2020 CSLabs Extension Project

Software Requirements Specification Report

****

Submitted to:

Dr. Ronald B. Finkbine

Professor of Computer Science

Presented by:

Yiliang Lu, Junet Bello, and Copper Martin

October 6th, 2020

## Table of Contents

[1. Introduction](#_Toc52873031) 1

* [Purpose 2](#_Toc52873032)
* [definitions 2](#_Toc52873032)
* [system overview 2](#_Toc52873032)
* [reference 2](#_Toc52873032)

[2. project description 2](#_Toc52873032)

* [Product perspective 2](#_Toc52873032)
  + [System Interfaces 2](#_Toc52873032)
  + [user Interfaces 2](#_Toc52873032)
  + [hardware Interfaces 2](#_Toc52873032)
  + [software Interfaces 2](#_Toc52873032)
  + [communication Interfaces 2](#_Toc52873032)
  + [memory constraints 2](#_Toc52873032)
  + [operations 2](#_Toc52873032)
  + [site adaption requirements 2](#_Toc52873032)
* [Product functions 2](#_Toc52873032)
* [user characteristics 2](#_Toc52873032)
* [Constraints, assumptions and dependencies 2](#_Toc52873032)

[3. specific requirements 2](#_Toc52873033)

* [external interface requirements 2](#_Toc52873032)
* [functional requirements 2](#_Toc52873032)
* [performance requirements 2](#_Toc52873032)
* [design constraints 2](#_Toc52873032)
  + [standards compliance 2](#_Toc52873032)
  + [logical database requirements 2](#_Toc52873032)
  + [software system attribtes 2](#_Toc52873032)
  + [reliability 2](#_Toc52873032)
  + [availability 2](#_Toc52873032)
  + [security 2](#_Toc52873032)
  + [maintainability 2](#_Toc52873032)
  + [portability 2](#_Toc52873032)
  + [other requirements 2](#_Toc52873032)

[4. Key Personnel and Contribution Breakdown](#_Toc52873039) 5

[5. WORKS CITED](#_Toc52873039) 5

This page is intentionally left blank.

# **1. Introduction**

### 1.1 Purpose

CSLabs is a virtual lab learning environment created and operated by the Indiana University Southeast (IUS) Computer Security Group (CSG). It is used by IUS faculty and students to practice computer security and learn other aspects of computer science using virtual machines (VM). A functional CSLabs will serve the CSG’s mission of providing “…technical education and hands on experience…” to its members and educating “…the public about security awareness”.

The CSLabs 2020 Capstone project is an extension of the previous year’s Capstone projects. It is aimed to add new functionalities to the CSLabs Web-app project and expand the overall system capability. Team Cosmic Alpha (TCA) has taken the challenge of the CSLabs 2020 Capstone project.

### 1.2 Definitions

The following definitions were originally defined in previous SRS report by Gallavin et al. (2019). The terms have been updated by TCA, and the list is sorted alphabetically.

|  |  |
| --- | --- |
| Admin | An administrator user with super-user / root privileges to an information system |
| ASP .Net Core | A free and open-source web framework and successor to ASP.NET. It is a server-side web-application framework designed for web development to produce dynamic web pages. |
| Badge | An award given in CSLabs |
| Cluster | A set of loosely or tightly connected computers (nodes) that work together and often can be viewed as a single system. Clusters are usually deployed to improve performance and availability over that of a single computer. |
| Kestrel Web Server | An open-source, event-driven, and asynchronous I/O based server hosting ASP.NET applications. |
| Lab | A single VM or a collection of VMs networked together |
| Module | Package(s) providing comprehensive learning experience in CSLabs |
| NGINX | The forward facing linux web server that proxies connections to the application server |
| Node | A computer that is part of a cluster running its own instance of an operating system. |
| Proxmox Virtual Environment (VE) | An open-source server virtualization management platform based on the Debian Linux distribution |
| React | An open-source and front-end JavaScript library for building user interfaces or UI components maintained by Facebook |
| Staff User | An IU faculty/staff member with designer privileges that can create modules |
| Standard User | An individual with non-administrative or non-designer privileges in an information system |
| VM | A Virtual Machine |

### 1.3 System Overview

On the hardware side, according to Clifton et al. (2019), CSLabs runs on donated hardware that is clustered and spawns multiple virtual environments in the CSG datacenter. The datacenter is currently residing in IUS Natural Science building on a ten Giga-bit intranet and one Giga-bit internet connection. Internet connectivity is tunneled through IU’s campus network and utilizing a cloud hosted router (CHR) service. Although no clear SLA is defined at the moment, the target availability is 90% and the average system uptime has been near 99.97%.

On the software side, according to Gallavin et al. (2019), CSLabs has a web-facing frontend that is written in typescript using the React JavaScript library. CSLabs also has an application server backend that is written in C# with the REST API. The frontend communicates with the backend using the ASP .NET Core API. The backend will leverage the proxmox API to create, turn on, turn off, and destroy VMs and networks.

