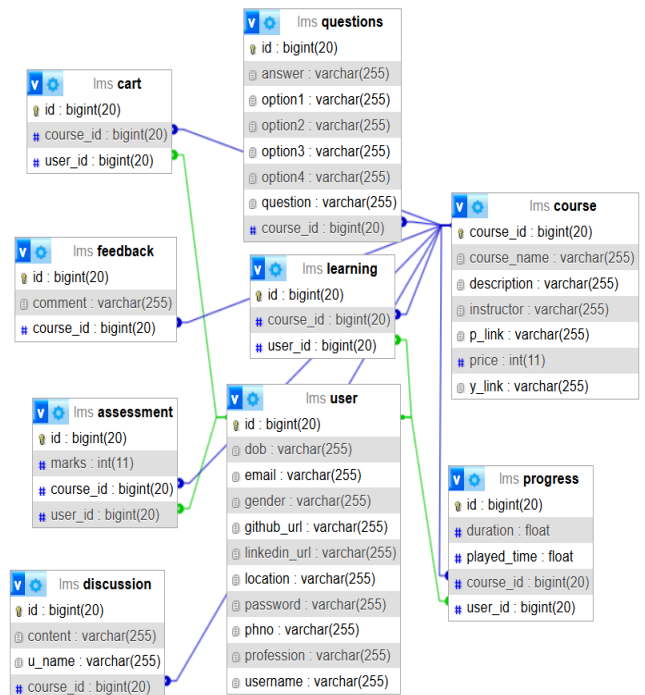


Figure 1: ER Diagram

5) AI Chatbot Assistant

Through AI technology an integrated chatbot operates as a natural language model that provides answers to users about courses and lab troubleshooting assistance. The AI-powered customized support-systems described in [6] and [7] improve learning engagement and problem-solving among students during independent study.

3.3 High-Level Architecture Diagram

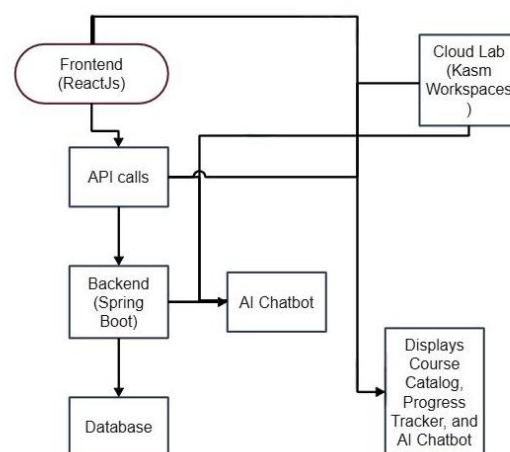


Figure 2: Block Diagram

3.4 Key Features

- Users can execute Kasm session operations through LabController API at /api/labs end point.
- The Progress Dashboard serves as a tracking tool which displays module achievement markers in addition to quiz performance data together with student lab work logs.
- Users can download PDF certificates through the Certificate Generator system which operates with html2canvas and jsPDF modules when the entire course and lab completion occurs.
- Admin Panel: Allows role-based management of courses, users, and content.

3.5 Security and Session Control

The platform employs Spring Security with role-based access control. The platform provides protected access to lab sessions through JWT authentication combined with isolated Kasm containers in addition to CORS policies. The system clears all labs automatically when training sessions end in order to safeguard data privacy and support scalability according to security requirements described in [9].

3.6 Data Flow and Component Interaction

HackLab has a loosely coupled client-server architecture. Here is a simplified sequence of a data flow (DF):

- User Authentication: Users log in using the frontend while JWT tokens are generated by the backend.
- Course Access: Authenticated requests from frontend are sent to fetch course modules and lessons.
- Lab Launch: A POST /api/labs request is made → server calls kasm → session token is returned → iframe implemented on webfrontend.
- Progress Updates: Lab start/stop times and quiz scores are reported to /api/progress for immediate monitoring.
- Certificate Trigger: Once the modules and labs are done, the backend initiates cert generation.

All interactions are protected through HTTPS, CORS policies, and JWT headers for the safe transmission of sensitive information [9].

