A Linux VM Based Virtual Networking Environment for Education Settings

Thomas Boxall

up2108121

BSc (Hons) Computer Science

School of Computing

Engineering Project (PJE40)

Supervised by Ioannis Kagalidis

May 2026

10,967 words

Declaration

Access Rights

- ☑ I consent to my dissertation in this attributed format (not anonymous), subject to final approval by the Board of Examiners, being made available electronically in the Library Dissertation Repository and/or Department/School/Subject Group digital repositories. Dissertations will normally be kept for a maximum of ten years.
- ☑ I consent for this document to be made available to other students as examples of previous work.
- ☑ I understand that if I consent, this dissertation will be accessible only to staff and students for reference only.
- ☑ This permission may be revoked at any time by e-mailing data-protection@port.ac.uk.

Ethics

- ☑ I understand that ethical approval was needed for this project regardless of whether I carried out primary research or not.
- ☑ I understand that not obtaining ethical approval is grounds for academic misconduct and may result in my failing my project module.
- ☑ I have received ethical approval for this work and include my ethics certificate at Appendix B (p75).
- N/A If I conducted primary research, I have included all raw data including research documentation (e.g.survey and participant sheet). I understand that not including the raw data can mean potentially failing the project as this is a mandatory requirement.
- N/A If I have carried out primary research involving employees of a company, I have included evidence i.e. a letter or email, showing that I have the company's permission to involve their staff in my research.
- N/A I confirm that the primary research carried out, matches the description in my ethics application.

Authenticity of work

- ☑ I confirm that I have read the University rules in respect of plagiarism and student misconduct.
- ☑ I declare that this work is substantially my own. Any material from an external source (e.g. text, images and source code) is appropriately cited and referenced.
- ☑ For engineering projects: I confirm that the majority of my artifact has been coded by me and is not substantially created by web-based site builders and other content management systems

Name: Thomas Boxall

Date: 6 May 2026

Abstract

A key method of learning about the way in which computer networks work is laboratory exercises. These either come at high complexity and high cost, when using enterprise hardware, or high complexity when using off-the-shelf virtualisation software; both of these are problematic in education settings. This project explores the feasibility of creating an experimental virtual networking environment using small Linux virtual machines simulating networking appliances. It finds that through using Proxmox as a type-1 hypervisor, lightweight Debian Linux virtual machines, and custom software it is possible to simulate a network of virtual routers and explore the way they interact using basic routing protocols. This unconventional paradigm uses a bespoke web management interface to control the application and leaves room for further development.

Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Ioannis Kagalidis, for his guidance and support throughout this project; as well as pushing me to tackle a more challenging project at the initiation phase. Thank you also to my moderator, Dr. Shikun Zhou, for his expertise and time in moderating my project. I would also like to extend thanks to my family & friends for their support throughout this project, for being compassionate sounding boards, and listening to my rants about “making the boxes behave”.

Contents

List of Tables

List of Figures

List of Abbreviations

API	Application Programming Interface
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
JSON	JavaScript Object Notation
Proxmox VE	Proxmox Virtual Environment
PVE	Proxmox Virtual Environment
VM	Virtual Machine
WCAG	Web Content Accessibility Guidelines
YAML	YAML Ain’t Markup Language

Chapter 1

Introduction

1.1 Project Background

A fundamental component of computer science, information technology and cyber security courses at the higher education (HE) level is a module teaching students about the basic functions of a computer network. This can be a complicated and difficult topic for students to understand (Janitor et al., 2010). In teaching the basics of computer networks, a common method used is a laboratory environment simulating networks which gives learners real-world experience. This has been proven to help them understand the topic better (Janitor et al., 2010). Various laboratory based approaches have been used to teach networking modules, however no method has proven superior; for example, Kizilirmak et al. (2023) conducted a study comparing hands-on learning in a laboratory environment and simulation-based learning in a virtual environment. They found that students had no preference, noting that students found the simulation environment helpful for visualising the network and for rapid testing.

This project explored the creation of an experimental virtual networking environment for education settings using lightweight Linux virtual machines (VMs) acting as network appliances. This alternative paradigm offers the opportunity for an easier to use application, which is tailored for entry-level learners while allowing for some cost savings.

1.2 Project Aims and Objectives

The aim of this project was as follows: *"To design and implement a Linux virtual machine based virtual networking environment for education settings, which has a web-based management interface for simple use."*

This aim was achieved through completing a series of objectives:

1. Carry out a literature review to understand the current publications, resources and existing solutions related to the topic.

    (a) Gain a high-level understanding of different pedagogical approaches to teaching computer networks and the role which virtual networking based education plays in this, providing context to this project.

   (b) Gain an understanding of the existing solutions on the market; their key features and their shortcomings.

2. Architect the technical solution to manage the provisioning of VMs, the control of guest operating systems, and to provide a management interface to end-users.

   (a) Explore and review free and open-source software which already exists to understand its functionality and shortcomings

   (b) Design a solution to connect the selected software packages together

3. Design and implement a control system for management of VMs on top of an existing type 1 hypervisor

   (a) System to be able to pass commands to the hypervisor to control hypervisor functionality

   (b) System to be able to pass commands to the VMs to control operating system level functionality

4. Design and implement a management interface for the application

   (a) System interface should provide simple controls to the user

   (b) System to provide educational advice to users to support their learning

5. Test the application

At the start of this project, a literature review was conducted which provided the context within which the project takes place. This review explored existing network virtualisation environments, and the pedagogical approach used in higher education to teach computer networks.

After the existing solutions were reviewed, the technical architecture of the solution was designed. This design underpinned the entire project, echoing the requirements elicited during the literature review. Through architecting the solution in one go, it was possible to ensure the interoperability of subsystems.

Once the overall design has been completed, the design and implementation of individual subsystems could commence. These subsystems work together to provide remote control over VMs network interfaces and configuration so that the user can configure networks.

Finally the system was tested. This end-to-end testing ensured that the system functions as expected as well as providing data used in the conclusions chapter of this report to reflect on the successes and limitation of the solution.

1.3 Constraints

There were a number of constraints imposed on this project, which are outlined below.

- The deadline for this project was fixed.
- This was only a BSc project.
- There was limited time for work on this project due to other studies and commitments.

1.4 Legal, Ethical, Professional & Social Issues

There were a number of pre-existing virtual network solutions available on the market; this means there was an inherent risk of copyright infringement. This was mitigated in this project by simply not copying an existing solution, rather exploring an alternative paradigm.

Due to the nature of providing remote control over Linux VMs, there were risks surrounding the security of the environment in relation to compromise and misuse by a threat actor. This was mitigated through ensuring only authorised individuals had physical or electronic access to devices used in this project. The development & deployment environments were also kept behind a firewall.

No personal information was gathered during this project; there were no research questionnaires or interviews.

No intellectual property was used or breached as part of this project.

1.5 Risk Management

Due to the nature of this project, there were a number of risks. These are outlined, including a mitigation measure, in Table 1.1 (p3).

Table 1.1: Project risk register

Title	Likelihood	Impact	Mitigation
Laptop failure	Medium	Delay to project, data loss	Monitor laptop health; backup data regularly
Bug in software package	Low	Delay to project, re-architecting required	Used well-known and well-used stable software

continued on next page

Table 1.1: Project risk register (continued)

Title	Likelihood	Impact	Mitigation
Scope creep	High	More work to be completed	Undertook requirements gathering process and stuck to requirements
Unexpectedly complex project	Medium	Work is harder, takes longer	Research completed during project initiation & literature review
Incapacity to work	Low	No work completed for period of time	Planned contingency into timeline & work hard while able

1.6 Methodological Approach

To ensure good time management for this project a Gantt Chart, Appendix C (p77), was developed during the initiation phase. This details the major activities to be completed during the project. The biggest section of the projects lifespan was implementation. To manage this time more effectively a Software Development Lifecycle was explored and used.

Somerville (2017) describes the Waterfall lifecycle as taking the individual stages as separate phases, moving through them one by one and not returning to a previous step once moved to a subsequent one. This method alone was not suitable for this project due to the fact that there were too many unknowns going into it, some of which ended up resulting in a redesign or shift in the fundamental structure of the application. Somerville (2017) continues to explore the Incremental approach where features are specified, developed and validated in an iterative fashion, each subsequent iteration building on the output of the previous. This was more akin to the rapid development needed for this project, with enough flexibility to be able to redesign a component should the need arise.

Ultimately it was decided to use Rapid Application Development (RAD) as the quick turnaround on prototyping, testing, and refining fit best with the nature of this application (Crelin, 2023). Primarily this was due to the fact that the requirements were set for the application as a whole and not for the individual components used to construct the application; through using RAD it was possible to focus on a single component for some time before moving to the next component while being flexible enough to shift priorities as issues arose. Everett and McLeod (2007) explore that through using RAD to develop software, large projects can have their implementation time cut down by 2-3 years.

1.7 The Report

This report is broken down into 7 main chapters.

After this introduction, chapter 2 (p7) explores existing literature, critically analysing pre-existing solutions on the market, and gains an understanding of their features and shortcomings. It also develops an understanding to the different pedagogical approaches to teaching computer networks, and the importance of virtual networking environments within these. Finally this review explores some of the technology used in this project.

Following on from the literature review, chapter 3 (p15) discusses the requirements for this application and how the requirements were gathered.

In chapter 4 (p23) the design of the application is discussed. It covers the architectural design of the application; proof-of-concept testing; and the design approach.

The development for this project will be recounted in chapter 5 (p35). It explores the development environment configuration; what order implementation was completed in; and provide commentary on decisions made during the implementation journey.

After development, chapter 6 (p43) demonstrates the testing conducted as part of the application development process. It includes commentary on debugging activities and the test strategy used.

Finally, chapter 7 (p49) brings this report to a close, analysing the artifact; reflecting on the journey; and exploring possibilities for future development.

Chapter 2

Literature Review

2.1 Introduction

A Literature Review was carried out to gain an understanding of the landscape of published literature relating to the topic. This review analyses existing virtual networking environments, their importance in the education settings, and use of type 1 hypervisors.

2.2 Searching Strategy

For this literature review, three search engines were used; *Google Scholar* and *EBSCO Discovery* provided a broad range of papers, and *IEEE Xplore* provided a very narrow set of papers which had already been returned by the prior two engines. Searches carried out returned high numbers of papers which made it hard to identify relevant papers, so a restriction was imposed whereby if more than 250 were returned, only the first 100 were reviewed. Appendix E (p81) contains the data from the searches conducted, which includes Figure E.1 (p81) containing a PRISMA diagram showing the searching and screening process used, and Table E.1 (p82) containing the searches conducted and number of results returned. Databases were searched with the criteria that the papers had to be published within the last five years.

Additional technical documentation and reports were sought out specifically where a gap in the literature emerged to provide clarity on specific subjects. Furthermore some sources were 'snowballed' from other sources, to deepen the understanding a source provided.

2.3 Virtual Networking

The underlying concept explored within this project is that of network virtualisation, which is the idea of abstracting physical networking resources and replacing with software defined implementations (VMWare, n.d.).

Commonly networks are virtualised using simulation which can be defined as "a program enabling a computer to execute programs written for a different computer" (Oxford English Dictionary, n.d.-c). Simulation is used for a number of reasons which all link back to the cost associated with purchasing and configuring a set of physical hardware for testing or devel-

opment, where a simulated network is often cheaper (Fujimoto et al., 2022). A commonly used example of a network simulator is Cisco Networking Academy's Packet Tracker (Cisco Networking Academy, n.d.-a).

An alternative paradigm to simulation is emulation, which can be defined as a "technique by which a computer is enabled, by means of special hardware or software, to execute programs written for a different type of computer" (Oxford English Dictionary, n.d.-a). Emulation is used within virtual networking to virtualise the hardware which runs a given specialist operating system (i.e. Cisco IOS). This gives a better experience to the user, as compared to a simulator, due to the fact that the emulated hardware will behave more similarly to that of the real life hardware (Neumann, 2015). A popular network emulator is Graphical Network Simulator 3 (GNS3) (GNS3, n.d.), which Helali (2020) details can be used to emulate any type of equipment from a multitude of manufacturers including virtual PC hosts for a holistic approach to network virtualisation.

Smera and Sandeep (2022) conducted research into the plethora of network virtualisation tools currently available, focusing on networks simulators. They found that over the last 40 years the market has advanced to match advancements in technology. Early simulators, such as NS-2 (NS-2, n.d.) were purely text-based with users interacting with it via scripts and textual commands (Fall & Varadhan, 2011). In more recent times, network virtualisation tools are designed for more specific use cases, such as Mininet (Mininet, n.d.) is oriented towards software defined network modelling. Alternatively, GNS3 (GNS3, n.d.) is more generalised and is suitable for modelling complex networks; or EVE-NG (EVE-NG, n.d.-a) being oriented towards academia and research. From this, it can be seen that there is no 'right' approach to network virtualisation and that there are no examples who utilise small Linux VMs as networking appliances.

2.4 Use of Existing Virtual Networking Environments in Education

It is a commonly accepted theory that learning about computer networks is a complex task which takes a considerable amount of energy both on the learner and teacher's part (Allison, 2022b). Often HE institutions use laboratory based sessions to apply theoretical concepts to real-world situations as this enables students to better grasp an understanding of the solution (Janitor et al., 2010). However, with increasing constraints on resources, it is becoming less and less possible for HE institutions to have dedicated network laboratories with specialist hardware. Compounded by the recent pandemic and its requirement for remote-learning, there has been a surge in HE institutions turning to virtualisation programs to either replace or supplement access to dedicated network laboratories (Kizilirmak et al., 2023).

In a study conducted by Kizilirmak et al. (2023) looking into the effectiveness of simulation-based learning compared to hands on learning, they found that learners had no preference

of either paradigm. Their study compared students experiences of first working alone to construct a small network using Cisco Packet Tracer and then working in groups to construct the same network using real-life hardware. Their survey of students experiences concluded that the perceived efficacy to the student was broadly similar between the simulation environment and the hands-on learning; however the students found that they spent longer working with, and encountered more problems when working on the physical hardware. Nurdiana et al. (2024) conducted a similar study, exploring students perceptions towards using simulator applications for virtual experiments. Their survey to students was considerably more extensive than that of Kizilirmak et al. (2023) which leads to a greater understanding of students likes and dislikes through asking a greater number of quantitative questions. Students were very satisfied with the usefulness, ease of use and ease of learning of Cisco Packet Tracer. In the study conducted by Nurdiana et al. (2024), they found that students scored the ease of learning factor of using Cisco Packet Tracer 3.79 on average. However Kizilirmak et al. (2023) found that the average was slightly higher, with the students rating that they learned the concept with an average of 4. This difference could be because the students in Kizilirmak et al. (2023) were also exposed to hardware experiments and found these more challenging than the software simulation experiments. Both studies captured student feedback through the use of a questionnaire, however only Nurdiana et al. (2024) processed the results against a Likert scale to capture a more holistic understanding of the students thoughts. Without any reference to whether the students were exposed to hands-on with hardware experiments, Nurdiana et al. (2024) found that students rated "Can Cisco Packet Tracer replace network computer lab facilities and infrastructure?" with an average of 3.97; without knowing if students were exposed, the truth behind the final figure is unreliable.

Arguably the most popular network simulation tool used within education settings is Cisco Networking Academy's Packet Tracer, boasting over seven million users (Cisco Networking Academy, n.d.-a). Packet Tracer is a simulation based virtualisation environment which offers learners the ability to build networks across different devices, not just routers and switches; it can be used to configure networks and understand how packets travel through the network (Cisco Networking Academy, 2024). However, Packet Tracer is designed to supplement physical equipment in the classroom, with Cisco recognising the importance of gaining hands-on experience with hardware as best practice (Cisco Networking Academy, n.d.-c). This echos the results of the study conducted by Kizilirmak et al. (2023), who found that the two learning paradigms complement each other. A study conducted by Allison (2022b) as part of teaching a first-year undergraduate module paired teaching theoretical concepts with nine Packet Tracer laboratory sessions, culminating in a final assessment to construct a complex network using Packet Tracer; with no use of a physical laboratory to supplement the simulation environment. They found that the laboratory sessions appeared to help the understanding of the theoretical concepts of networking, however in designing the first few sessions, Allison (2022b) had to assume that students had no knowledge of both Packet Tracer and networking to ensure that they were suitable for students with any amount, including no, prior knowledge. Allison (2022b) found that there were common issues arising throughout the module,

with students commonly not setting default gateways or IP addresses of the correct interface, using the wrong cabling and not saving configurations; which whilst all elements of dealing with real-world hardware, do certainly increase the complexity of learning for students who are entirely new to the field of computer networking. This is compounded into a more complex issue as Packet Tracer does not tell the user what the issue is when something is incorrectly configured, making it challenging to diagnose the fault. This paper utilises a tool in a way which it was not designed to be used in, and the methodology used by Allison (2022b) seems to have worked, with 76% of students passing the module; however there is no direct feedback from learners or other data gathered as part of this study which makes it hard to determine how the students obtained that pass-rate.

Extending into the professional world and to more advanced classrooms, it is common to find Emulated Virtual Environment Next Generation (EVE-NG) (EVE-NG, n.d.-a) being used. EVE-NG provides users tools to use around virtual devices and to interconnect them with other virtual or physical devices; simplifying the use, management and interconnection between them (Dzerkals, 2025). Furthermore, EVE-NG is not just for networking, it is also for testing the software within a simulated network environment, such as Active Directory or file servers. EVE-NG is installed on a server either using virtualisation or onto the bare metal (EVE-NG, n.d.-b), with clients connecting over HTTP(S) to a web interface to control the software (Dzerkals, 2025). This client-server paradigm is useful within education settings as the resource-intensive computing required to run the EVE-NG application and emulated devices does not have to be handled by the student facing machines in the laboratory. However, unlike the Cisco Packet Tracer, EVE-NG does not come with any inbuilt network devices, meaning there is additional configuration required to load images of network devices and set them up within the application (Dzerkals, 2025). In a questionnaire carried out by Harahus et al. (2023) which explored student satisfaction of using EVE-NG at University level, they found that the students enjoyed using EVE-NG and found it easy. Students found the documentation and setup of EVE-NG to be simple, with over 75% of students rating the process 4 or 5 out of 5 where 5 denotes the highest level of satisfaction; however they found it more challenging to load a new virtual device with an average difficulty score of 2.05. Qualitative feedback gathered through an open-ended question at the end of the questionnaire revealed that there was a significant need for detailed installation instructions and that the tool has a substantial learning curve; which when compared to Packet Tracer and the success experienced by Allison (2022b) suggests that EVE-NG is less suited as a virtual networking environment for education settings than Packet Tracer.