### 1.4 References

https://github.com/ius-csg/cslabs-backend

https://github.com/ius-csg/cslabs-webapp

https://reactjs.org/

https://www.proxmox.com/en/

# **2. Project Description**

CSLabs allows standard users to access the learning modules using a modern web browser. It provides a unified user interface without a hefty system requirement or a long list of software dependencies.

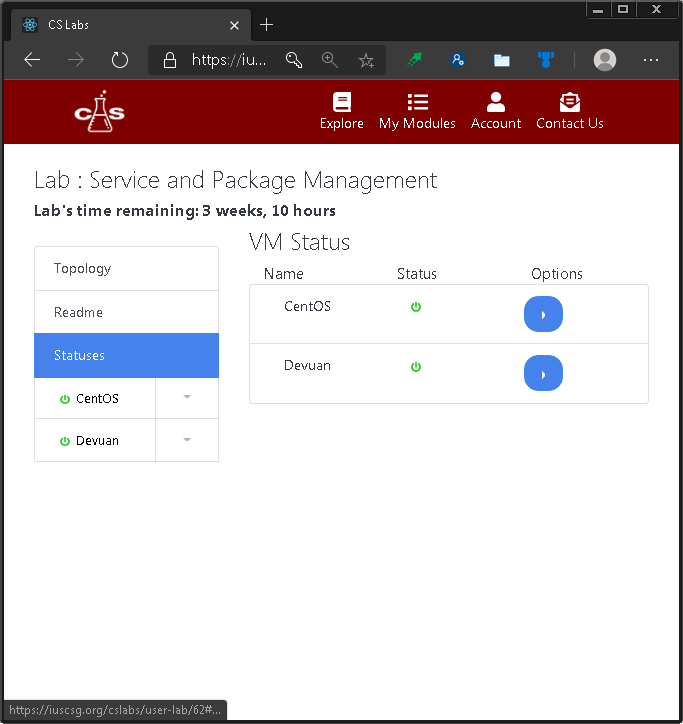
### 2.1 Product Perspective

#### a. System Interfaces

According to Gallavin et al. (2019), the backend HTTP API serve as the system interface for any staff member that wants to automate actions. It also serves as an interface for the user interface. The backend uses JSON to communicate data back and forth.

#### b. User Interfaces

According to Gallavin et al. (2019), the React Typescript front end will drive the interactive experience allowing users to navigate seamlessly to other pages and VMs.



#### c. Hardware interfaces

* Fast servers with multi-core processors and large RAM capacity.
* Networked storage with terabytes of data space
* 10 Giga-bit network switches and routers
* Backup infrastructure

.

#### d. Software interfaces

The following passages are paraphrased from the previous SRS report per Gallavin et al. (2019) due to TCA’s limited knowledge of the system.

* The FreeIPA Kerberos server is used for authentication providing a single password for all CSG applications.
* Proxmox is used to manage VMs.
* A Logstash server will be provided by CSG to provide analytics on errors and performance of the application.
* Rundeck will serve as an automation repository for managing VMs. This will help make communicating with Proxmox a lot easier.
* The backend will utilize C# for a strong typed experience.
* The backend will utilize .Net Core to run C# on Linux
* The Database (Maria DB) will be used to store the application’s entities.
* Typescript and React will serve as the front end software interface to create attractive user interface.

#### e. Communication Interfaces

CSLabs content is delivered using any supported modern browser using HTTPS. Email is used to facilitate account creation and service. Other types of communication may be delivered during CSG meetings and class lectures.

#### f. Memory Constraints

According to Gallavin et al. (2019), the Proxmox VE servers cluster will likely face memory constraints due to large memory requirements of running multiple instances of VMs discussed in a later section. CSG expects to support up to 30 simultaneous VM instances.

#### g. Operations

According to Gallavin et al. (2019), updates will be applied monthly to ensure the security of the servers. Updates to the frontend and backend frameworks will also be applied to mitigate exploits and reduce our attack vector.

#### h. Site Adaptation Requirements

According to Gallavin et al. (2019), a server with at least 64 GB of RAM is recommended. Any server that Proxmox can run on should work. The Kerberos authentication is optional. The Logstash integration is optional but recommended. Rundeck will be required to run the automations.

### 2.2 Product functions

### 2.3 User characteristics

The majority of standard users of CSLabs will be freshman to senior computer science students of IUS. Most of the standard users are well educated and proficient with computers in general. Although IUS has a good diversity pool of computer science students of different age, gender, race, nationality, and ethnicity groups, it is very likely that more than half of CSLabs users will be white males students in their early 20s.

### 2.4 Constraints, assumptions, and dependencies

According to Gallavin et al. (2019), Proxmox servers’ memory requirement is the largest system constraint. Running multiple VM instances can take a large amount of RAM, and the system is planned to support up to30 VMs simultaneously. To optimize the overall system performance, VM memory allocation and CPU threads can be configured depending on the available hardware resources.

The following is a list of other dependencies in the system:

* .Net Core - Backend development framework
* C# - Backend programming language
* ReactJS - Front end development framework
* Typescript - Front end programming language
* React Bootstrap - Theming framework for ReactJS
* Proxmox - Hypervisor to run VMs
* Rundeck - Required to run automation on Proxmox

# **3. Specific requirements**

### 3.1 External interface requirements

* Kerberos requires the backend to communicate using its protocol
* Proxmox requires you to use their API that is useable in .Net Core
* Rundeck provides an HTTP API, which can be easily implemented in .Net Core

### 3.2 Functional requirements

The following is a list of proposed requirements, although it may not be practical to complete everything listed due to time constraints.