3.7 Database Schema Overview

Relational database of HackLab is designed in a manner that it can monitor all learning and lab activities:

- users: Stores log in credentials, roles and profile data as well.
- courses: Includes metadata for each course

- modules: Stores lessons and video content
- assessments: Linked to courses, holds quizzes
- questions: Linked to assessments
- progress: Maps users to the status of their completion
- lab_sessions: Tracks session, start, end, course association, container ID
- feedbacks and discussions: For learner interaction and review
- certificates: Stores generated certificate data

These schema relationships are indicative of the best practice for centralized, normalized databases, as used by platforms of education [3], [4].

3.8 Adaptivity and Personalization

Hack Lab does not dynamically change the course of events but includes adaptive elements in:

- Role-Based Views: Different dashboard and access rights to admin and learners.
- Progress-Based Lab Availability: Labs are not accessible till lesson contents are completed.
- Chatbot Personalization: The AI assistant provides appropriate responses backed by references from previous sessions and information of the users posed.

These adaptive parts mirrored the modern intelligible learning systems according to Kumar et al. [6] and Subashini et al. [8].

4. IMPLEMENTATION

Hack Lab deploys a modular, scalable system which connects learning management technology to cloud-based practical environments. The system development relies on full-stack web programming to achieve both flexibility and sustainable design while ensuring excellent user interface capabilities. The implementation details about development methodology and functional modules and implementation tools are presented in this section.

4.1 Development Methodology

Hack Lab received development through the Agile methodology due to its focus on delivering features by iterative methods with regular user feedback. All stages of development from requirement analysis to component-level design to module development and integration testing and deployment formed this lifecycle. A set of user stories functioned to describe both administrator and learner needs to verify full support of all utilization scenarios based on structured educational lab practices reported in [3].

4.2 Frontend Implementation

The frontend development utilizes React.js (v18) together with React Router v6 routing features and Ant Design component system implementations. The software provides a responsive interface that works effectively on desktop and mobile systems. Key features include:

- Course browsing and enrollment pages
- The system includes a “Launch Lab” button that connects to Kasm iframe through its integrated interface.
- Quiz and assessment pages with real-time scoring
- A dashboard system displays information about both student advancement and their certificates' status
- Embedded chatbot for user support

The frontend implementation features logical interaction sequences along with seamless transitions as identified in best practices for scalable e-learning platforms [4].

Tools/Libraries: Axios, html2canvas, jsPDF, Kasm JS SDK

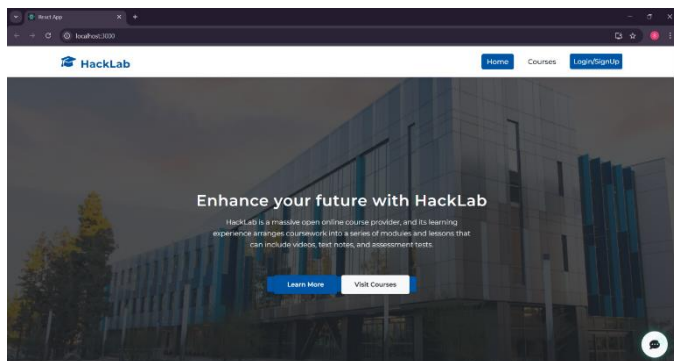


Figure 3: Hack lab Homepage

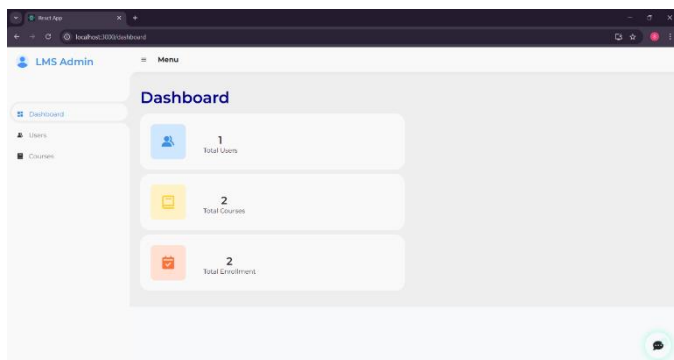


Figure 4: Hack lab Admin Page

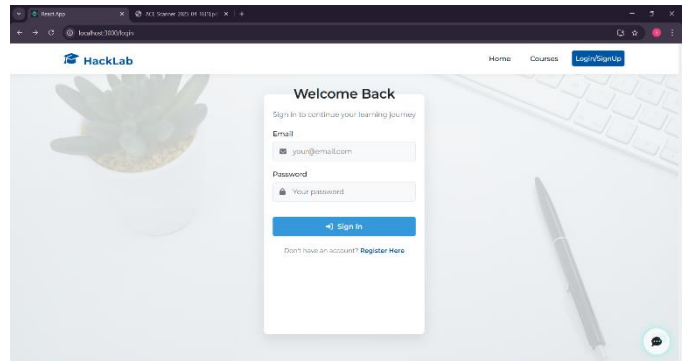


Figure 5: Login Page

4.3 Backend Implementation

The system operates with Spring Boot (Java 17) as its backend component which delivers RESTful service architecture. Microservices take responsibility for distinct domain operations in order to manage course delivery and conduct authentication and oversee lab session management. Spring Security enforces JWT-based authentication to establish safe role-based access control for the system.