While purely software based virtual networking environments are most common within education environments, perhaps due to their simpler installation methods, Ma et al. (2024) explored the creation of ‘Klonet’, a virtual networking environment designed for education settings. Klonet improves on existing solutions by adding dedicated teaching tools, including a repository of experiments used both for students to review explore in the lab and as a way for lecturers to mark students work; as well as experiment tools such as built-in traffic generation and monitoring tools. Klonet is designed to run a distributed workload across a cluster

of physical servers, utilising Docker containers for emulating nodes with customised images made available including OVS switches, FRR router, and Ubuntu hosts. A key motivation for this project was to eliminate complex configuration of the virtual environment which had to be completed by the students, echoing the struggles experienced by Harahus et al. (2023). To combat this, Klonet runs in the Browser/Server architecture where no local installation of utilities is required for end-users (Ma et al., 2024). Furthermore, Klonet provides an Application Programming Interface (API) for control of the virtual environment as an alternative to the web-based Graphical User Interface (GUI). Klonet was designed to run experiments relating to algorithms such as OSPF, rather than teaching the basics of networking, students found this tool useful with 76.96% of undergraduate users saying that they "have a deeper understanding of course knowledge" (Ma et al., 2024).

2.5 Proxmox as a Type-1 Hypervisor

Dordevic et al. (2022) speculates that virtualisation can be considered the most important topic in IT at the present time. Given the lower cost and simplified maintenance a virtualised server presents over a physical server, it's to be expected that companies and educational environments are transitioning towards them. Portnoy (2016) defines virtualisation as the "abstraction of some physical components into a logical object." This is often seen as a VM representing logically a computer, or server. This VM, running within a hypervisor, will virtualise all of the hardware resources which a traditional computer would have direct access to. A Hypervisor is "a program that enables a computer system to host one or more different operating systems at the same time, managing access to hardware resources for each..." (Oxford English Dictionary, n.d.-b). Hypervisors fall into one of two main types. A type 1 hypervisor "runs directly on the server hardware without an operating system beneath it," whereas a type 2 hypervisor "is an application that runs atop a traditional operating system" (Portnoy, 2016). The core difference here is that a type 1 hypervisor is the operating system and is installed directly onto the bare metal, for example Microsoft Hyper-V (Microsot Learn, n.d.) or VMWare vSphere (Broadcom, n.d.-c); while a type 2 hypervisor is an application which runs inside the operating system already directly installed on the bare metal, for example VMWare Workstation (Broadcom, n.d.-a) or VirtualBox (Oracle VirtualBox, n.d.).

Many type 1 hypervisors are designed for commercial use, as they are designed to be installed directly onto servers. While Broadcom's VMWare ESXi product does have a limited free version (Broadcom, n.d.-b), these limitations may be seen as too restrictive especially for people requiring API access to the environment. On the other hand, Proxmox Virtual Environment (Proxmox VE, or PVE) (Proxmox, n.d.-e) is an open source type 1 hypervisor based on Debian GNU/Linux. PVE is a popular choice for educational projects or institutions because of its free Community Edition which has all the features of the paid-for edition, except enterprise support. PVE provides an intuitive web-based management interface as well as a REST API for complete programmatic access to all features (Proxmox, n.d.-b).

Ford et al. (2023) developed an environment for cyber security education using PVE as its core with the aim to create a system for provisioning dedicated cyber security education environments in a scalable, reliable and freely accessible way. Their system is based around PVE, with a command-line based management tool available to simplify administrator and instructor's control of the environment as compared to other pre-existing systems (Ford et al., 2023). The management tool is implemented using PVE's API using the Proxmoxer Python library (Proxmoxer, n.d.), and offers the ability to create a 'template' based on an existing VM; provision an environment for a learner based on a set of VMs and provided parameters such as IP addresses, DNS servers, subnet masks, etc; as well as to delete sets of VMs, for example at the end of the semester to clean up. Through their experimentation, to clone six VMs and configure all required networking took 14.96 seconds, which is a significant time-saving on what it would take for this to be completed by hand; however this is obviously dependent on what the guest operating systems are and the specifications of the host machines. The students access their environment through either a web interface or SSH through a Open VPN Server. The solution developed by Ford et al. (2023) utilised pfSense (pfSense, n.d.) as a firewall providing security to their application.

In a similar approach to Ford et al. (2023), an EVE-NG based virtual networking environment was designed by Savu-Jivanov et al. (2025). This hybrid virtual networking environment centres around PVE within which, EVE-NG is installed in a VM allowing emulation of networking appliances. Installing EVE-NG in this way allows for it to interact with other 'external' devices which are in reality other VMs within PVE all connected to an internal virtual Linux bridge network device. Furthermore advanced monitoring is implemented using Nagios and network security implemented through a pfSense VM sitting between the external network connection of the PVE host and the internal virtual Linux bridge network device. Savu-Jivanov et al. (2025) demonstrated through their project a way to create a virtual networking environment through linking together existing applications with no additional software development required. This is a different approach to Ford et al. (2023), however both were successful and accomplished the creation of a virtual environment for different educational contexts using PVE as the core component.

With both of the environments discussed above, there is some level of security implemented through the pfSense firewall. However, in a different environment proposed by Dumistracel et al. (2025), we see students being assigned a VM hosted within PVE for nondescript educational purposes, with concerns over students (un)intentionally creating insecure configuration. This demonstrates the need for ongoing monitoring at the guest level, especially given the rise in cyber incidents over the last few years (National Cyber Security Centre, 2025). Dumistracel et al. (2025) explores two off-the-shelf solutions to monitor both network usage and system resource utilisation, neither of which are suitable to be deployed in this context as they do not have the tools to capture the PVE guest ID or user information. It is important to capture the system resource utilisation as a deviation from normal usage may indicate a compromised machine; and to monitor the network traffic for origin/destination as anomalous data may

suggest an attack in progress. Through a custom monitoring application, Dumistracel et al. (2025) proposes to monitor the system resources by periodically querying the PVE API from a Python script to obtain current utilisation metrics which are stored in a database where anomaly detection takes place; which is dynamic in nature, as it responds to trends over time rather than comparing live data to static baseline data. This API access is granted using a dedicated API account which has restricted permissions within PVE, where it can only read data rather than make changes to the system configuration. Network traffic is analysed through capturing the packet header metadata and querying an online public IP database to determine the organisation which the IP address is registered to, which is then compared to entries in a trusted IP database. After a period of system learning, the system functioned effectively - providing alerts with less than 5% false positives out of 120,000 data points generated in 24 hours. From this figure, the importance of event-based security monitoring within virtualised environments can be seen, especially where the traditional network security paradigm may not be applicable.

2.6 Summary

In summary, a literature review has been conducted which ascertains the importance of virtual networking in education settings, as well as explores the technology used in these settings and in type-1 hypervisors more broadly. This information fed into the requirements elicitation process where existing simulation environments were explored in more detail to understand their function.

Chapter 3

Requirements

3.1 Introduction

This chapter will explore the process undertaken to elicit requirements for this project. It will then summate these findings into a set of functional and non-functional requirements.

3.2 Requirements Elicitation Process

Early on in this project, it was decided to not conduct primary research to gather user requirements. This was due to the fact that there existed a significant quantity of research relating to virtual networking environments in education settings and the possibility to combine strengths and weaknesses from these to build requirements for this project.

3.2.1 Reviewing Existing Literature

As seen in the Literature Review, chapter 2 (p7), there are numerous pre-existing Virtual Networking environments which have been used in Education Settings. Allison (2022b) makes the laboratory instruction sheets available (Allison, 2022a). The earlier worksheets in the set provide a useful insight into the level of functionality expected in a virtual networking environment. These worksheets begin taking students through the fundamentals of Packet Tracer, then explore the way to build a basic network comprising of switches, routers and workstations. The topics then move onto learning about static and directly connected routes; all of which are built-in functions of Packet Tracer.

A core issue with existing virtual network tools is their complexity. While, Packet Tracer is a teaching tool, it is also an enterprise grade network simulation tool. This means that there is a significant amount of knowledge required to be able to use the software in the first place, as evidenced by the requirement in Allison (2022a) for their students to complete Cisco provided introductory training before completing laboratory worksheets.

3.2.2 Reviewing Existing Systems

The 'Learn to Use Packet Tracer' tutorial (Cisco Networking Academy, n.d.-b) provides details on the key features of Cisco Packet Tracer. the software offers the ability for an end user to add 'devices' to the workspace and then apply configuration to these devices. These devices are based off real-world devices with names and configurations to match; the options allow users to configure IP addresses and applications running on the devices. The Packet Tracer user interface abstracts all underlying complexity of the simulation software; however it does expose a Cisco IOS style command-line interface for nodes to allow advanced control.

EVE-NG provides many similar options. While the nodes within this application represent real-world emulated devices, EVE-NG still provides the option to add and manage them through a user interface. Users can control the number, type and settings therein of nodes from a catalogue of available nodes (Dzerkals, 2025).

Both Packet Tracer and EVE-NG are highly complex virtual networking applications, offering countless devices which they can simulate or emulate, respectively. They can be used to virtualise entire corporate networks and as such provide numerous complex features which would not be used in a basic networking education environment. This makes them too complex for use in education settings; therefore the requirements for this project have been extrapolated from the core functionality of these enterprise grade applications.

3.3 Requirements Specification

Requirements for this project have been divided into two types: functional and non-functional. Somerville (2017) defines functional requirements as "... statements of service the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations." These requirements were used to guide the developmental journey of the project to ensure they were met. Somerville (2017) defines non-functional requirements as "... constraints on the services or functions offered by the system." Whilst at first glance it may seem that the functional and non-functional requirements are defined similarly, the difference is nuanced. Non-functional requirements look at the way in which the system behaves rather than the operation that the system performs.

All requirements were assigned a priority as follows:

- P1: must be implemented to ensure the core functionality of the application.
- P2: should be implemented to ensure the quality of operation of the application.
- P3: could be implemented to add functionality to the application.

The functional requirements can be found in Table 3.1 (p18), and the non-functional requirements can be found in Table 3.2 (p21).

3.4 Summary

In summary, existing network virtualisation software has been reviewed and a set of 13 functional and 2 non-functional requirements elicited. These underpinned the design work in the following chapter.

Table 3.1: Functional requirements

ID	Title	Description	Priority	Acceptance Criteria
FR01-1	Control Interfaces	Users should be able to control the number of interfaces.	P1	A user can operate the web management interface to add, edit and remove virtual interfaces attached to a VM. The software will interpret these commands and runs the required operation on the hypervisor.
FR01-2	Control IPs	Users should be able to control the IP addresses and other associated parameters assigned to an interface.	P1	A user can operate the web management interface to add or edit IP addresses and other associated parameters assigned to a virtual interface. The software will interpret these commands and run the required operation on the VMs.
FR01-3	IP Addressing Support	The software should provide feedback to the users when they enter an incorrect IP address.	P2	A user can enter an incorrect IP address into a field on the web management interface and the system tell them what is wrong about it to support them to learn.
FR01-4	Control Quantity of Virtual Networking Appliances	Users should be able to control the number of virtual networking appliances through the web management interface.	P1	A user can operate the web management interface to spin-up or spin-down a virtual networking appliance, specifying its type and configuration parameters. The system should interpret these commands and run them on the hypervisor / VM as needed.

continued on next page

Table 3.1: Functional requirements (continued)

ID	Title	Description	Priority	Acceptance Criteria
FR02-1	Visualise Network	The software should allow the users to visualise the network.	P2	A user can view the web management interface and see a diagrammatic representation of the network.
FR02-2	Application Control	Users should be able to control the application entirely through the web management interface, after its initial boot.	P1	The users should be able to perform all the core functions of the application through interacting with standard form-controls on the web management interface rather than having to enter commands into a command prompt.
FR03-1	Node Type: Router	The software supports a virtual network router.	P1	Within the application there is a simulated router running in a VM which the user can interact with, and it performs basic router functionality.
FR03-2	Node Type: Switch	The software supports a virtual network switch.	P2	Within the application there is a simulated switch running as a VM which the user can interact with, and it performs basic switch functionality.
FR04-1	Generate Traffic	The software should allow users to easily generate traffic on the network from one host to another host.	P2	Users are able to operate the web management interface to generate traffic from one host, specifying a destination host.

continued on next page

Table 3.1: Functional requirements (continued)

ID	Title	Description	Priority	Acceptance Criteria
FR04-2	Monitor Traffic	The software should report traffic statistics to users allowing them to see how traffic propagates through the network.	P3	Users are able to view the web management interface and see real-time traffic monitoring.
FR05-1	Configure Routing Protocols	The software should allow users to configure routing protocols used per-interface on a router node.	P1	Users are able to define a routing protocol for a given interface on a router node through a simple UI control.
FR05-2	View Routing Information	The software should have a way for users to view the information generated by routing protocols, including protocol-specific information.	P2	Users are able to view routing information within the web management interface.
FR05-3	Perform Traceroutes	The software should include a way for users to perform a traceroute from a node to a specified node.	P2	Users are able to enter an IP address of a specific interface into an input field and the system will perform the traceroute, returning the output to the user.

Table 3.2: Non-Functional requirements

ID	Title	Description	Priority	Acceptance Criteria
NR01	Accessibility	The Web Interface should conform to modern web accessibility standards.	P1	Run the Web Accessibility Evaluation Tool (WebAIM, n.d.) on the web interface and receive 0 errors.
NR02	Ease of Use	The web interface of the system should be easy to use, taking no more than 10 minutes of explanation.	P2	New users should be able to be inducted on how to use the system within 10 minutes and then they should be able to independently conduct experiment.

Chapter 4

Design

4.1 Introduction

In this chapter, the design of the solution is discussed. First the architecture of the solution will be discussed and evaluated, then a number of proof of concept tests will be recounted, which supported the design process. After this the persistent data storage options will be discussed and selected before finally the user interface will be designed and justified.

4.2 Architecture of the Solution

The architecture of the application has three key components:

- Web Management Interface
- Type 1 Hypervisor
- VM Control Application to run on Linux VMs

This architecture has been designed such that it is possible to develop the different discrete components at separate times then connect them together into one homogenous application. The interfaces between the different components are HTTP-based REST APIs due to their interoperability and support for different platforms.

4.2.1 Web Management Interface

The web management interface is comprised of two components, as can be seen in Figure 4.1 (p24). As a whole, this subsystem runs in a Docker container, which runs within the AppServer VM on the PVE host. This VM also contains a 'resource-server' Docker container that is used in the initial deployment stages of the application; further details of which are available in Appendix H (p125).

The first sub-component is the front end application with which the users interact. This provides users a simple GUI to control the VMs and configuration therein, abstracting complexity to a series of simple buttons and input fields. This was designed to use client side technologies

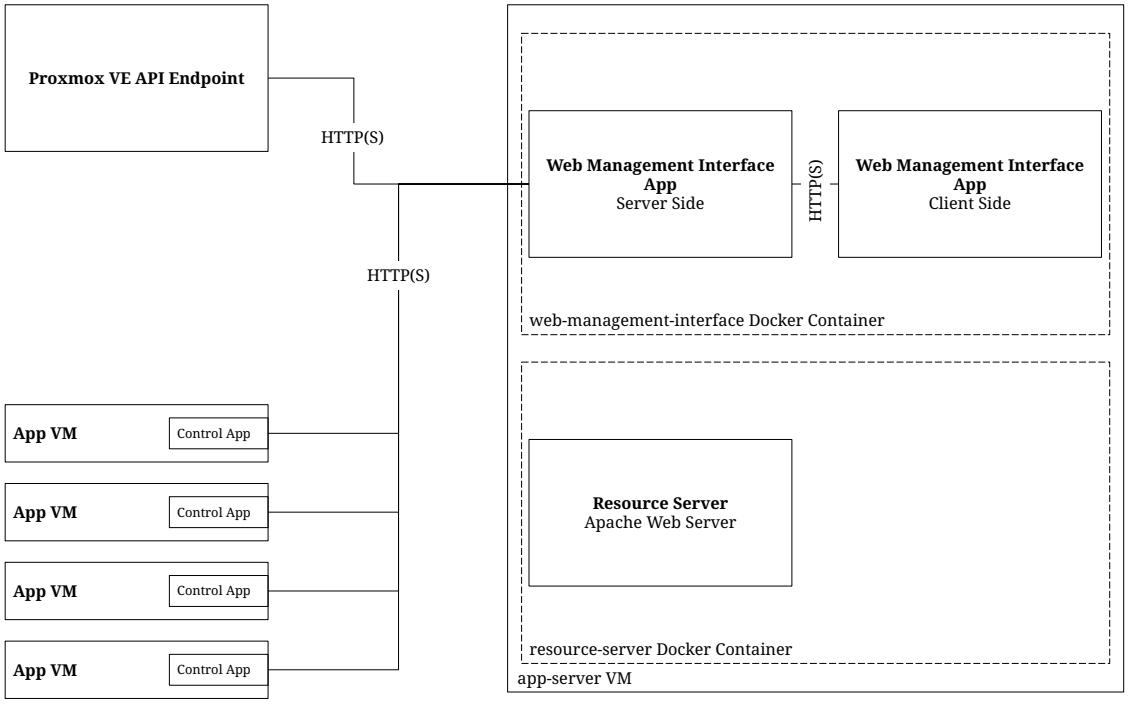


Figure 4.1: Logical system architecture

only, making requests to the 'server' component of the web management interface. A user interface framework was used to ensure consistent and aesthetically pleasing styling of this application.

The second sub-component of the web management interface is the 'server' component. This runs in the same docker container and is designed using a server side language to negate CORS issues with using client side JavaScript to call remote servers. This component also manages the persistent data storage for the application.

The client-side component and server-side component of the Web Management Interface interacts through HTTP requests using body parameters to pass data between them.