* Admins should be able to manage the virtualization servers from inside the application.
* The Team should add requested updates to the different features already in place.
* Creators should be able to link and embed videos.
* Creators should be able to remove users from a private module.
* Creators should be able to stop all usage of the module from users without deleting it.
* Admins should be able to stop all usage of a module from users and creators without deleting.
* Users should be able to review and rate modules and labs.
* Creators should be able to see reviews and rating.
* Creators should be able to make reviews and rating public or private for labs and modules.
* Admins should be able to hot migrate labs if a server needs maintenance.
* Admins should be able to see system statistics.
* Admins should be able to schedule maintenance.
* Users should be able to check off checklist’s boxes in a lab.
* Creators should be able to create checklists for labs.
* The applications should send emails to all users.
* Creators should be able to generate n many temporary accounts for a module.
* Users should have a button to extend a lab time.
* Users should be able to complete a lab if all checkboxes are checked in a checklist
* Creators should see a graphical representation of the network of the lab
* Creators should be able to create labs from an interactive GUI
* Users should be able to restart a lab
* User’s checklist on a lab should be persistent
* Users should be able to request to have an admin to upgrade their account to Creator

### 3.3 Performance requirements

Labs must be able to be created in just a few minutes. If users have to wait too long, then it will ruin the whole experience. SSDs will be used on priority labs to speed up the process. The web frontend will only require as much resources as the web browser you are using. The application is very light and can be run on older machines with 2 GB of ram using a lightweight browser.

**3.4 Design constraints**

TCA plans to dedicate approximately twelve hours a week to the project. Because of the finite amount of time available before the end of the Spring 2021 semester, it is expected that many desired features will not be implemented. This will require coordination between the project group and the stake holders.

#### a. Standards Compliance

The application Abides by the HTTP and HTTPS standards allowing browsers to utilize the backend. In the future the application may take payments which will subject the application to The Payment Card Industry Data Security Standard. We will gain compliance by keeping the server up to date, and use a third party payment system.

**3.5 Logical database requirement**

The backend relies on MariaDB which is an alternative to mySQL. MariaDB provides normalization of our entities allowing constraints to be applied to the relationships. The amount of storage space will be fairly minimal because we are only storing information about modules, labs, users, badges, and VMs.

**3.6 Software System attributes**

#### a. Reliability

The application contains automated tests alone with manual tests performed by the team. End-to-end tests help validate that things work in the real world when everything is communicating. These Unit, integration, and end-to-end testing will be performed before production releases to minimize bugs. When a bug is reported we will be on it in 24 hours.

#### b. Availability

During the first 3 months of use we guarantee 95% uptime, after that it will go up to 99.9%. In the early phases, we will be providing rapid updates as features arrive. We have multiple nodes in our Proxmox server to provide a HA environment.

#### c. Security

The servers will be running the latest fedora distro with the latest .NET Core runtime. The CSG group will be handling firewall rules on an external firewall to enhance the security of the application. NGINX will be used to proxy requests to the .Net Core Kestrel web server providing a tcp layer protection for the internal service.

#### d. Maintainability

The backend and front end systems are built in .Net Core and ReactJS in Typescript respectively. .Net Core is the future of Microsoft’s efforts to expand the use of C# to other operating systems. The overall design of the architecture is simplified with .Net Core. ReactJS provides an easy to use and componentized front end to increase code reuse and reduce bugs.

#### e. Portability

The frontend is very portable as it can run in all the major browsers. The hosting of the VMs requires significant investment of memory and disk heavy servers. Users can use either their school address or personal address allowing outside users to use this system as well.

### 3.7 Other Requirements

### None as of this moment.

**4. Key Personnel and Contribution Breakdown**

|  |  |  |
| --- | --- | --- |
| Position | Name | Contribution |
| Project leader | Lu, Yiliang | Conduct primary and secondary research; draft and edit reports; coordinate events and meetings; establish a liaison with external parties and advisors. |
| Full Stack Developer | Bello, Junet | Create and manage backlogs; project feasibility consulting; primary developer for the CSLabs backend; alternate project POC |
| Full Stack Developer | Martin, Cooper | Primary developer for the CSLabs web-app frontend; alternate event coordinator; unit testing |

Works Cited

Clifton, Zac et al. " CS labs Infrastructure Hardware Requirements Specification for CS labs Operations and Application."  14 Oct. 2019, <https://github.com/ius-csg/CSLabs-Capstone-Documentation/tree/master/cslabs-Infra-2019-2020/REPORTS>.  6 Oct. 2020

Gallavin, Jason et al. " CS Labs – Web Software Requirements Specification."  1 Oct. 2019, <https://github.com/ius-csg/CSLabs-Capstone-Documentation/tree/master/cslabs-web-2019-2020/DesignDocs>.  6 Oct. 2020