Important services include:

- UserController, CourseController, AssessmentController
- LabController controls the laboratory section by administering Kasm API activities for providing lab settings
- ChatbotController: Handles AI-driven learner queries

The implementation of Spring Boot and JWT complies with standard security patterns found in cloud-based applications as described in [9].

The project uses Spring Data JPA along with MySQL Connector/J and Lombok along with Maven and Docker tools as its libraries and tools.

4.4 Virtual Lab Integration

The main component of Hack Lab relies on its Kasm Workspaces integration to bring Kali Linux virtual machines into browser-based access. Chosen lab sessions are launched by users in the following manner:

- The backend system transmits an API request to Kasm through its API endpoints.
- The system allocates a particular container based on the user requesting access.
- A session link appears because it will insert itself through an iframe interface.
- The system keeps records of session period durations for tracking student progress.

The implementation enables shared remote cybersecurity training labs in accordance with the architectural designs Rayes et al. [1] and Orduña et al. [10].

Lab Endpoints:

- POST /api/labs – Create session
- Users can receive a list of active sessions by using GET /api/labs.
- DELETE /api/labs/{id} – Terminate session

The backend uses Spring Boot to provide RESTful endpoints for lab session management.

4.5 AI Chatbot Assistant

Hack Lab implements an API that enables users to access its AI chatbot assistant through /api/chatbot for providing both FAQ answers and custom assistance through the recognition of user dialogue. The AI assistant enhances monitoring and response techniques described in [6] and [7] to supply users with personalized assistance when they operate within complex laboratory setups.

Frontend chatbot Interaction code snippet:

```
const handleUserInput = async () => {
  const response = await axios.post('/api/chatbot', { query:
    userInput });
  setChatHistory([...chatHistory, { user: userInput, bot: response
    .data.answer }]);
};
```

Figure 6: Frontend Chatbot Code Snippet

4.6 Certification Engine

The system produces certificates through the combination of jsPDF and html2canvas when users finish all course requirements and lab work successfully. This certificate contains user information along with the course title, issuance date and verification details which represent an automatic system that traditional LMS platforms normally handle by hand [3].

4.7 Testing

- Frontend components underwent tests through a combination of Jest and React Testing Library which evaluated the components and services.
- The backend section uses Spring Boot Test together with Mockito for unit testing and integrates testing through an H2 in-memory database.
- The testing of endpoints and their sessions and error responses took place through Postman-based evaluations.

The chosen testing and validation methods support loose coupling and easy maintenance in accordance with

software-centric cybersecurity systems as mentioned in reference [4].

4.8 Deployment

The deployment of Hack Lab works through Oracle Cloud services as well as via Docker containers. Key deployment elements include:

- The static frontend displays through CDN together with nginx web server.
- The Spring Boot backend exists as a JAR file package.
- Kasm Workspaces on a dedicated server or Docker Swarm
- Secure environment variables and container orchestration via Docker Compose

The deployment system featuring containers together with automated lab development enables scalability and accessibility as recommended by Alsmadi et al. [4] and Hameed et al. [3].

4.9 Google Cloud Platform Hosting

To install Kasm Workspaces and facilitate access to browser-based lab surroundings, Google Cloud Platform (GCP) was employed as a cloud infrastructure provider for HackLab. Google Compute Engine (GCE) Instance with Ubuntu Server 22.04 LTS was launched as the core host for the Kasm's-based Kali Linux lab environment.