4.2.2 Type 1 Hypervisor

The core component of this application is the type 1 hypervisor which hosts the VMs that act as networking appliances. As seen in section 2.5 (p11), PVE is a popular choice for this setting, so this was used. The web management interface server-side component interfaces with the PVE host using the PVE API (Proxmox, n.d.-c).

Proxmox VE was decided on for this project due to the open-source nature of the software which contributes to meeting one of the objectives of this project. Figure 4.2 (p25) shows the physical architecture of the solution, focusing on the relationship between the different components of the project to the type 1 hypervisor and the communications between them.

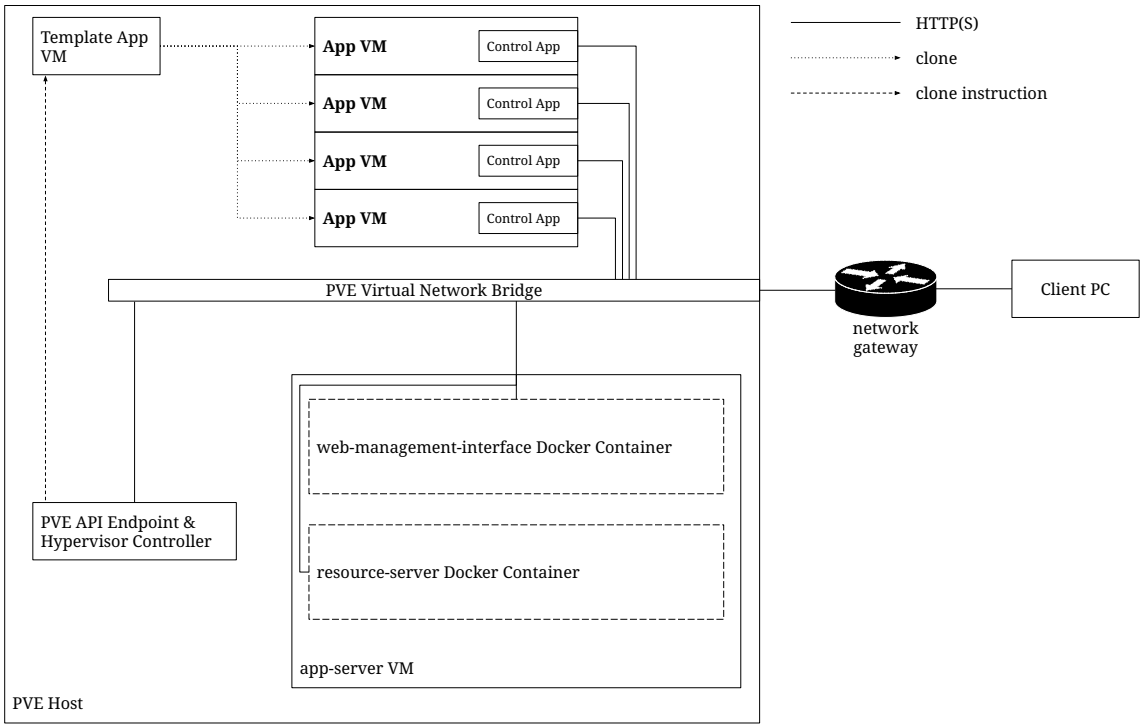


Figure 4.2: Physical system architecture

4.2.3 VMs & Control Application

The final component of the architecture of this solution sits furthest away from the end-users; Linux VMs running within the PVE host which behave as networking appliances. They run a bespoke application exposing a HTTP API endpoint for the web management interface server-side component to interface with.

It was decided to develop a bespoke control application as this provides some amount of platform independence for the VMs, meaning a VM could be run through a different hypervisor or as a physical machine; this being the alternative to utilising the QEMU Guest Exec options within the PVE API.

4.3 Proof of Concept Testing

As part of the development of the architecture of the solution, initial proof of concept testing was conducted to confirm hardware and software behaved as expected.

4.3.1 Test 1: Inter-VM Routing

The first test conducted was to confirm if it is possible to configure an Alpine Linux VM as a router to route a second Alpine Linux VM's network traffic through, using the network shown in Figure 4.3 (p26). The test was successful with `testenv-a-02-cli`, Figure 4.5 (p27), being able to route through `textenv-a-01-rt`, Figure 4.4 (p27) to the wider internet. The `traceroute`

command was used to verify routes used.

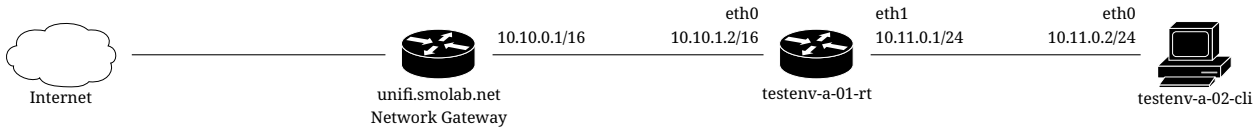


Figure 4.3: Proof of concept test 1 - network diagram (icons: Cisco, n.d.)

4.3.2 Test 2: HTML Canvas

For the web management interface, it was hoped to use a Canvas element to provide a graphical representation of the virtual network created. This component was tested to ensure it operated as expected as the available documentation (MDN, n.d.) is complex. The test was successful, and demonstrated that the HTML Canvas element can be controlled using JavaScript so therefore would be a viable technology to design into the application. Figure 4.6 (p28) shows the tests conducted with the Canvas as well as the Firefox DevTools console showing debug information used to verify the application was working as expected.

4.4 Persistent Data Storage

It was decided to not use a relational database for persistent data storage. This was because the data to be stored between sessions would always be needed in one go, as the client application requires all the data within the application to be able to correctly render the GUI. Furthermore, including a database in this solution would further complicate the application, making development and maintenance more challenging. It was explored to use a lightweight database engine such as SQLite (SQLite, 2026) however this still included a too-significant developmental overhead to be considered an appropriate option for this application.

Instead of using a relational database, it was decided to use a flat file to store the application data. This provides the benefit that there are very few complexities in configuring and maintaining this file as well as the fact that a structured flat files such as YAML Ain't Markup Language (YAML) (YAML Language Development Team, 2021) or JavaScript Object Notation (JSON) (ECMA International, 2017) provides the ease of access a relational database provides without any of the added data processing complexities, as all components of the application can handle the data in the same format from storage to use in the client-side management interface.

The schema defining the persistent data storage JSON file can be found in Appendix F (p85).

4.5 User Interface Design & User Experience Considerations

The non-functional requirements for this project stipulate that the User Interface (UI) must be accessible to user using assistive technology (NR01) and easy to use (NR02). To achieve NR01,

```
testenv-a-01-rt:~# ip r
default via 10.010.0.1 dev eth0 metric 1 online
10.10.0.0/16 dev eth0 scope link src 10.10.1.2
10.11.0.0/24 dev eth1 scope link src 10.11.0.1
testenv-a-01-rt:~# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 46 byte packets
 1  unifi.smolab.net (10.10.0.1)  0.601 ms  0.350 ms  0.240 ms
 2  10.34.128.1 (10.34.128.1)  0.836 ms  0.844 ms  0.681 ms
 3  gw-vlan2577.edge2.tcw-r1.ask4.net (213.143.8.65)  7.853 ms  15.385 ms  5.818 ms
 4  core1.lond1.uk.ask4.net (213.143.0.134)  5.195 ms  5.356 ms  core2.lond1.uk.ask4.net
    (213.143.0.136)  5.255 ms
 5  core1.lond1.uk.ask4.net (213.143.0.128)  5.147 ms  5.192 ms  5.183 ms
 6  *  *  *
 7  *  dns.google (8.8.8.8)  5.734 ms  5.968 ms
testenv-a-01-rt:~#
```

Figure 4.4: Proof of concept test 1 - router output

```
testenv-a-02-cli:~# ip r
default via 10.010.0.1 dev eth0 metric 1 online
10.11.0.0/24 dev eth1 scope link src 10.11.0.2
testenv-a-02-cli:~# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 46 byte packets
 1  10.11.0.1 (10.11.0.1)  0.397 ms  0.323 ms  0.201  ms
 2  unifi.smolab.net (10.10.0.1)  0.536 ms  0.579 ms  0.515 ms
 3  10.34.128.1 (10.34.128.1)  0.914 ms  0.984 ms  0.876 ms
 4  gw-vlan2577.edge2.tcw-r1.ask4.net (213.143.8.65)  6.145 ms  6.128 ms  6.112 ms
 5  core1.lond1.uk.ask4.net (213.143.0.134)  5.658 ms  5.572 ms  5.504 ms
 6  core1.lond1.uk.ask4.net (213.143.0.128)  5.508 ms  5.894 ms  *
 7  *  *  *
 8  *  dns.google (8.8.8.8)  6.008 ms
testenv-a-02-cli:~#
```

Figure 4.5: Proof of concept test 1 - client output

the UI should conform to the Web Content Accessibility Guidelines (WCAG 2) standard (W3C, 2025). To achieve NR02, the UI should conform to Nielsen's 10 Usability Heuristics for User Interface Design (Jakob Nielsen, n.d.). The design for the interfaces can be seen in Figures 4.7–4.14

4.5.1 UI Framework

From early on in this project, it was clear that the best option for UI design would be to utilise an existing UI framework and customise, where needed, to suit this application. A number of different frameworks were explored, each evaluated against the following criteria:

- Aesthetics: Does the UI framework look nice?

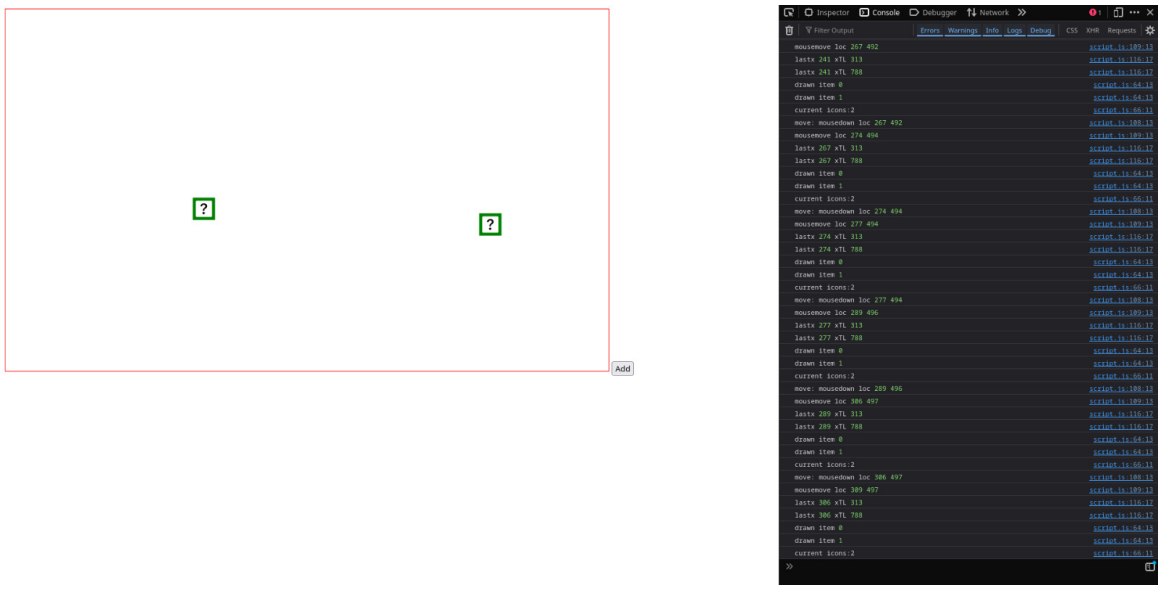


Figure 4.6: Proof of concept test 2

- Functionality: Does the UI framework work across different device sizes, automatically scaling to fit?
- Simplicity: Is the UI framework simple to implement, does it require additional overhead which will complicate code and future maintenance?
- Accessibility: Is the UI framework compliant with WCAG requirements out-of-the-box when used correctly?
- Documentation: Is clear and concise documentation provided?
- Examples: Are there detailed examples provided?

Pico CSS (Pico CSS, n.d.) was explored first. This provides a simple CSS framework designed around semantic HTML rather than complex class-lists. Comprehensive documentation is provided, however no claims are made, nor is documentation provided, as to its accessible compliance.

Material Design 3 (Google, n.d.-a) was then explored. This design system from Google provides comprehensive guides and components for UI design across multiple platforms. It comes complete with extensive development documentation targeting the web platform; however this design system is targeting large, complex, applications which makes it unsuitable for this project.

Bootstrap (Bootstrap Team, n.d.) was the last UI framework explored. This provides a simple and responsive style framework which works natively with WCAG requirements and provides comprehensive documentation. Whilst it is more complex than the Pico CSS framework, the Bootstrap framework provides customisation making it suitable for numerous different appli-

cations, this project included.

Ultimately it was decided to use the Bootstrap UI framework because it looks nice, is natively responsive, and simple to implement, as well as supporting WCAG requirements and providing extremely comprehensive documentation & examples.

4.5.2 UI Design Principles

The web management application was designed as a single-page application, making use of modal components to add additional user interface controls when required. Icons were be used as primary action indicators. This simple UI makes the application easy to understand, without lots of complex menus and hidden buttons, working towards meeting requirement NR02.

4.5.3 UI Wireframe Mock-ups

Wireframe UI designs were produced to understand how the UI might look. This allowed for design work to take place in a graphical editor rather than prototyping in code, speeding up the design process. Not all components are shown on these wireframes, such as icons used on buttons or form input labels.

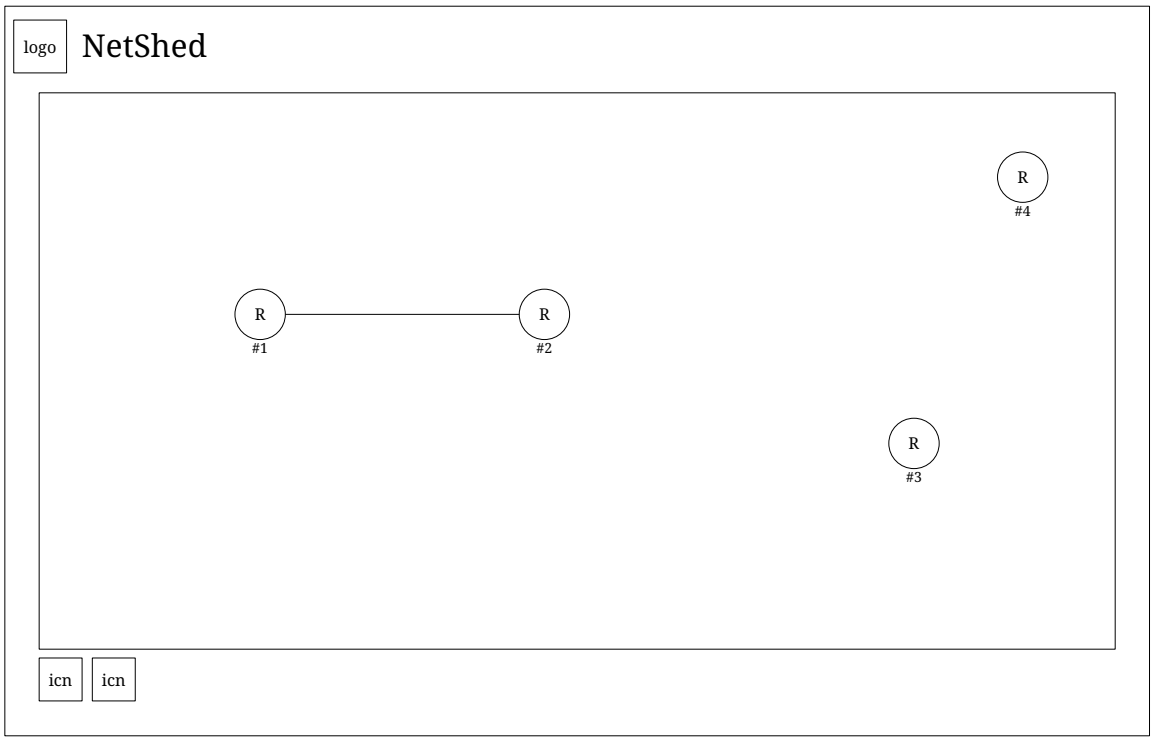


Figure 4.7: Wireframe design of main app page

Figure 4.7 (p29) shows the main application page. There are four nodes displayed, with two connected. Currently no nodes are selected, however when a node is selected two additional buttons appear below the node canvas; one to delete the selected node and one to open the

nodes configuration modal. Each node on the diagram represents an active VM, which may or may not be configured.

As introduced above, all the controls for a single node are situated within a modal which uses a tabbed navigation bar to provide five views. Figures 4.8–4.12 show the wireframe designed of this modal for a router node. Note that designs were not completed for the switch node type due to opting to not implement it, further details in subsection 5.5.6 (p41). Figure 4.8 (p31) shows the interface configuration tab where the IP address, subnet, PVE network ID, routing protocol and link state of interfaces can be controlled, as well as adding and removing interfaces. Figure 4.9 (p31) shows the routes tab where the standard Linux routing table and more detailed protocol specific routing information can be viewed. Figure 4.10 (p32) shows the actions tab where ping and traceroute operations can be performed. Figure 4.11 (p32) shows the status tab which gives status information about the VM. Figure 4.12 (p33) shows the help tab which contains help information to assist the user with the operation of the app.

An additional 'Welcome Modal' is used to welcome users into the application and is rendered whenever there are 0 nodes on the canvas on page load. Figures 4.13 & 4.14 show the two tabs of this; one for welcoming users into the application and another for providing help information.

4.6 Summary

In summary, the architecture, the look and the feel of the solution has been designed, while undergoing a review of possible options. This information was used in the next chapter to develop the solution.

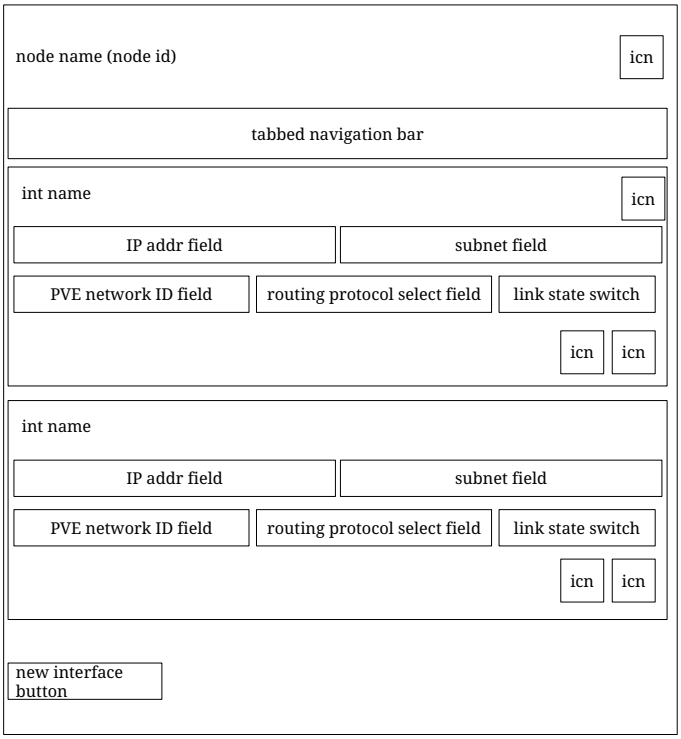


Figure 4.8: Wireframe UI design for interfaces tab of configuration modal

Figure 4.9: Wireframe UI design for routes tab of configuration modal

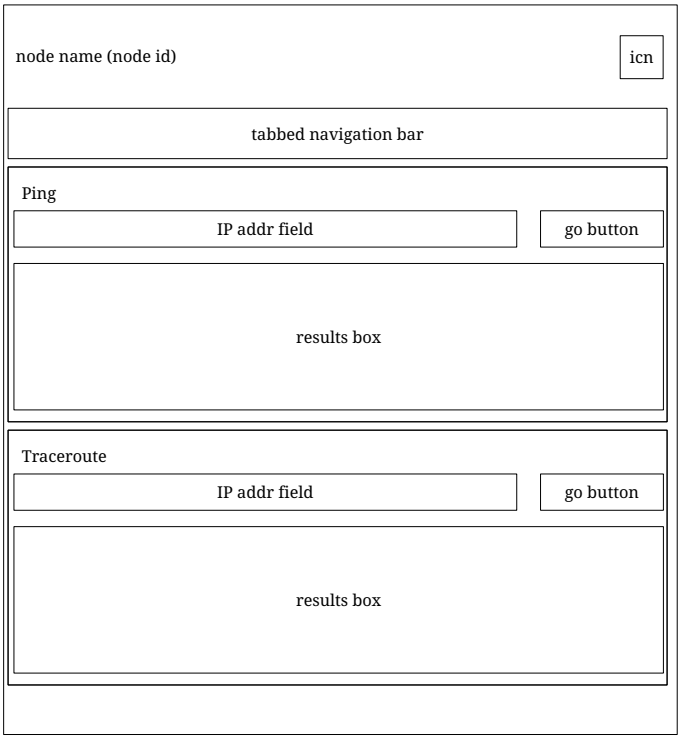


Figure 4.10: Wireframe UI design for actions tab of configuration modal

Figure 4.11: Wireframe UI design for status tab of configuration modal

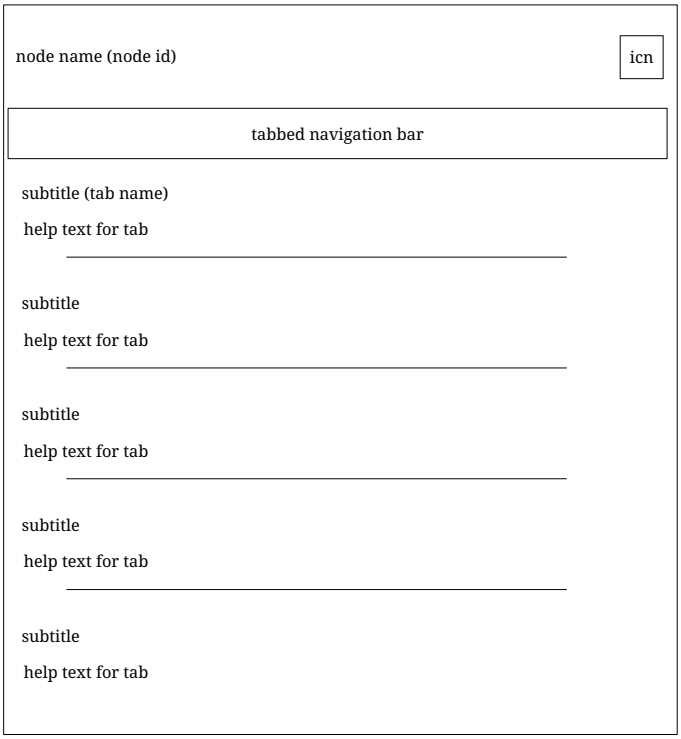


Figure 4.12: Wireframe UI design for help tab of configuration modal

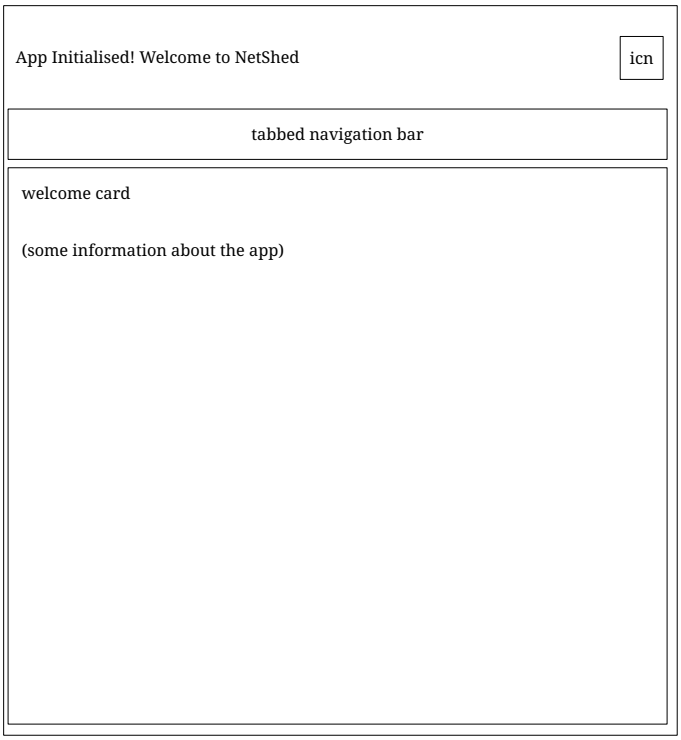


Figure 4.13: Wireframe UI design for welcome tab of welcome modal

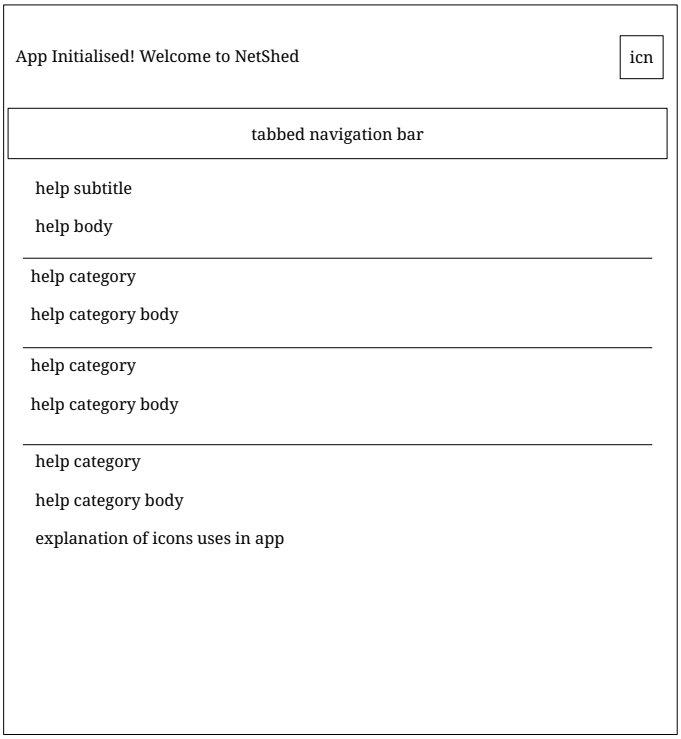


Figure 4.14: Wireframe UI design for help tab of welcome modal

Chapter 5

Development

5.1 Introduction

This chapter will discuss the development stage of the project. First it will recount the configuration of the development environment and the technology used for this. After which the implementation of the RAD methodology will be explored before finally issues encountered during the development stage and their resolutions are discussed.

5.2 Development Tools

A number of different development tools were used to support the development of this project.

5.2.1 Code & Version Control

The application code and any supporting technical files for this project were stored in a single git repository (a 'monorepo') which was hosted on the GitHub platform. Using a monorepo allowed for simpler code management with a number of discrete applications needing to work together as a unified solution.

5.2.2 Development Environments

The primary development environment used for this project was Visual Studio Code. This was selected due to the fact it works with a large number of languages so could be used for all languages used in this project providing a unified experience when developing each discrete component.

Development was conducted on a laptop running Fedora 43, with test versions of the VM Control Application being transferred using SCP to test VMs. This allowed for easier development as the development environment only had to be configured once. Furthermore, the Docker container running the web management interface could be run locally on the laptop used for development - saving time as files did not need to be transferred to a remote host.

5.2.3 API Development

During the development of the HTTP-based REST APIs in this project, both the VM Control Application and the server-side web management interface, and to test the PVE API - Postman was used. Postman (Postman, n.d.) provides the ability to test an API in isolation without needing the component which calls the API to be functional. This decoupling enables faster development as it is possible to focus on one component of the solution at a time.

5.3 Technology Stack

Due to the complex nature of this project, each component required a slightly different technology stack to function.

5.3.1 Web Management Interface

The server side of the web management interface was developed using PHP (Bergmann, n.d.) which was selected due to its simple yet powerful syntax, good documentation, and ability to work without complex routing libraries. The individual API 'routes' were configured such that they returned JSON encoded responses as this is interoperable with the JavaScript-based client side.

The client side of the web management interface was developed using client side JavaScript, HTML and CSS. This was selected due to its simplicity and good documentation. As discussed in subsection 4.5.1 (p27), the Bootstrap frontend toolkit was used to style the frontend of the application.

The web management interface was deployed in a Docker container within a Debian Linux VM on the hypervisor host to keep the entire application able to be hosted on a single piece of hardware. Alternatively, this could be hosted in the same Docker container on the end-user's PC.

5.3.2 Virtual Machines

The VMs to be used in this project ran the Debian Linux distribution. Initially, it had been thought they could run Alpine Linux, however due to issues explored in subsection 5.5.1 (p38), this was not suitable. Each VM is assigned 2 CPU cores, 4GB of RAM and an 8GB hard drive. The VMs ran on a HP Z440 Workstation equipped with 32 CPU cores, 128GB RAM, and a 1TB SSD which is running PVE version 9.

5.3.3 Virtual Machine Control Application

The application developed to provide external control over the function of the VMs was developed using Dart (Dart Team, n.d.-a) with the Shelf framework (Dart Team, n.d.-b) to provide

a simple HTTP server. Dart was selected due to its ability to be compiled for Linux devices, its ability to run Bash commands directly from a Dart application itself, and as it is easy to deploy as a systemd service which can run as the root user to mitigate any permission issues within this project. The Dart language has extensive documentation (Google, 2025) making it an pleasant language to work with.

For router nodes, the custom control application interfaces with the FRRouting package which provides an OSPF and RIP implementation (FRRouting Project, n.d.).

5.4 Development Approach

In line with the Rapid Application Development (RAD) approach used for the development process of this application, a number of different stages were used to provide focus to the otherwise unstructured iterative cycle.

The first stage to be developed was the Linux VM router control application. This was developed first as it underpins the rest of the entire application and any issues encountered with this would have resulted in a redesign of some other components of the application. After hand-configuring the machines to behave as expected, it was possible to write Dart code to automate this process.

The second stage to be developed was the management interface. This stage began with development of the frontend, ensuring that the visual components would function as expected; then moved onto the backend components and the integration to the router control application.

The third stage to be explored was the management of the hypervisor through its APIs and therefore also the cloning and disposing of VMs. There were a number of options as to how to implement this, hence it was covered so late in the process. Through a combination of the extensive PVE API documentation (Proxmox, n.d.-d) and Postman, it was possible to identify and test the PVE API routes before integrating them into the Server Side of the web management interface.

Finally, the last planned stage to be developed would be the educational components for the front end, as well as accessibility options to meet FR01-3 and NR01. However this, as well as the implementation of a Switch Virtual Appliance (FR03-2), were not achieved due to time constraints.

5.5 Development Issues Encountered

Throughout the development process, a number of issues or challenges were encountered. These have been recounted in this section.

5.5.1 Alpine Linux C library

At the time of beginning to develop the VM control application, it was intended to use Alpine Linux as this is incredibly lightweight. Alpine Linux had been tested with the requisite network configuration which had been successful. However on testing the functionality of the VM control application, it was discovered that Dart ahead of time compiled applications are not compatible with Alpine Linux.

Dart complies to use glibc (GNU C Library), and when building for Linux it requires the glibc dynamic linker to be present. Alpine Linux does not use the glibc library, rather it uses the musl libc library. This issue, and Dart's incompatibility with Alpine Linux, is something which has been discussed at length in an issue on the Dart GitHub repository (TheOneWithTheBraid, 2020) - with the Dart Development team acknowledging the issue. There are a number of workarounds to run glbic compiled code on Alpine (Alpine Linux, n.d.), however they add significant complexity to the overall implementation, so it was decided to explore other options hence the move from Alpine Linux to Debian Linux.

5.5.2 Designing an Algorithm for Visualising Node Connections

As part of the UI development, an algorithm had to be developed to visually represent the connections between different nodes. The nodes themselves are displayed on a HTML Canvas element so naturally using JavaScript to draw lines between connected nodes was the logical choice. After developing the algorithm to identify pairs of nodes to connect together, where their PVE Network IDs matched, the next step was to use the saved coordinates of the nodes, the top left corner, and connecting them together. This can be seen in Figure 5.1 (p38).

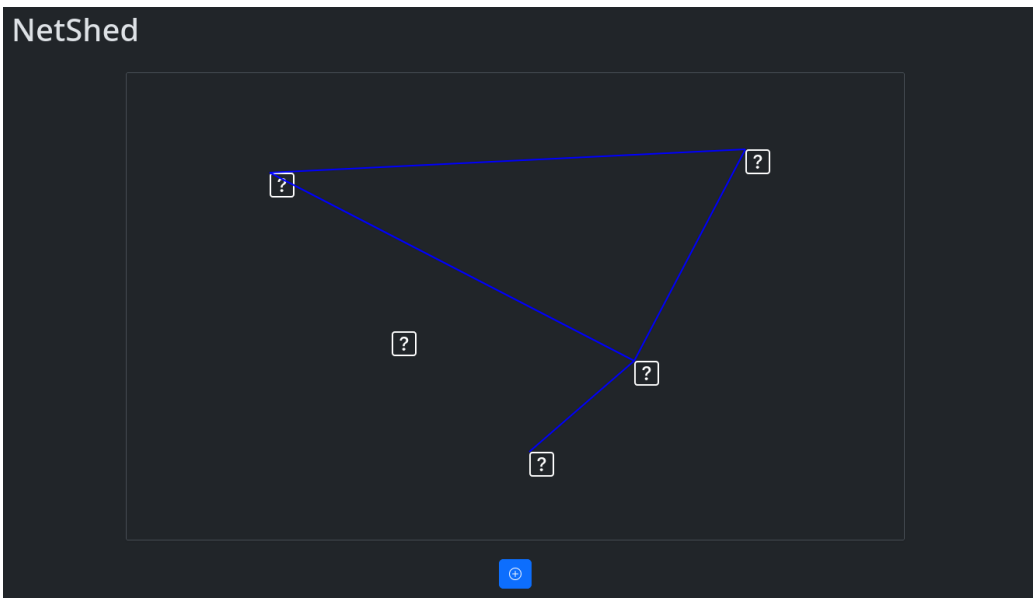


Figure 5.1: Node connection algorithm step 1

After connecting the known points of the nodes, an algorithm was designed to locate which side of the node to grow the line from, and therefore which side of the second node to connect

into. This algorithm used the smaller of the difference in x coordinates and y coordinates to decide whether to extrude from the node horizontally or vertically.

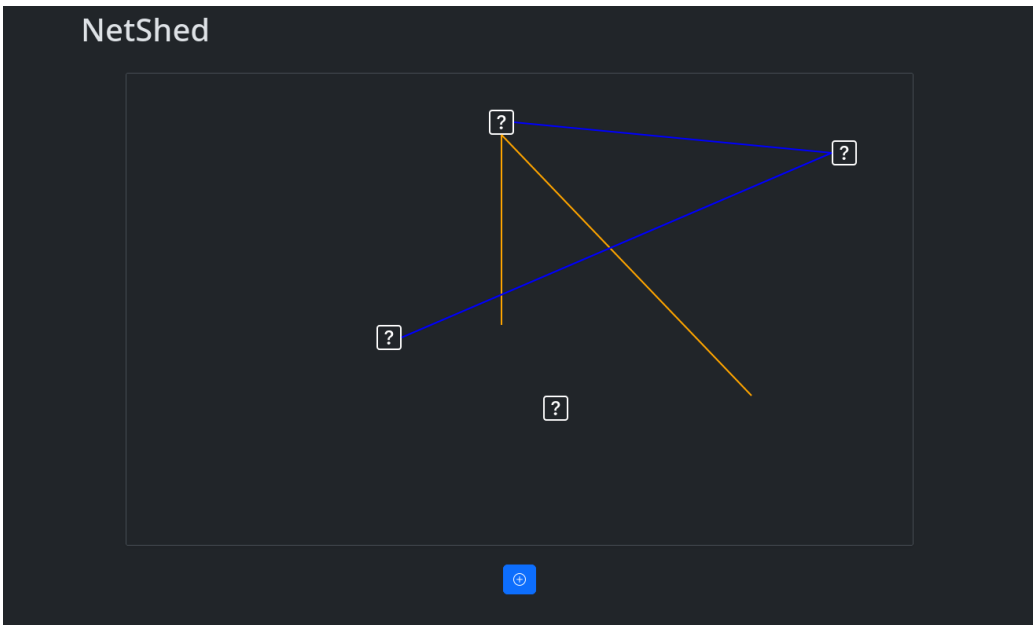


Figure 5.2: Node connection algorithm step 2

As can be seen in Figure 5.2 (p39), this algorithm didn't work at first due to an incorrect variable usage. This was corrected and the node icon changed with the finished result visible in Figure 5.3 (p39).