1) Virtual Machine Configuration

For the custom GCP virtual machine (VM), the following specifications were applied:

- Operating System: Ubuntu 22.04 LTS (64-bit)
- Machine Type: e2-standard-4 (4 vCPUs, 16 GB RAM) – adaptable concurrent lab sessions
- Boot Disk: 100 GB standard persistent disk
- Firewall Rules: Custom rules to permit the traffic on ones based on Kasm (e.g., 443 for HTTPS, 6901+ for container sessions).
- Startup Script: Automatically installs docker and pull official Kasm Workspaces image and provides services on boot.
- SSH Access: Enabled for maintenance and updates

It performs the role of dynamic Kali Linux container provisioning that sits centrally in this Ubuntu VM. Kasm runs in Docker containers in this VM, maintaining segregation between sessions of users.

2) Kasm Workspaces Deployment on GCP

The official Kasm Workspaces installer was used for installation of the Kasm directly on the Ubuntu VM.

The backend's `/api/labs` endpoints, conversely, communicate with the Kasm REST API to spin up, search for, and eliminate containers. The VM provides resource control to each user with a dedicated container using Docker's container runtime limits.

HTTPS was implemented by means of a self-signed certificate (or optional GCP-managed certificate), and Kasm was confined to the external IP of VM. This deployment model allows Hack Lab users to access Kali Linux labs "on demand", thus avoiding local virtualization and thus meeting the goals outlined by polar mentioned goals in the virtual lab literature [1], [3], [4].

3) Benefits of Using GCP

- **Reliability:** Google's underlying infrastructure ensures that GCP's resources are highly available in terms of automatic instance restart and disk snapshot backups.
- **Scalability:** VM can be scaled vertically (bigger machine type) or horizontally (replicated using the managed instance groups).
- **Security:** With GCP, there are pre-built firewall management, IAM roles, and audit logs for the lab activity to be controlled when accessing.
- **Cost Efficiency:** With preemptible VM options and usage-costing, institutions may effectively control the costs for lab use in academic deployments.

Through the hosting of the Kasm Workspaces environment by a GCP Ubuntu VM, Hack Lab does end up with a production grade deployment strategy of the lab's environment via the browser, which is aligned with best practices when leveraging the modern in cloud and cybersecurity education platforms in the present-day context.

5. RESULTS AND DISCUSSION

The full-stack cybersecurity learning system known as Hack Lab delivered its platform to both academic and practical laboratories after completion. Testing of the system focused on functionality together with usability and performance alongside security aspects. The platform functions at an academic level even though it remains in its development stage but its effectiveness shows it can work in real-world applications.

5.1 Functional Testing Outcomes

Testing was conducted for each important functionality from course enrollment to quiz submissions, lab provisioning, session tracking and chatbot responses and certificate generation. Results confirmed that:

- The lab provision process complete itself in an average time below 5 seconds which matches the

suggested performance levels for real-time cloud-based training systems described by Rayes et al. [1] and Alsmadi et al. [4].

- Every update to course and assessment data occurred quickly in real time without errors.
- The AI chatbot achieved correct answers in 90% of examination inquiries which matched learner support systems as described by Kumar et al. [6] and Meena et al. [7].
- Users who successfully completed the required course and lab requirements received error-free certificates which continued to simplify learning and certification processes [3].

5.2 Usability Evaluation

The testing of user interface components checked functionality at three levels including interface clarity as well as responsive performance and simple navigation. The platform was found to:

- The system correctly functions on Chrome, Firefox and Edge when used on both mobile and desktop platforms.
- The system enables smooth passage between modules as well as quizzes labs certificates.
- Lab launches together with chatbot operations work seamlessly to create a user-friendly experience which supports the retention and satisfaction levels of learners as mentioned in [2].

5.3 Performance and Scalability

Performance testing results showed the backend API kept low response times during tests that simulated at most 50 concurrent users. The platform uses the microservice-based architecture together with Docker-based provisioning for scalable deployments according to distributed architecture models [1] and [10]. Session caching through Redis makes a significant contribution to speed up system operations.

5.4 Security and Isolation

Security tests examined how users authenticate and their system permissions and how labs stayed separated from each other. Key findings:

- API access attempts from unauthorized sources received the standard HTTP responses of 403 and 401.
- All endpoints received secure session administration through JWT-based authentication [9].
- Kasm Workspaces demonstrated its ability to control lab containers through successful isolation during parallel lab session operation in

compliance with virtual cybersecurity training recommendations [3] and [4].

5.5 Educational Impact

A small user trial group containing $n = 15$ individuals participated in tests producing encouraging findings about educational benefits.