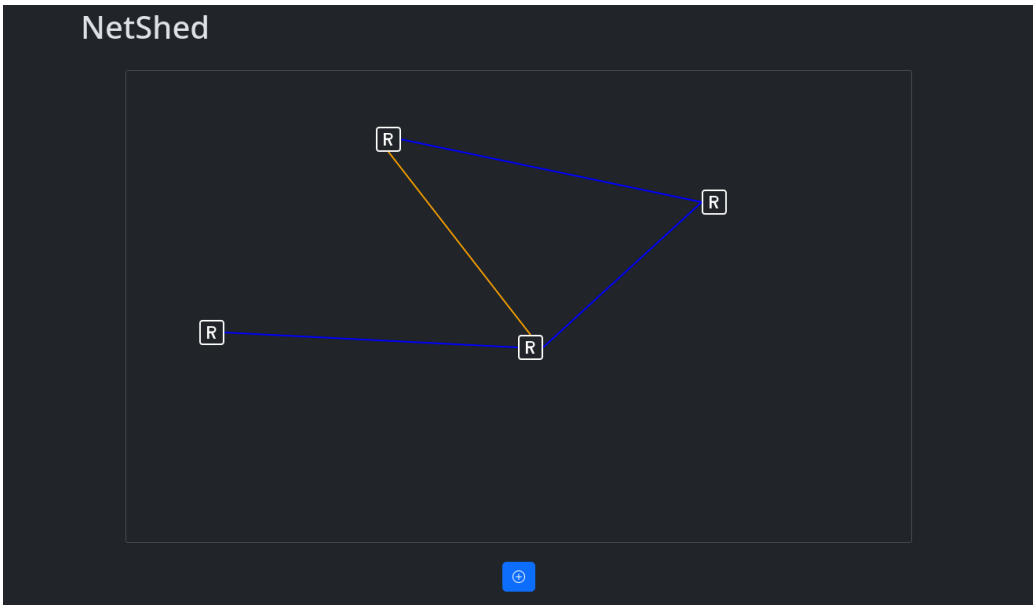


Figure 5.3: Node connection algorithm step 3

5.5.3 Deleting an Unconfigured Interface within VM

As part of the development of the Router API application to run on the VM and provide external command over the VM, an error was encountered in the function which deletes the

configuration file for an interface. This file would only be created when configuration was applied to an interface, not when the interface was created. So in the event that the interface was created and no configuration applied, then deleted - an error would be thrown. Due to the error handling methods in the Router API software - this would return a HTTP 500 error code, which the PHP Web API would interpret as failed operation. However this is not the case, as the interface would still successfully be deleted. The error received by the PHP Web API can be seen in Figure 5.4 (p40).

```
"vmrespDecoded": {
"status": "500",
"stdOut": "",
"stdErr": "rm: cannot remove '/etc/network/interfaces.d/ens21':
            No such file or directory"
},
```

Figure 5.4: Unconfigured interface deletion error

There were two options to solve this issue: one being to edit the interface deletion command to add error handling for this situation; or to modify the new interface command to create an empty file, which could later be populated when configuring the interface. Ultimately it was decided to implement the interface creation subroutine to create a blank file as this is a more robust method than ignoring the error, where the throwing of that error could be indicative of some other issue.

5.5.4 Using cURL In PHP

An unexpected challenge when developing the server-side component of the web management interface was utilising PHP to make calls to external HTTP APIs. PHP offers a 'Client URL' library (PHP, n.d.-a) which provides capabilities to interact with external servers. However, the library is complex to use due to the number of statements required to make a simple GET request to a remote server. To combat this complexity a wrapper was developed which, when provided appropriate inputs, makes the call to the remote server and return the HTTP Response Code as well as a decoded JSON body. Through abstracting these functions into a common file - it was possible to use them across all of the server-side 'routes'.

5.5.5 Deploying Template VM

A key feature of the application is its ability to automatically generate a new node VM. PVE supports the cloning of a template VM to create a new VM, however this includes all the configuration used in the template VM when converted to a template (Proxmox, n.d.-f). This presented the challenge of needing to reconfigure the 'management' IP address of the cloned VM, which is not possible using the VM Control Application as changing the IP address would interrupt the HTTP connection to the web management interface, therefore returning an error; or using

the PVE Guest Exec options as this would require a reboot of the VM which adds significant time to the deployment of a new node.

To circumvent these issues - the Cloud-Init feature was explored (Proxmox, n.d.-a). This provides the functionality wherein the host can define parameters of a VM which get applied to the VM on the their first boot after cloning. In this use case, it was possible to use Cloud-Init to configure the IP address of the 'management' interface of the VM. Moving to use the Cloud-Init Debian 13 image did present an additional challenge as this uses the NetPlan network manager while VM Control Application had already been developed to work with the ifupdown network manager (Debian, n.d.). This issue was overcome through developing a Bash script which reconfigures the network manager as part of preparing the template VM image.

5.5.6 Time Constraints

The biggest challenge experienced during the development stage of this project was the underestimation of how long some stages would take to complete. The biggest example of this was hand-provisioning the test environments consisting of a number of VMs, each being deployed from an ISO image with custom configuration applied manually to each. It was common for two or three environments to be needed for each test scenario, to fully understand the way in which the pre-existing software works.

Due to these time constraints, it was decided to de-prioritise the implementation of the switch node type (FR03-2), to focus on meeting the other requirements and objectives of this project.

5.6 Summary

In summary, the journey undertaken to develop the application was not smooth or straightforward due to the issues recounted above. The application will next be tested to ascertain how many of its requirements it meets.

Chapter 6

Testing

6.1 Introduction

The testing for this project was a two-fold process. During the development stages, testing was carried out to identify if discrete components of the application were working during and after development or modification. After completion of the development stage, the application was tested holistically to identify if it met the requirements identified in chapter 3 (p15).

It was explored to complete automated unit testing, which would have been technically possible with Dart (Dart Team, 2025), PHP (PHP, n.d.-b), and JavaScript (MDN, 2026) all supporting automated testing; however due to the highly-interconnected nature of the discrete applications as well as the use of the PVE API, it would not have been possible to configure a unit testing environment. To test the front end of the application, both Puppeteer (Google, n.d.-b) and Selenium (Selenium, 2024) were explored, however both discounted due time constraints with there not being enough time to understand how the applications work and how to get effective results from them.

6.2 Testing During Development

During the development process, it was important to test frequently to ensure that both discrete components functioned as intended, and that the wider system functioned after modifications to individual components. These tests were unstructured and ad-hoc, as this best suited the rapid development methodology.

Where tests failed, or components didn't work as intended - debugging would be carried out. This first involved a visual review of the code to check for typos, incorrect variable assignment, or other syntax errors. After that, 'dump outputs' would be used to identify where the application got to before failing. For the Web Management Interface, this would commonly take the form of an output to the browsers console, an example of which can be seen in Figure 6.1 (p44); for the VM Control Application, this would either be printed textual responses to the console of the VM or API responses viewed using Postman. Figure 6.2 (p44) shows the output of one such debugging test, looking at receiving HTTP requests containing JSON arrays and parsing them.

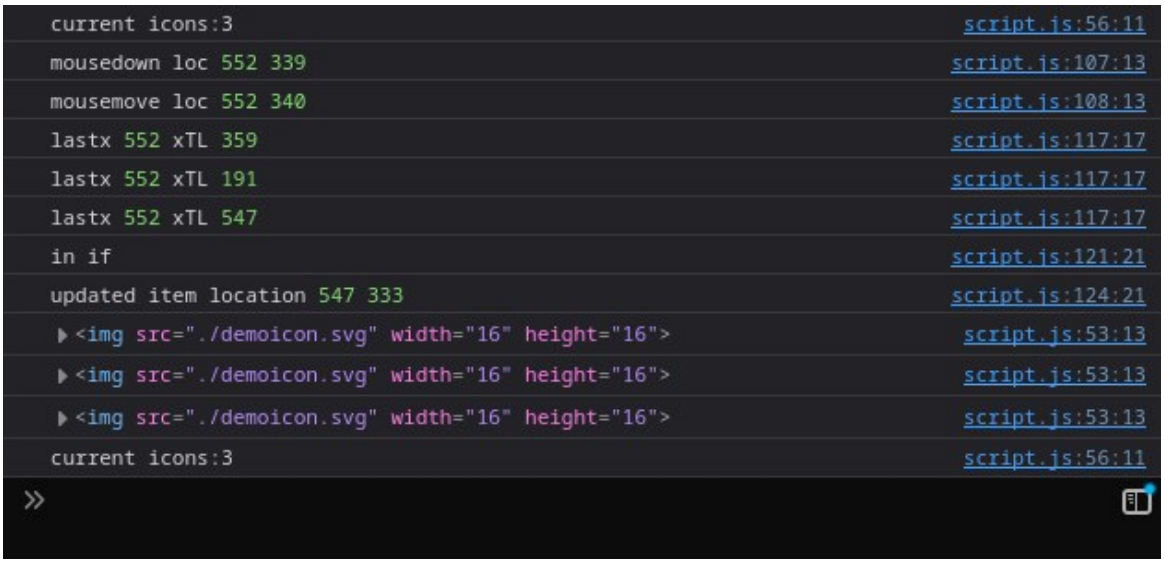


Figure 6.1: Browser console output during testing node positioning

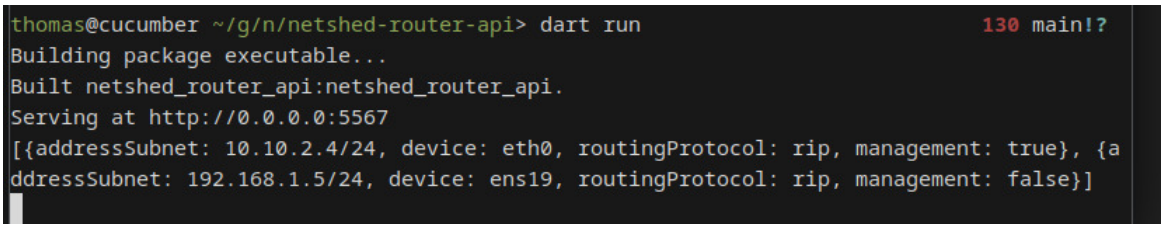


Figure 6.2: Console output from VM control application during testing

Another common debugging strategy was to consult pre-existing forum posts such as on Stack Overflow, PHP.net, or issues on GitHub projects as well as specific documentation. This, along with Google searching resolved all bugs.

In some cases generative artificial intelligence (AI) was used as a research tool. Prompting *Google Gemini* to identify a resolution for a known bad section of code, or suggest options for resolving developmental process problems, worked in a small number of cases. More often than not, Gemini was able to identify the issue and suggest a corrective action. However this action was often incorrect, or of poor quality. Through using AI, it was possible to identify a possible type of solution which could then be explored using a more traditional search engine, such as Google or Stack Overflow.

6.3 Final Validation Testing

To support the final validation of the application, a series of 25 test cases were developed. These cases represent an action to be taken in the application where there are multiple possible outcomes and a single expected outcome. The test cases can be found in Table G.1 (p90).

To validate the overall functionality of the application an end-to-end test was utilised combining multiple different test cases together. This test aims to simulate a real-world use of the

application with the following scenario.

Create a network with four router-type nodes connected R_1–R_2–R_3–R_4 with relevant configured interfaces. R_1–R_2–R_3 should use OSPF and R_3–R_4 should use the RIP routing protocol. It should be possible to (a) ping from R_1 to R_4 and (b) confirm in routing tables that R_1's subnetwork is being advertised to R_4.

Note that FR03-2 had been de-prioritised, and not completed, so has not been included in this test, all other requirements are to be tested regardless of known implementation state.

The network diagram used for this scenario can be found in Figure G.1 (p89), and the test plan for this scenario can be found in Table G.2 (p97) which includes each step's outcome. This test was mostly successful, with 89 steps passing, 19 partially passing and 2 failing out of a total of 110.

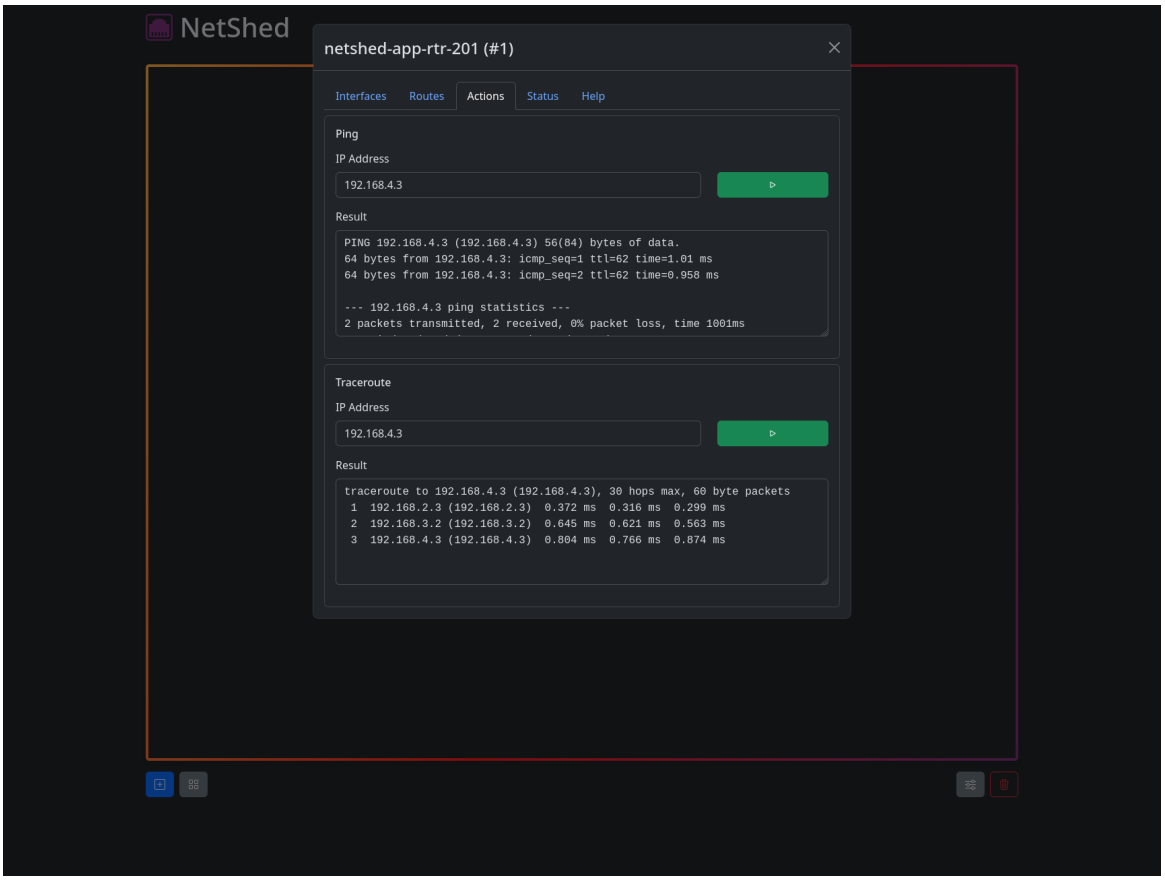


Figure 6.3: Application UI showing a successful ping and traceroute to 192.168.4.3

Figure 6.3 (p45) shows the final outcome of the test (a), with it being possible to ping from R_1 (indicated by the title of the modal) to R_4 through R_2 and R_3 as indicated by the Traceroute command shown in the lower half of the modal.

Figure 6.4 (p46) shows the final outcome of test (b), with R_4 routing tables showing the three subnetworks used in this example (192.168.2.0/24, 192.168.3.0/24, and 192.168.4.0/24) as well as the RIP routing table showing the metric and next hops which demonstrates that the RIP

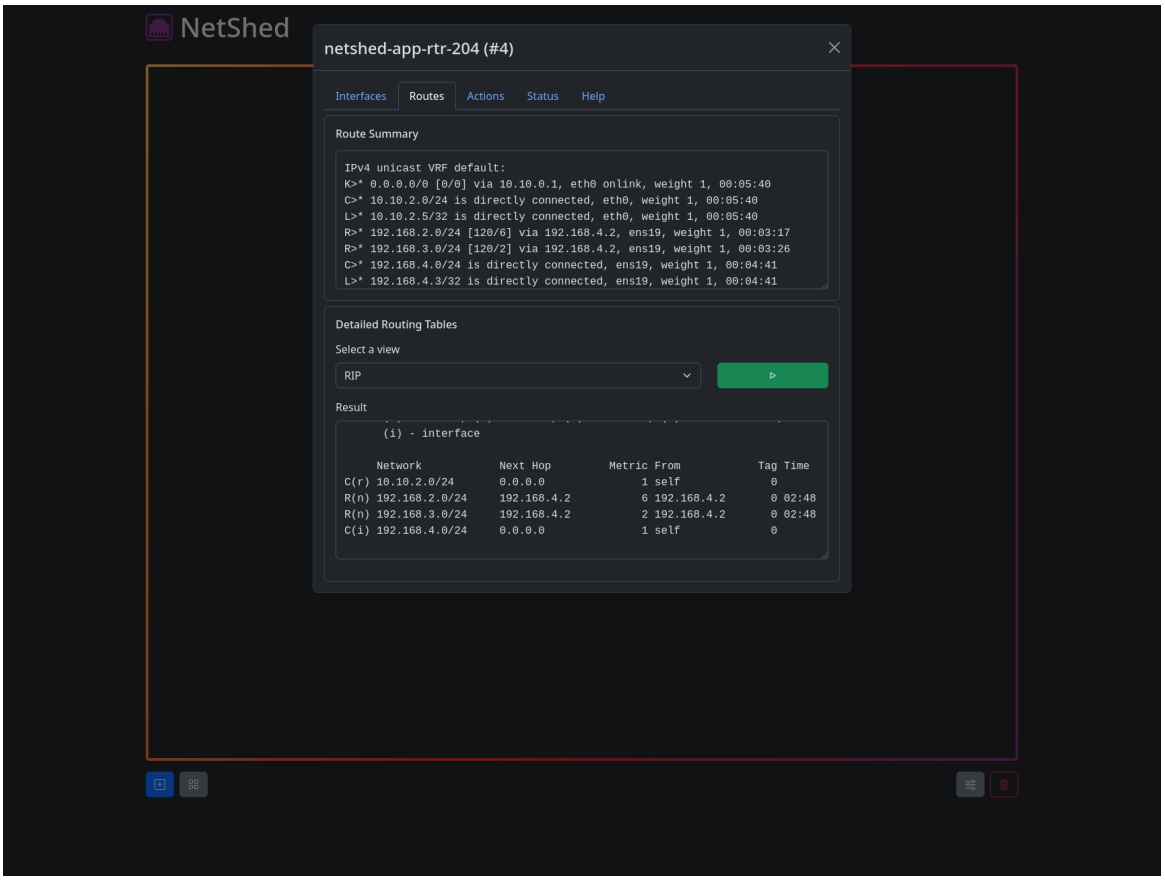


Figure 6.4: Application UI showing R_4s route summary and RIP routing table

algorithm is working as intended.

6.4 Limitations of Testing

The testing conducted as part of this report had a significant limitation in that it has all been conducted by the developer of the project. This means that the usability aspects are unable to be tested properly as the developer understands the inner workings of the projects.

Through the use of end-to-end testing, as has been completed here, it was only possible to ascertain a picture of the system as a whole. It was not possible to identify how individual discrete components of the system behave when they are provided with bad inputs. For example, step 4-6 of the final validation testing tests entering an incorrect IP address; which is caught by the web management interface client-side. It has not been tested as to what happens if a bad IP is supplied to the 'inner' components of the application. This could have been tested through the use of unit testing, however due to the limitations as discussed in section 6.1 (p43), it was not possible to use unit testing.

Furthermore, the testing conducted here only used four nodes. This is a limiting factor because this isn't very strenuous on the system. However, an increase in the nodes would be handled by the hardware running the application, rather than the application itself.

6.5 Summary

In summary, after exploring different methods of testing which could be used to test the application - end-to-end testing was used. The tests yielded an acceptable pass-rate of 81% with all the failures being minor bugs.

Chapter 7

Conclusion

7.1 Introduction

This chapter concludes the project; First it validates the requirements, then provides insight into the limitations and scope for future work. After this, an evaluation into the project management was conducted before finally space for personal reflections about the experience of completing this project.

7.2 Validation of Requirements

The requirements, defined in chapter 3 (p15), have been validated against the acceptance criteria found in Table 3.1 (p18) and Table 3.2 (p21) using using the testing steps found in section 6.3 (p44). The outcome from this validation can be found in Table 7.1 (p50).

Of the 13 functional requirements, 11 passed and 2 failed. Both of the failed requirements were P2, and both failed due to time constraints as discussed in subsection 5.5.6 (p41). Of the 2 non-functional requirements, 1 passed and 1 failed. The failed requirement failed in part due to time constraints, but also to inexperience implementing WCAG requirements.

7.3 Reviewing Aims & Objectives

The aims and objectives for this project were defined in section 1.2 (p1). The aim of this project was mostly met, with the specific educational components being the only elements not met. Otherwise, the remainder of the aim statement was achieved.

A similar result was seen for the objectives, wherein the majority were achieved, except for 4b which references the educational advice. This has been partially achieved through the validation of the input fields, except it does not provide commentary to the user on the how and why their input is incorrect.

These aims and objectives suited this project well and provided guidance as to the scope of the project, through which it was possible to avoid adding new features and risking more core features not being implemented.

Table 7.1: Requirement validation

ID	Test Action	Outcome	FVT Reference	Pass / Fail
FR01-1 (P1)	Add interface to a VM using web management interface	Interface is added to the VM; user provided positive feedback	2-8	Pass
FR01-2 (P1)	Configure interface on a VM using web management interface	IP address is configured on the VM; user provided positive feedback	2-9	Pass
FR01-3 (P2)	Enter an incorrect IP address, system should inform user why it's incorrect	System returns simple 'Incorrect IP' error with no information as to how it's wrong	4-6	Fail
FR01-4 (P1)	Add a virtual router to the environment	Virtual router added to the environment; user provided positive feedback	3-1	Pass
FR02-1 (P2)	There exists a network diagram which represents the live network	Web management interface contains a customisable network diagram	12-1	Pass
FR02-2 (P1)	Operate the app using only the web management interface	App is entirely operable using the web management interface	all	Pass
FR03-1 (P1)	Add and configure a Virtual Router to the virtual network	It is possible to add and configure a virtual router	2-1	Pass
FR03-2 (P2)	Add and configure a virtual network switch to the network	Virtual Switch has not been implemented		Fail

continued on next page

Table 7.1: Requirement validation (continued)

ID	Test Action	Outcome	FVT Reference	Pass / Fail
FR04-1 (P2)	Ping from one virtual node to another	It is possible to ping from one virtual node to another	5-4	Pass
FR04-2 (P3)	View the outcome of a Ping; monitor VM network incoming / outgoing traffic values	After performing a ping - the output is displayed; the total amount of traffic in and out of a VM is visible	2-6	Pass
FR05-1 (P1)	Select the RIP routing protocol for an interface of a router	The virtual router configures itself to use RIP for advertising and learning routes on the specified port	8-12	Pass
FR05-2 (P2)	Observe routing table	Web management interface provides ability to view the routing table	5-6	Pass
FR05-3 (P2)	Run traceroute from one VM to another	Traceroute is run and result returned displayed	5-5	Pass
NR01 (P1)	Run Web Accessibility Evaluation Tool	5 errors found	Figure 7.1 (p52)	Fail
NR02 (P2)	Application should be easy to use	Application is easy to use and provides help information	1-2	Pass

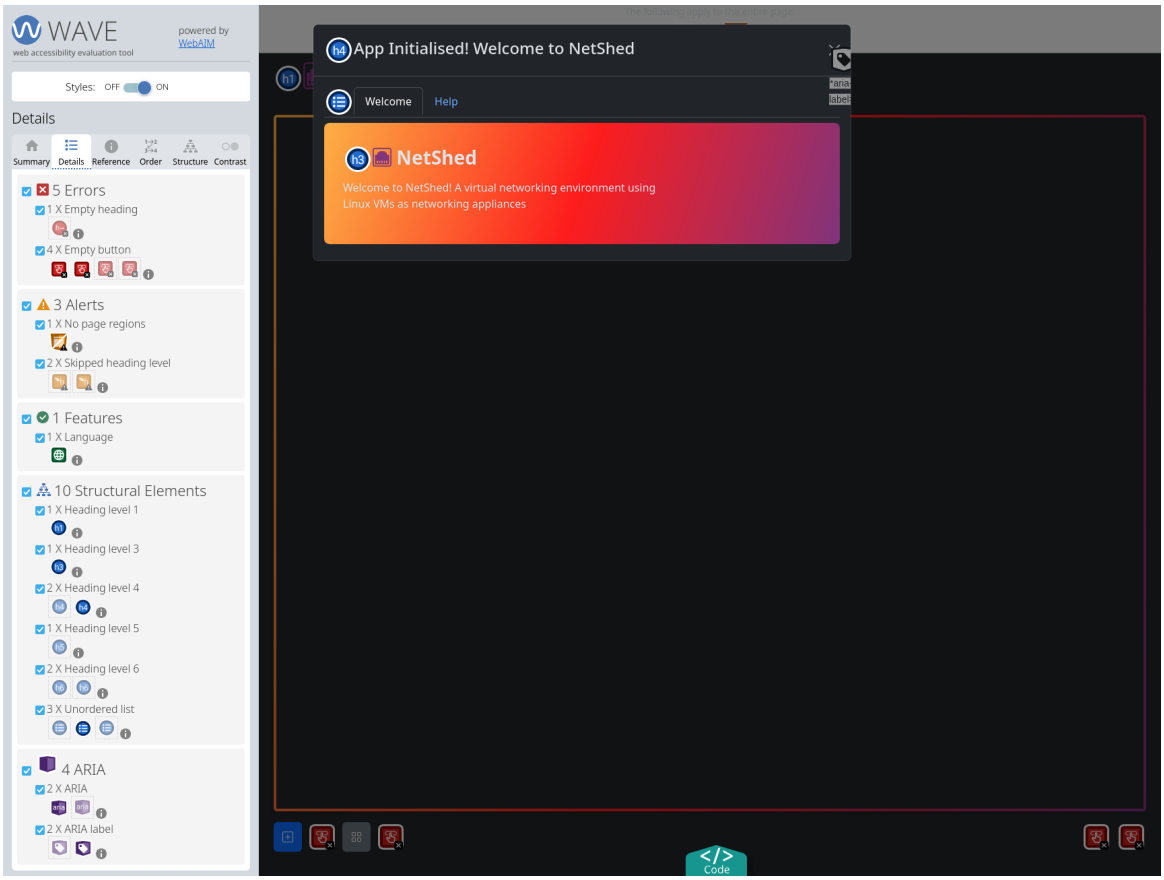


Figure 7.1: Result of WAVE tool

7.4 Known Limitations of the Solution

The biggest limitation of the solution is that switches were not implemented. While the application is functioning and can be used to simulate simple virtual router networks. Switches are hardware devices, the performance of which is very difficult to emulate even for enterprise level software packages such as Cisco Packet Tracer or GNS3. Although investigated, it was not possible to implement a switch appliance during this project.

This solution also requires incredibly specific software to run, which makes it awkward to deploy and difficult to maintain. Better research at the project initiation and literature review stages would have identified some of the edge-case software uses then alternatives could have been sought. The biggest example of this being that the VM Control application was developed to use the ifupdown network manager not the NetPlan network manager, as discussed in subsection 5.5.5 (p40). In hindsight, further research may have negated these issues, so this is considered to be valuable knowledge gained for any future work.

7.5 Future Works

This project leaves space for future works to be conducted in this area. Any future work should firstly consider completing the requirements of this stage which were not met. After this, future work should look to develop more features of the application.

A possible future extension would be to implement the ability for the virtual network, using VMs, to extend to a real-world device. In practicality - this would look like a second external network interface being added to the hypervisor host which is then connected to a network interface on a second device. The second device would run the VM control application so it would then be able to act as a router / switch.

An alternative extension would be to explore the hosting of this project in the cloud, using Google Cloud or Amazon Web Services to host the VMs and their virtual private cloud options for the networks between the nodes. This, if combined with a controller application to bring up and dispose of environments, would make this software even more suitable for use in education settings.

Further testing should also be considered, including using real users to validate the usability of the application and automated unit testing as well as automated UI testing. The latter would support making the application more robust, which in turn supports its aim of being suitable for education settings.

7.6 Evaluation of Risk Management

Through the development of the risk register for this project, Table 1.1 (p3), many of the risks were understood and a mitigation planned. A risk which wasn't factored in was software packages not working as advertised, or details around them changing during the lifespan of the project; a key example of this being the issues surrounding the deploying of a template VM as discussed in subsection 5.5.5 (p40).

Another issue faced, however mitigated, was a failure of the SSD in the laptop being used for development. This resulted in the loss of a configuration file used in the development environment as well as the all software configuration for the development tools. This caused a delay to the project of approximately two weeks.

7.7 Evaluation of Project Management

The original Gantt chart produced during the initiation phase of this project, Appendix C (p77), broadly held true. A revised Gantt chart is available in Appendix D (p79). The changes seen between the original and actual plan came down to lack of experience of working on a project of this scale before, and therefore not knowing how long stages would take. This can be seen, for

example, in the technical design process where it was planned to take 6 weeks but ultimately took 9 weeks, which reflects the proof of concept testing conducted.

The use of the Rapid Application Development methodology worked very well for this project. Through the use of an iterative development structure, there was enough flexibility in the methodology to dynamically respond to the needs of the project when deciding what to implement; for example prototyping two different components in parallel to understand the way in which they would work together. As a result of using this model, it was possible to return to previously implemented code and refactor or clean-up, to implement a standard which emerged after its initial development. This allowed for quick prototyping and latter refinement which allowed for quick progress within the implementation of this project.

Overall the project management methods used within this project worked and importantly didn't add overhead to working on the project, such as complex sprint planning, as well as being flexible enough to fit around other commitments and changing priorities throughout the 14 month combined life-span of this project. Furthermore allowing for a week of contingency gave leeway which may have been required for a more significant setback (i.e. illness) during the project itself.

7.8 Conclusion

Through the experience of completing this project, I have gained a deeper understanding of an approach to developing discrete components within a multi-part application, how to tackle a significant engineering project, and more about the individual tools and technologies used (i.e. Linux networking, PVE, Debian, PHP, and JavaScript to name a few).

A key learning point from this project, and something I would change if I was to be given another 12 months to re-do this project, is that a system like this requires a 'jobs queue.' This would allow actions which take some time (i.e. adding a new node to the network or adding a new interface) to be queued for asynchronous completion. This would enable the server component of the web management interface to return its HTTP response immediately; however this would require using a different language to PHP as it does not natively support asynchronous code. A server-side capable JavaScript framework or Dart with Shelf should be considered for this to fit in with the rest of the application.

In conclusion, this project has shown that it is possible to create a virtual router networking environment using Linux VMs acting as simulated routers; and has produced a working prototype application in this unconventional paradigm.

References

Allison, J. (2022a). *Packet-tracer-labs*. Retrieved March 31, 2026, from https://github.com/jrallison93/Packet-Tracer-Labs

Allison, J. (2022b). Simulation-based learning via cisco packet tracer to enhance the teaching of computer networks. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE, 1*, 68–74. https://doi.org/10.1145/3502718.3524739

Alpine Linux. (n.d.). *Software management*. Retrieved February 6, 2026, from https://wiki.alpinelinux.org/wiki/Software_management

Bergmann, S. (n.d.). *Php*. Retrieved January 2, 2026, from https://www.php.net

Bootstrap Team. (n.d.). *Bootstrap - the most popular html, css, and js library in the world.* Retrieved December 29, 2025, from https://getbootstrap.com/

Broadcom. (n.d.-a). *Fusion and workstation*. Retrieved January 4, 2026, from https://www.vmware.com/products/desktop-hypervisor/workstation-and-fusion

Broadcom. (n.d.-b). *Vmware esxi 8.0 update 3e now available as a free hypervisor*. Retrieved January 3, 2026, from https://knowledge.broadcom.com/external/article/399823/vmware-esxi-80-update-3e-now-available-a.html

Broadcom. (n.d.-c). *Vmware vsphere*. Retrieved January 4, 2026, from https://www.vmware.com/products/cloud-infrastructure/vsphere

Cisco. (n.d.). *Network topology icons - doing business with cisco - cisco*. Retrieved April 22, 2026, from https://www.cisco.com/c/en/us/about/brand-center/network-topology-icons.html

Cisco Networking Academy. (n.d.-a). *Cisco packet tracer | cisco packet tracer*. Retrieved December 29, 2025, from https://www.netacad.com/cisco-packet-tracer

Cisco Networking Academy. (n.d.-b). *Packet tracer - learn to use packet tracer*. Retrieved March 31, 2026, from https://contenthub.netacad.com/legacy/NetEss/1.0/en/course/files/3.5.2.4%20Packet%20Tracer%20-%20Learn%20to%20Use%20Packet%20Tracer.pdf

Cisco Networking Academy. (n.d.-c). *Teach with us | join over 29,000 passionate educators*. Retrieved December 29, 2025, from https://www.netacad.com/educators

Cisco Networking Academy. (2024). *Practice tools overview*. Retrieved December 29, 2025, from https://prod-tf-ui.s3.amazonaws.com/s/ff9e491c-49be-4734-803e-a79e6e83dab1/resource/file/50e2ddc0-7e9a-429f-9d99-a1d4c3558194/v1/en-US/practice_tools_overview_-_aug_2024.pdf?response-cache-control=no-cache&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20251229T121916Z&X-Amz-SignedHeaders=host&X-Amz-Credential=AKIAUJZRIEYBD5LJIEBJ%2F20251229%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Expires=30&X-Amz-Signature=f8413cea0abbcd43ce7ffe3e0c3b3e954fc5b

Crelin, J. (2023). Rapid application development (rad). https://www.ebsco.com/research-starters/computer-science/rapid-application-development-rad

Dart Team. (n.d.-a). *Dart programming language*. Retrieved May 23, 2026, from https://dart.dev

Dart Team. (n.d.-b). *Shelf | dart package*. Retrieved April 23, 2026, from https://pub.dev/packages/shelf

Dart Team. (2025). *Dart testing*. Retrieved April 26, 2026, from https://dart.dev/tools/testing

Debian. (n.d.). *5.4 the modern network configuration for cloud*. Retrieved April 8, 2026, from https://www.debian.org/doc/manuals/debian-reference/ch05.en.html#_the_modern_network_configuration_for_cloud

Dordevic, B., Timcenko, V., Kraljevic, N., & Jovicic, N. (2022). Performance comparison of kvm and proxmox type-1 hypervisors. *2022 30th Telecommunications Forum, TELFOR 2022 - Proceedings*. https://doi.org/10.1109/TELFOR56187.2022.9983666

Dumistracel, E., Istrate, L., & Radovici, A. (2025). Automation and monitoring of virtualized infrastructure on proxmox servers. *Proceedings - RoEduNet IEEE International Conference*. https://doi.org/10.1109/RoEduNet68395.2025.11208397

Dzerkals, U. (2025). *Eve-ng ce community edition cookbook*. Retrieved December 29, 2025, from https://www.eve-ng.net/wp-content/uploads/2025/04/EVE-CE-BOOK-6.3-2024.pdf

ECMA International. (2017). *The json data interchange syntax* (Standard No. ECMA-404:2017). ECMA International. Geneva, Switzerland. https://ecma-international.org/wp-content/uploads/ECMA-404_2nd_edition_december_2017.pdf