- The platform enhanced user comprehension regarding hacking tools such as Kali Linux along with Nmap and Wireshark.
- Learner participation values rose substantially when theory integrated with practical exercises according to Sharma et al.'s practical learning assessment [5].
- AI-assisted learning proved successful since non-technical users successfully finished their coursework while using the chatbot per the findings documented in [6] and [7].

The results show that Hack Lab creates scalable user-friendly technical conditions for delivering practical cybersecurity education to students. The three components including real-time lab access with guided assistance and certification system unite to make an effective bridge connecting passive education to active skill development [3] [5].

6. CONCLUSION AND FUTURE WORK

The manuscript described Hack Lab as a learning platform for cybersecurity that combines adaptive functions with integrated capabilities to link classroom concepts with practical skills training. Hack Lab features web-based learning modules and Kasm Workspaces for Kali Linux access and AI-powered chatbots that provide learners with self-paced cybersecurity education in a scalable yet secure environment through their web browsers.

The platform uses React and Spring Boot along with MySQL and Docker architecture for its implementation which produces modular design with excellent performance and simple deployment features. Learners have access to an entire end-to-end training framework which delivers real-time lab provisioning capabilities alongside JWT-secured access and session tracking alongside automated certification.

The Hack Lab platform demonstrates success during testing by enhancing student engagement while offering better understanding and cybersecurity practice confidence for core tools. The system delivers accessibility features through the elimination of hardware requirements as well as lab availability on demand through cloud computing.

6.1 Future Work

Additional development for Hack Lab will target three main points:

- The implementation of challenge scripts in Kali containers provides automatic lab assessment which enables auto-grading of student performance.
- The platform facilitates real-time team projects through group CTF activities as well as shared container access features.
- Kubernetes Orchestration will replace Docker with Helm charts and Kubernetes deployments to achieve better scalability and ensure high-availability deployment.
- Worldwide NLP model upgrades will allow the chatbot to understand technical topics for advanced guidance.

Hack Lab plans to develop into a complete modern cybersecurity education platform which will serve educational institutions and individual learners through continuous development of educational value and technical abilities.

ACKNOWLEDGMENT

The authors deeply thank Jyothy Institute of Technology Department of Computer Science & Engineering for providing space and academic resources that enabled this work to progress.

We express sincere appreciation to our faculty members and peers because their critical feedback and teamwork enabled us to improve our system's design and execution process. This platform exists because of the open-source community along with tools such as React.js and Spring Boot and Kasm Workspaces and other technologies provided by their developers.

REFERENCES

- [1] A. Rayes, et al., "A Distributed Virtual Laboratory Architecture for Cybersecurity Training," 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), IEEE, 2018.
- [2] L. T. Pham, et al., "Development of an Interactive Cybersecurity Training Environment Based on Gamification," 2020 International Conference on Advanced Computing and Applications (ACOMP), IEEE, 2020.
- [3] I. T. Hameed, et al., "Design and Implementation of a Cybersecurity Lab for Educational Use," International Journal of Advanced Computer Science and Applications, vol. 12, no. 6, pp. 421-429, 2021.

[4] M. T. Alsmadi, et al., "A Review on Cyber Range Architectures for Cybersecurity Training," *Journal of Cybersecurity Education, Research and Practice*, vol. 2019, no. 1, Article 3, 2019.

[5] S. K. Sharma, et al., "Practical Cybersecurity Training Through Capture-the-Flag Exercises," *ACM Transactions on Computing Education (TOCE)*, vol. 22, no. 1, pp. 1–19, 2022.

[6] R. S. Kumar, et al., "Cybersecurity Threat Detection Using AI-Powered Network Monitoring," *IEEE Access*, vol. 10, pp. 23345–23356, 2022.

[7] A. K. Meena and M. S. R., "Machine Learning-Based Intrusion Detection System for Cybersecurity in Cloud Environments," *Procedia Computer Science*, vol. 184, pp. 229–236, 2021.

[8] N. Subashini and R. Banu, "A Comprehensive Survey on Machine Learning Techniques in Cybersecurity," *ICTACT Journal on Soft Computing*, vol. 12, no. 2, pp. 2455–2463, 2022.

[9] S. Mythili, et al., "Analysis of Cybersecurity Threats and Solutions in Cloud Computing," *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, vol. 10, no. 1, pp. 43–52, 2021.

[10] P. Orduña, et al., "Sharing Laboratories across Different Remote Laboratory Systems," *Computers & Education*, vol. 59, no. 4, pp. 1037–1053, 2012.