EVE-NG. (n.d.-a). *Eve-ng home page*. Retrieved December 29, 2025, from https://www.eve-ng.net/

EVE-NG. (n.d.-b). *Supported virtualization software and hardware for eve-ng*. Retrieved December 29, 2025, from https://www.eve-ng.net/index.php/supported-hardware-and-software-systems/

Everett, G. D., & McLeod, R. (2007). *Software testing testing across the entire software development life cycle*. Wiley-IEEE Press. https://ieeexplore.ieee.org/servlet/opac?bknumber=5201507

Fall, K., & Varadhan, K. (2011). *The ns manual*. Retrieved January 1, 2026, from https://www.isi.edu/websites/nsnam/ns/doc/ns_doc.pdf

Ford, J., Arnold, D., & Saniie, J. (2023). Environment provisioning and management for cybersecurity education. *IEEE International Conference on Electro Information Technology*, *2023-May*, 368–372. https://doi.org/10.1109/eIT57321.2023.10187365

FRRouting Project. (n.d.). *Frrouting project*. Retrieved April 29, 2026, from https://frrouting.org/

Fujimoto, R., Perumalla, K., & Riley, G. (2022). *Network simulation*. Springer International Publishing. https://books.google.co.uk/books?id=bYNyEAAAQBAJ

GNS3. (n.d.). *Gns3 home page*. Retrieved January 1, 2026, from https://gns3.com/

Google. (n.d.-a). *Material design 3 - google's latest open source design system*. Retrieved April 7, 2026, from https://m3.material.io/

Google. (n.d.-b). *Puppeteer*. Retrieved April 26, 2026, from https://pptr.dev/

Google. (2025). *Dart documentation*. Retrieved February 5, 2026, from https://dart.dev/docs

Harahus, M., Čavojský, M., Bugár, G., & Pleva, M. (2023). Interactive network learning: An assessment of eve-ng platform in educational settings. *Acta Electrotechnica et Informatica*, *23*, 3–9. https://doi.org/10.2478/aei-2023-0011

Helali, S. (2020). Simulating network architectures with gns3. In *Systems and network infrastructure integration: Design, implementation, safety and supervision* (pp. 9–25). Wiley. https://doi.org/10.1002/9781119779964.ch2

Jakob Nielsen. (n.d.). *10 usability heuristics for user interface design*. Retrieved February 2, 2026, from https://www.nngroup.com/articles/ten-usability-heuristics/

Janitor, J., Jakab, F., & Kniewald, K. (2010). Visual learning tools for teaching/learning computer networks: Cisco networking academy and packet tracer. *6th International Conference on Networking and Services, ICNS 2010, Includes LMPCNA 2010; INTENSIVE 2010*, 351–355. https://doi.org/10.1109/ICNS.2010.55

Kizilirmak, R. C., Kassayev, A., & Ukaegbu, I. A. (2023). A comparative study of simulated and hands-on experiments in teaching computer networks laboratory course. *IEEE Global Engineering Education Conference, EDUCON, 2023-May*. https://doi.org/10.1109/EDUCON54358.2023.10125188

Ma, T., Luo, L., Yu, H., Ma, C., Xie, Y., Sun, G., Wei, T., Chen, L., Laboratory, Z., Xu, Y., Zhang, N., Chen, X., & Xie, J. (2024). Klonet: An easy-to-use and scalable platform for computer networks education. *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation*. https://www.usenix.org/conference/nsdi24/presentation/ma

MDN. (n.d.). *Canvas api*. Retrieved February 6, 2026, from https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API

MDN. (2026). *Introduction to automated testing*. Retrieved April 26, 2026, from https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Testing/Automated_testing

Microsot Learn. (n.d.). *Install hyper-v*. Retrieved January 3, 2026, from https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/get-started/install-hyper-v

Mininet. (n.d.). *Mininet*. Retrieved January 1, 2026, from https://mininet.org/

National Cyber Security Centre. (2025). *It's time to act - ncsc annual review 2025* (tech. rep.). National Cyber Security Centre. https://www.ncsc.gov.uk/files/ncsc-annual-review-2025.pdf

Neumann, J. C. (2015). *The book of gns3*. No Starch Press, Incorporated. https://ebookcentral.proquest.com/lib/portsmouth-ebooks/detail.action?docID=4503137

NS-2. (n.d.). *The network simulator - ns-2*. Retrieved January 1, 2026, from https://www.isi.edu/websites/nsnam/ns/

Nurdiana, D., Hasanah, S. H., & Maulana, R. (2024). College students' perceptions toward usability of simulator application as a form of virtual experiment at the distance learning. *International Journal of Global Operations Research, 5*, 22–29. http://www.iorajournal.org/index.php/ijgor/index

Oracle VirtualBox. (n.d.). *Virtualbox*. Retrieved January 4, 2026, from https://www.virtualbox.org/

Oxford English Dictionary. (n.d.-a). Emulation, n., 5. Oxford University Press. Retrieved January 1, 2026, from https://doi.org/10.1093/OED/6599666254

Oxford English Dictionary. (n.d.-b). Hypervisor, n. Oxford University Press. Retrieved January 3, 2026, from https://doi.org/10.1093/OED/1091081860

Oxford English Dictionary. (n.d.-c). Simulator program, in simulator, n., 2.c. Oxford University Press. Retrieved January 1, 2026, from https://doi.org/10.1093/OED/5520434336

pfSense. (n.d.). *Pfsense homepage*. Retrieved January 4, 2026, from https://www.pfsense.org/

PHP. (n.d.-a). *Client url library*. Retrieved April 8, 2026, from https://www.php.net/manual/en/book.curl.php

PHP. (n.d.-b). *Phpunit: The testing framework for php*. Retrieved April 26, 2026, from https://phpunit.de/index.html

Pico CSS. (n.d.). *Pico css - minimal css framework for semantic html*. Retrieved April 7, 2026, from https://picocss.com/

Portnoy, M. (2016). *Virtualization essentials*. John Wiley & Sons, Incorporated.

Postman. (n.d.). *Postman api platform*. Retrieved April 8, 2026, from https://www.postman.com/product/

Proxmox. (n.d.-a). *Cloud-init support*. Retrieved April 8, 2026, from https://pve.proxmox.com/wiki/Cloud-Init_Support

Proxmox. (n.d.-b). *Features*. Retrieved January 3, 2026, from https://www.proxmox.com/en/products/proxmox-virtual-environment/features

Proxmox. (n.d.-c). *Proxmox ve api*. Retrieved January 4, 2026, from https://pve.proxmox.com/wiki/Proxmox_VE_API#PHP

Proxmox. (n.d.-d). *Proxmox ve api documentation*. Retrieved April 8, 2026, from https://pve.proxmox.com/pve-docs/api-viewer/

Proxmox. (n.d.-e). *Proxmox virtual environment*. Retrieved January 3, 2026, from https://www.proxmox.com/en/products/proxmox-virtual-environment/overview

Proxmox. (n.d.-f). *Vm templates and clones*. Retrieved April 9, 2026, from https://pve.proxmox.com/wiki/VM_Templates_and_Clones

Proxmoxer. (n.d.). *Welcome to proxmoxer*. Retrieved January 4, 2026, from https://proxmoxer.github.io/docs/latest/

Savu-Jivanov, A., Ancuţi, C., Stolojescu-Crişan, C., & Gal, J. (2025). Advanced virtual infrastructures for it education: An integrated proxmox and eve-ng approach. *Proceedings - RoEduNet IEEE International Conference*. https://doi.org/10.1109/RoEduNet68395.2025.11208365

Selenium. (2024). *Webdriver*. Retrieved April 26, 2026, from http://selenium.dev/documentation/webdriver/

Smera, C., & Sandeep, J. (2022). Networks simulation: Research based implementation using tools and approaches. *2022 IEEE 3rd Global Conference for Advancement in Technology, GCAT 2022*. https://doi.org/10.1109/GCAT55367.2022.9972119

Somerville, I. (2017). *Software engineering, 9th edition*. Pearson Education, Inc.

SQLite. (2026). *Sqlite home page*. Retrieved April 29, 2026, from https://sqlite.org/

TheOneWithTheBraid. (2020). *Build sdk with musl #40906*. Retrieved February 6, 2026, from https://github.com/dart-lang/sdk/issues/40906

VMWare. (n.d.). *What is network virtualization?* Retrieved January 1, 2026, from https://www.vmware.com/topics/network-virtualization

W3C. (2025). *Wcag 2 overview*. Retrieved April 7, 2026, from https://www.w3.org/WAI/standards-guidelines/wcag/

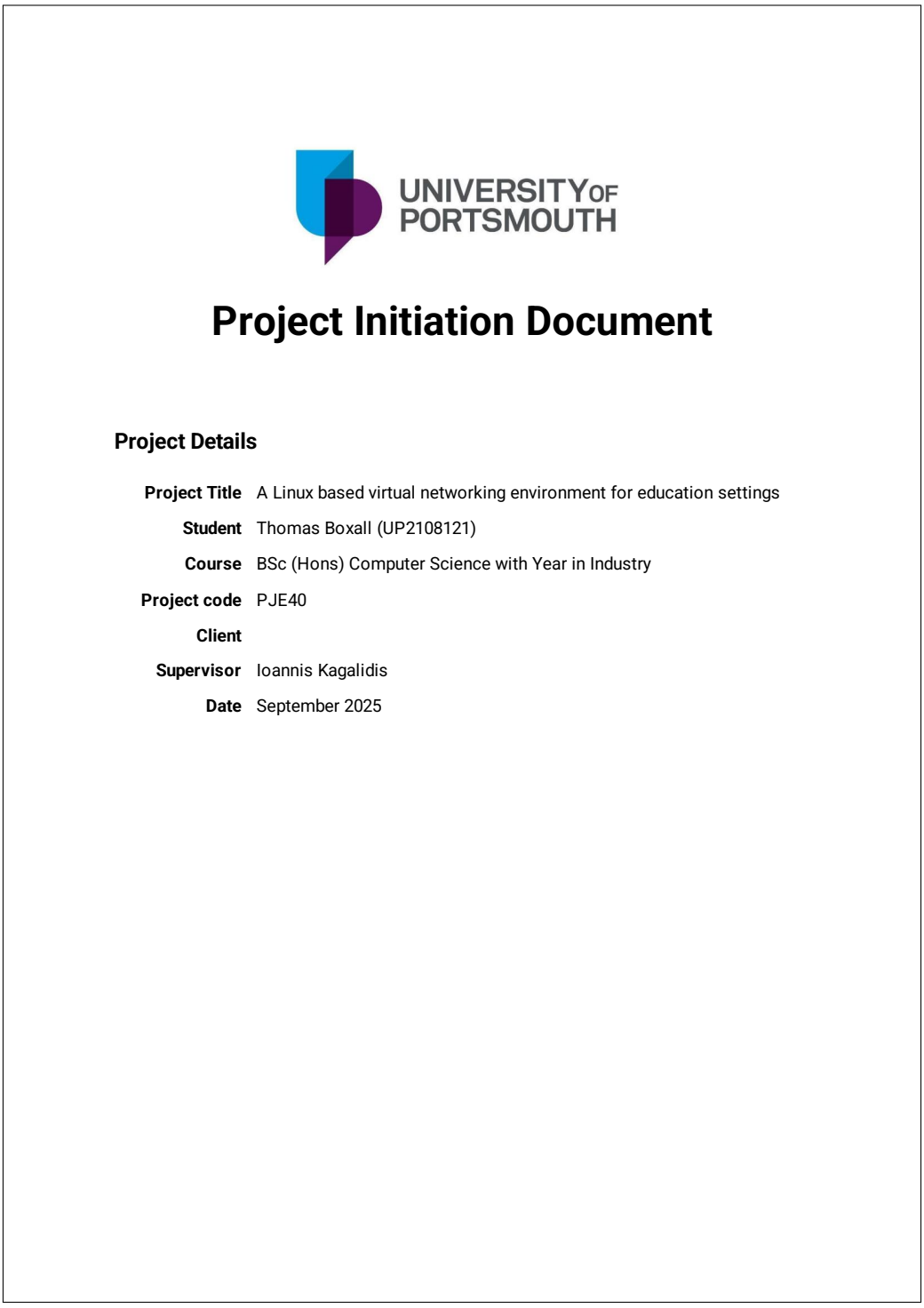
WebAIM. (n.d.). *Wave web accessibility evaluation tools*. Retrieved March 31, 2026, from https://wave.webaim.org/

YAML Language Development Team. (2021). *Yaml ain't markup language* (Standard No. 1.2.2 (2021-10-01)). YAML. https://yaml.org/spec/1.2.2/

Appendices

Appendix A

Project Initiation Document



UNIVERSITY OF PORTSMOUTH

Project Initiation Document

Project Details

Project Title	A Linux based virtual networking environment for education settings
Student	Thomas Boxall (UP2108121)
Course	BSc (Hons) Computer Science with Year in Industry
Project code	PJE40
Client	
Supervisor	Ioannis Kagalidis
Date	September 2025

1. Client / Target Audience

This project does not have a specific client. The broader target audience is Sixth Form or University Level Computer Networking Lecturers or Tutors.

2. Project background and problem to be solved

For Lecturers who are teaching basic Computer Networking, specifically: the fundamentals of IP Addressing; the purpose & functions of Gateways, Firewalls & Switches; and the design of basic networks, there are very few simple and easy ways to teach this. Lectures have the option of trying to teach this as a theoretical concept which students often struggle with because of the highly complex nature or attempting to use a Virtual Environment. The existing Virtual Environments are either costly or overly complicated to use for a simple virtual network, as we are describing in this case.

The solution which will be designed & implemented in this project is aimed Further Education & Higher Education academic staff who are looking for a Virtual Network Environment which provides the right level of functionality for this situation. There are contenders on the market, such as Graphical Network Simulator 3 (GNS3), or Cisco PacketTracer.

The existing solutions aren't designed for entry-level education. Cisco developed & maintains PacketTracer which it makes available for free as part of the Cisco Networking Academy. This software is a Network Virtualisation Environment which is designed for education settings. However, it goes above and beyond the basic networking which can be challenging to teach without the right tools. For example – all the devices within the software are adjusted implementations of real-world Cisco devices. This means that Academic staff will lose valuable teaching time to tell students what model Gateway, Switch or Firewall to use – rather than just to use one. Additionally, PacketTracer's User Interface does not provide full functionality for control of the networking appliances; therefore, users are expected to interact with the Cisco IOS command prompt which is another barrier to entry within the Networking field. Within the Open-Source Community, the software GNS3 is actively maintained. This works through virtualisation, which means that the networking devices are virtualised using real-world allocated hardware, as opposed to PacketTracer which uses simulation. The major flaw with GNS3 is the complexity to operate the software. We return to the idea that these software suites abstract very little from the user, which is great for those undertaking dedicated Networking qualifications, such as a Computer Networks Degree or equivalent; however, it poses a significant barrier to entry for those undertaking more junior or generalised qualifications.

Data published in 2023 shows that nearly one third of all 18-year-olds applying through UCAS applied to a Computing course. This figure is up on 9.5% from the previous year. The majority of these students will undertake some level of basic Networking education as part of their degree. This provides them with a well-rounded knowledge and can increase their job prospects. The lack of simple and easy to understand Virtual Networking Environments on the market currently, may lead to some students not being able to comprehend the complex topics and therefore not be able to fully realise their employment potential.

3. Project aim and objectives

The aim of this project is to create a new Virtual Networking Environment which is better suited for education settings.

The objectives for this project are as follows:

- To design and implement a virtual networking environment which uses Linux Virtual Machines to provide networking devices.
- To design and implement a web management interface for the virtual networking environment.
- To design and implement a single-touch provisioning option for the application such that it better suited for education settings.

4. Project constraints

There will be a number of constraints for this project:

- The deadline for the project is a fixed, immovable date.
- Budget for hardware & software is limited. This will restrict computation options which could be used in this project.
- Not infinite time available for this module between fixed commitments: other University studies & Employment and additional voluntary commitments.

5. Project management

I will manage my project in a combination of Predictive and Adaptive methods. As part of preparing this document, I have produced a timeline which will form the Predictive side of my management strategy. However, I will also be adaptive in that I won't be afraid to adjust the timeline as I'm progressing; not least because this is a long project which may hit many different challenges along the way.

The main way in which I will ensure progress is made is through setting goals for myself to meet – for example complete a Chapter of the Report; complete a piece of documentation or implement a feature. These micro-goals will then be decomposed into a series of tasks which need to be completed to enable the micro-goal to be completed. These micro-goals will be considerably less formal than anything set out in this document as they will often lead on from each other, and all fit into the bigger master timeline found in the next section.

I have experience Managing Projects using these strategies and find that they work well for me.

The Software Development stages of this project will follow the Rapid Application Development (RAD) Development Lifecycle. I have chosen this because it allows for iterations which can focus on getting a single feature to a working state. This is beneficial to my project as it allows me to focus on getting a feature working rather than having all my requirements for every feature fleshed out to begin with. Furthermore, it reduces some of the administration overhead which is needed for other methodologies – and given I am the only one working on this project, there is no need for many of these administrative burdens. My implementation of RAD will include things like the UI Design taking place during a sprint rather than during the design phase.

6. Tasks and timescales

The below table details the timeline for this project from initiation to submission. This data is also presented in a Gantt chart which can be found in Appendix A.

No	Stage	Dates	Main Tasks
1	Project Initiation	01/05/25 - 01/10/25	Choose Supervisor & Project 01/05/25 - 01/06/25 Write PID 01/07/25 - 01/10/25
2	Literature Review	22/09/25 - 19/01/26	Main Literature Review 22/09/25 - 20/10/25 Additional Literature Review 20/10/25 - 19/01/26
Milestone: Project Initiation Document Submission (01/10/25)			
3	Ethics	01/10/25 - 05/11/25	Write Ethics Application 01/10/25 - 05/11/25
4	Requirements Gathering	01/10/25 - 31/10/25	Gather Requirements 07/10/25 - 31/10/25
Milestone: Ethics Application Submission (05/11/25)			
5	Technical Architecture Design	20/10/25 - 24/11/25	Research Proof of Concept Testing Learning Technologies Where Needed Architecture Design
6	Implementation	01/12/25 - 20/04/26	Implementation On-The-Fly Design MVP production Testing
Milestone: Satisfactory Progress Presentation (08/12/25 - 12/12/25)			
7	Report Finalisation	20/04/26 - 27/04/26	
8	Contingency	27/04/26 - 06/05/26	
Milestone: Final Submission (06/05/25)			

7. Facilities and resources

This project will require the following resources:

- Me – to do the work;
- Supervisor – to supervise the work being completed;
- My laptop – for the work to be done on;
- Package Documentation – for Linux Packages used in the application;
- Programming Language Documentation;
- Standard Software Development Software (Git, VS Code, etc).

This project may also require the following resources:

- Additional Computing Power – if my laptop is not able to run enough VMs for development purposes. This would either come from additional Local Hardware or cloud VMs.

8. Project risks

This project has a number of risks. These have been outlined in the below table.

No	Description	Likelihood	Impact	Mitigation / Avoidance
1	Laptop Failure	Medium	Delay to project timeline, working data loss.	Monitor Laptop Health & backup work to cloud version control software.
2	Bug in External Technology (i.e. Virtualisation Software)	Low	Delay to project timeline, potential for re-architecting requirement.	Use well-known and well-used software running stable versions. Apply patches where needed.
3	Lack of Knowledge leading to poor decision	Medium	Poor decision must be rectified through increased workload.	Gather data before making decisions. Ensure all decisions are logged with reasons so this can be reviewed & changed if needed.
4	Online Account Compromise / Lockout	Low	Inability to access online account, for example GitHub or Cloud VM Provider.	Ensure appropriate 2FA is configured and recovery keys are stored suitably.

No	Description	Likelihood	Impact	Mitigation / Avoidance
5	Scope Creep	High	More work to be completed.	Before devoting time to designing / implementing / thinking about something – ensure it fits within the defined scope.
6	Unexpectedly Complicated Project	Medium	Work is harder so takes longer to figure out how to complete.	Research completed as part of Project Initiation to prove all elements are technically possible.
7	External Resource Changes Product so it is unusable	Medium; High if Google Used	Significant re-work to re-architect or find suitable alternative external resource.	Keep up to date with news for products being used and use stable products with no planned deprecation.
8	Illness / Incapacity to Work	Low	No work / low quantity of completed for period of time	Complete lots of while health is good. Plan contingency into Project Timeline

9. Project deliverables

This project will have a number of deliverables:

- The Virtual Networking Environment Application, including any supporting tools / micro-applications;
- The Project Report;
- User Guide for the Application;
- Entity Relationship Diagram for the application;
- Technical design notes for the application;
- Application Testing Strategies and / or logs.

10. Research

This project will require technical research in the following, not limited to, areas:

- Virtualisation Environments: to utilise a cloud provider or to host the VMs locally

- Linux Distribution: which distribution to use for the VMs which is able to run with minimal resource, be deployed from file and supports desired packages

- Linux Packages to be used for Routing, Switching and Firwewall functionality. Articles exist for Routing, Wildman 2022, and Switching, Liu 2018.

Research will also be required to obtain an understanding of what networking devices should be implemented as well as what software-level teaching aids (i.e. automatically assigning IP addresses) should be included to make this application suitable for a teaching environment. This data will come from a mix of existing research papers, teaching materials, and potentially from Academic Staff.

11. Legal, ethical, professional, social issues

Research Data
Any data gathered as part of the Requirements Gathering phase, whether this be through Interviews, informal conversations, or surveys will be securely stored on the Microsoft 365 OneDrive provided to me through the University of Portsmouth. This account is secured through 2 Factor Authentication which only I have access to.

Copyright Infringement
Given the nature that there are pre-existing solutions in this area of Virtual Networking Environments, there is an inherent risk of encroaching on Copyright Infringement of one of these solutions. This is to be mitigated by simply not replicating any pre-existing solutions like-for-like.

Security
Given the nature of the solution and the fact that there will be ways to remotely control Linux Virtual Machines, there is a need for security measures to ensure these Linux VMs are not compromised and repurposed by Threat Actors (as DDOS servers, for example). Security will be imposed through keeping the machines behind a firewall with no external access as well as through reviewing suspicious events in the access logs for the machines.

Personal Information
No personal information will be gathered during this project.

12. Supervisor meetings

Meetings will be held weekly in person.

13. Declarations

Please tick the following declarations:

Usage (optional)	
X	I give permission for this document to be made available to other students as examples of previous work.
Authenticity of work	
X	I confirm that I have read the University rules in respect of plagiarism and student misconduct.
X	I understand that if I use work from an external source, I must reference and cite the source in any work that I produce.
Ethics	
X	I understand that ethical approval will be needed for this project regardless of whether I carry out primary research or not.
X	I understand that if I do conduct primary research, I must include all raw data and research documentation in my final project report.
X	I understand that not including the raw data can mean potentially failing the project.
X	I understand that not obtaining ethical approval is grounds for academic misconduct and possible project failure.
Use of AI Artificial Intelligence (AI), such as ChatGPT, can be used as a tool to help assist and inform the initial development of your work but it is not a replacement for your own critical thinking and analysis. We require that your work is your own original content, demonstrating your knowledge, skills, and critical thinking abilities. To that end, AI-generated content must not be included in any work you submit for assessment unless suitably referenced. Not doing so is a breach of the University's Academic Regulations as outlined in the Student Conduct Policy and constitutes an assessment offence.	
X	I have read the University AI policy and if relevant, I will acknowledge the use of AI in any work I produce for this project with appropriate referencing.
X	I understand that not appropriately referencing the use of AI is an assessment offence equal to plagiarism and may result in a penalty which remains on my student record.

Name: Thomas Boxall

Date: 15/09/25

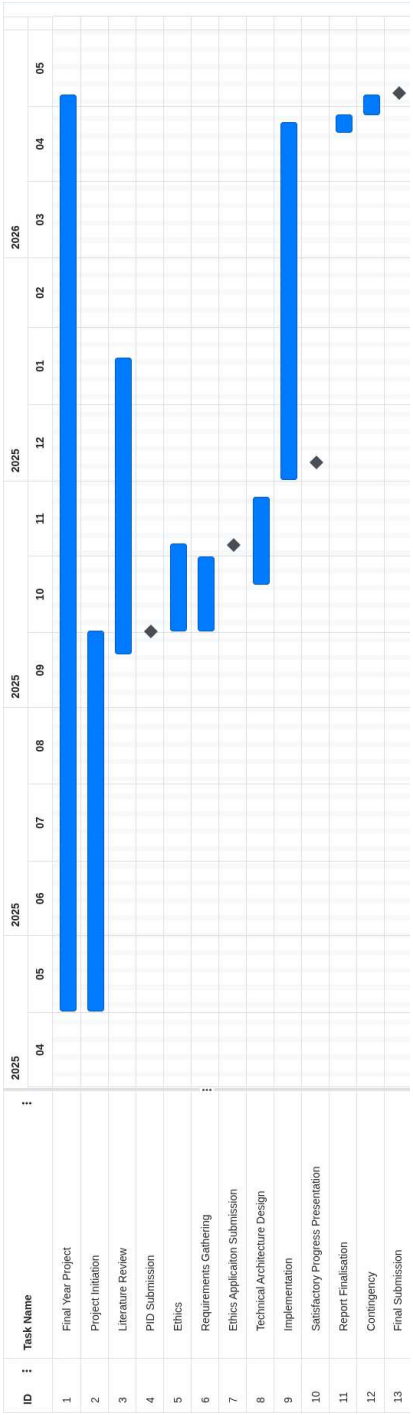
14. References

Liu, H. (2018, October 22). Introduction to Linux interfaces for virtual networking | Red Hat Developer. Red Hat Developer. https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking#bridge

UCAS (2023). UK 18-year-olds make record number of applications for computing courses. [online] UCAS. Available at: https://www.ucas.com/corporate/news-and-key-documents/news/uk-18-year-olds-make-record-number-applications-computing-courses.

Wildman, M. (2022, September 30). Configure Linux as a Router (IP Forwarding). Linode Guides & Tutorials; Linode. https://www.linode.com/docs/guides/linux-router-and-ip-forwarding/?tabs=iptables

Appendix A: Gantt chart



Appendix B

Ethics Certificate

Certificate of Ethics Review

Project title: A Linux VM Based Virtual Networking Environment for Education Settings

Name:	Thomas Boxall	**User ID**:	2108121	**Application date**:	03/11/2025 17:22:47	**ER Number**:	TETHIC-2025-112034

You must download your referral certificate, print a copy and keep it as a record of this review.

The FEC representative(s) for the **School of Computing** is/are **Elisavet Andrikopoulou, Kirsten Smith**

It is your responsibility to follow the University Code of Practice on Ethical Standards and any Department/School or professional guidelines in the conduct of your study including relevant guidelines regarding health and safety of researchers including the following:

- University Policy
- Safety on Geological Fieldwork

It is also your responsibility to follow University guidance on Data Protection Policy:

- General guidance for all data protection issues
- University Data Protection Policy

Which school/department do you belong to?: **School of Computing**
What is your primary role at the University?: **Undergraduate Student**
What is the name of the member of staff who is responsible for supervising your project?: **Ioannis Kagalidis**
Is the study likely to involve human subjects (observation) or participants?: No
Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?: No
Are there risks of significant damage to physical and/or ecological environmental features?: No
Are there risks of significant damage to features of historical or cultural heritage (e.g. impacts of study techniques, taking of samples)?: No
Does the project involve animals in any way?: No
Could the research outputs potentially be harmful to third parties?: No
Could your research/artefact be adapted and be misused?: No
Will your project or project deliverables be relevant to defence, the military, police or other security organisations and/or in addition, could it be used by others to threaten UK security?: No
Please read and confirm that you agree with the following statements: I confirm that I have considered the implications for data collection and use, taking into consideration legal requirements (UK GDPR, Data Protection Act 2018 etc.), I confirm that I have considered the impact of this work and and taken any reasonable action to mitigate potential misuse of the project outputs, I confirm that I will act ethically and honestly throughout this project

Supervisor Review

As supervisor, I will ensure that this work will be conducted in an ethical manner in line with the University Ethics Policy.

Supervisor comments:

Supervisor's Digital Signature: **ioannis.kagalidis@port.ac.uk** Date: **17/11/2025**

Appendix C

Original Gantt Chart

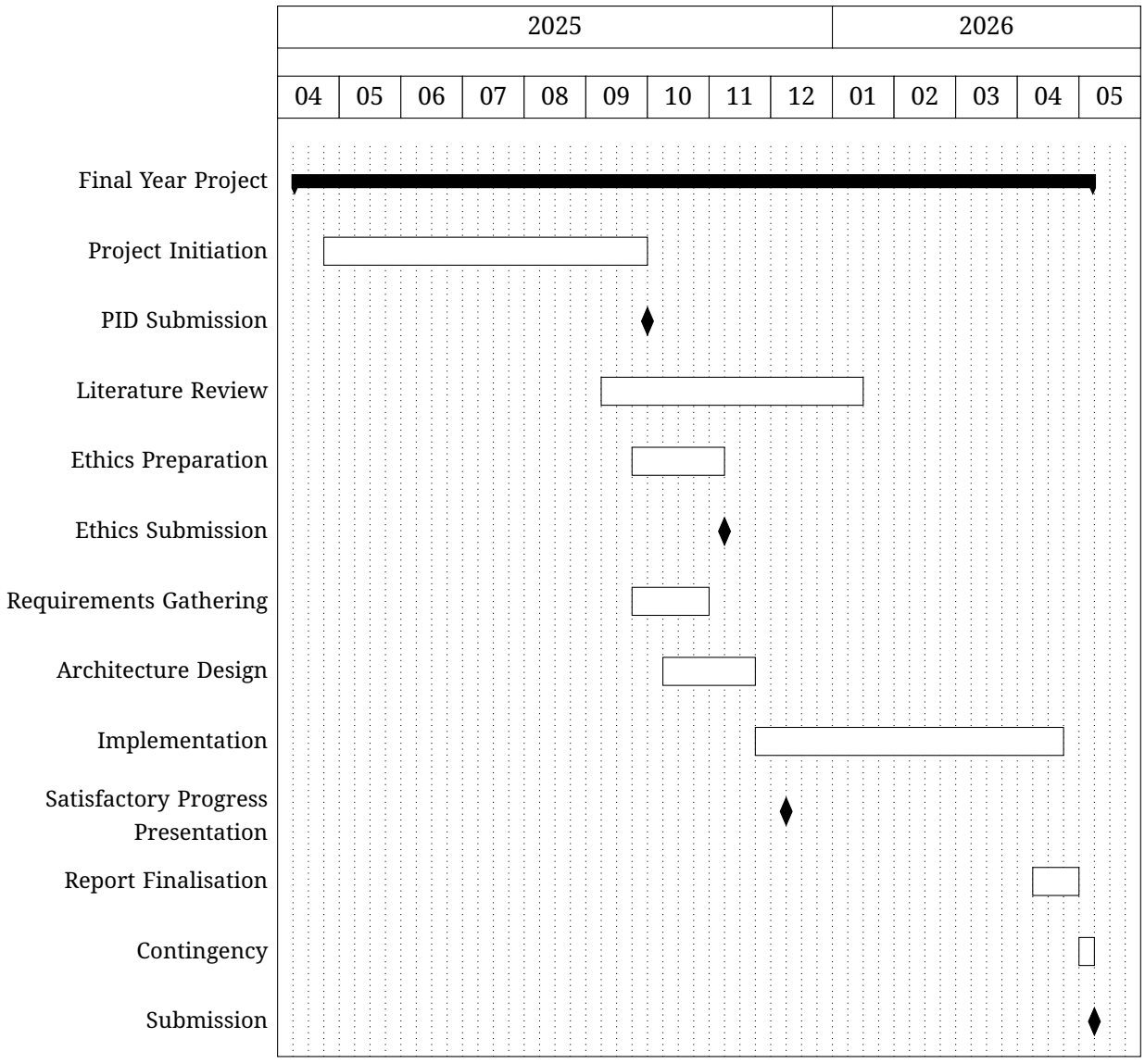


Figure C.1: Original Gantt Chart from project initiation document

Appendix D

Actual Gantt Chart

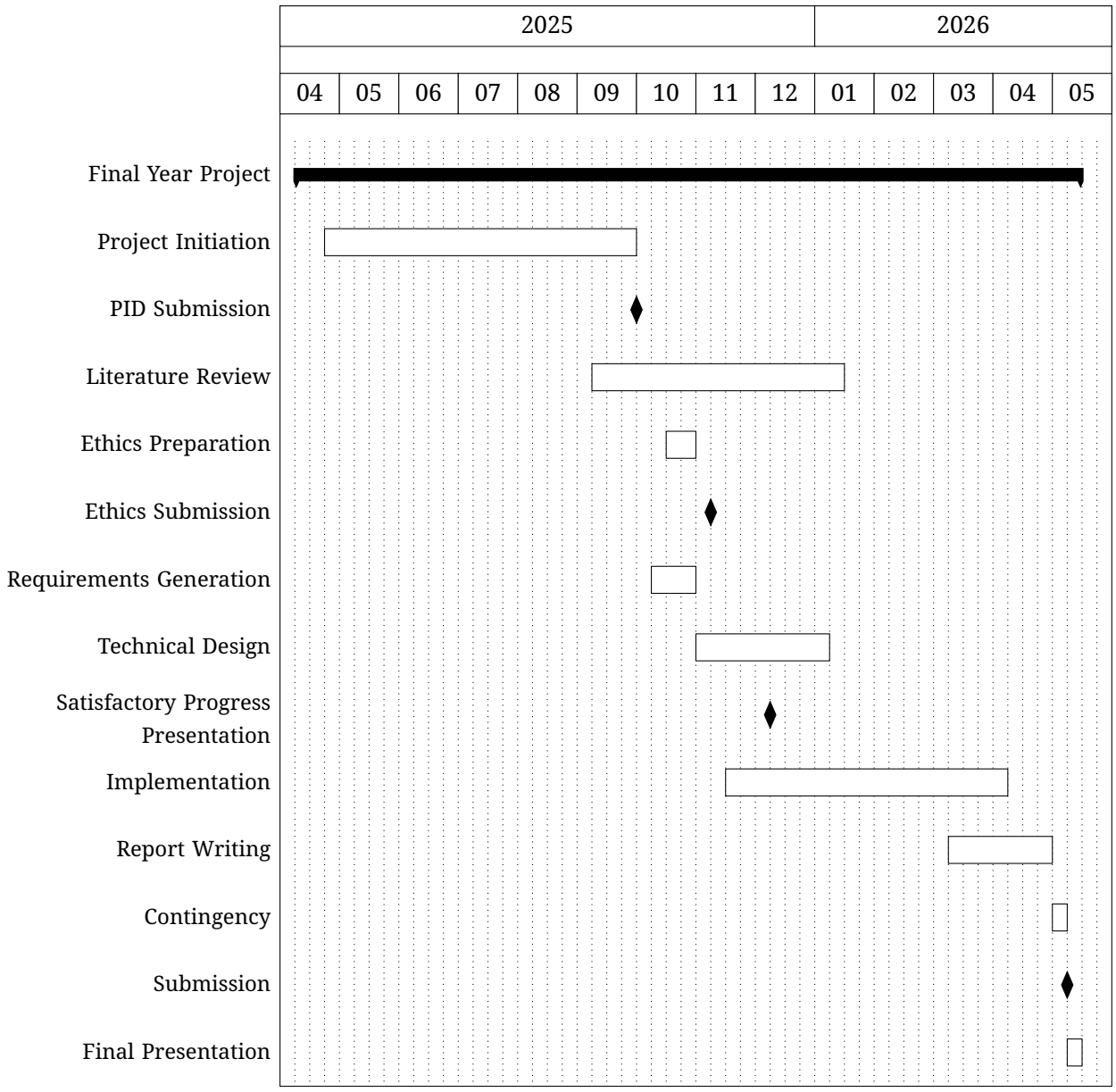


Figure D.1: Actual Gantt Chart, generated at end of project

Appendix E

Literature Review Search Data

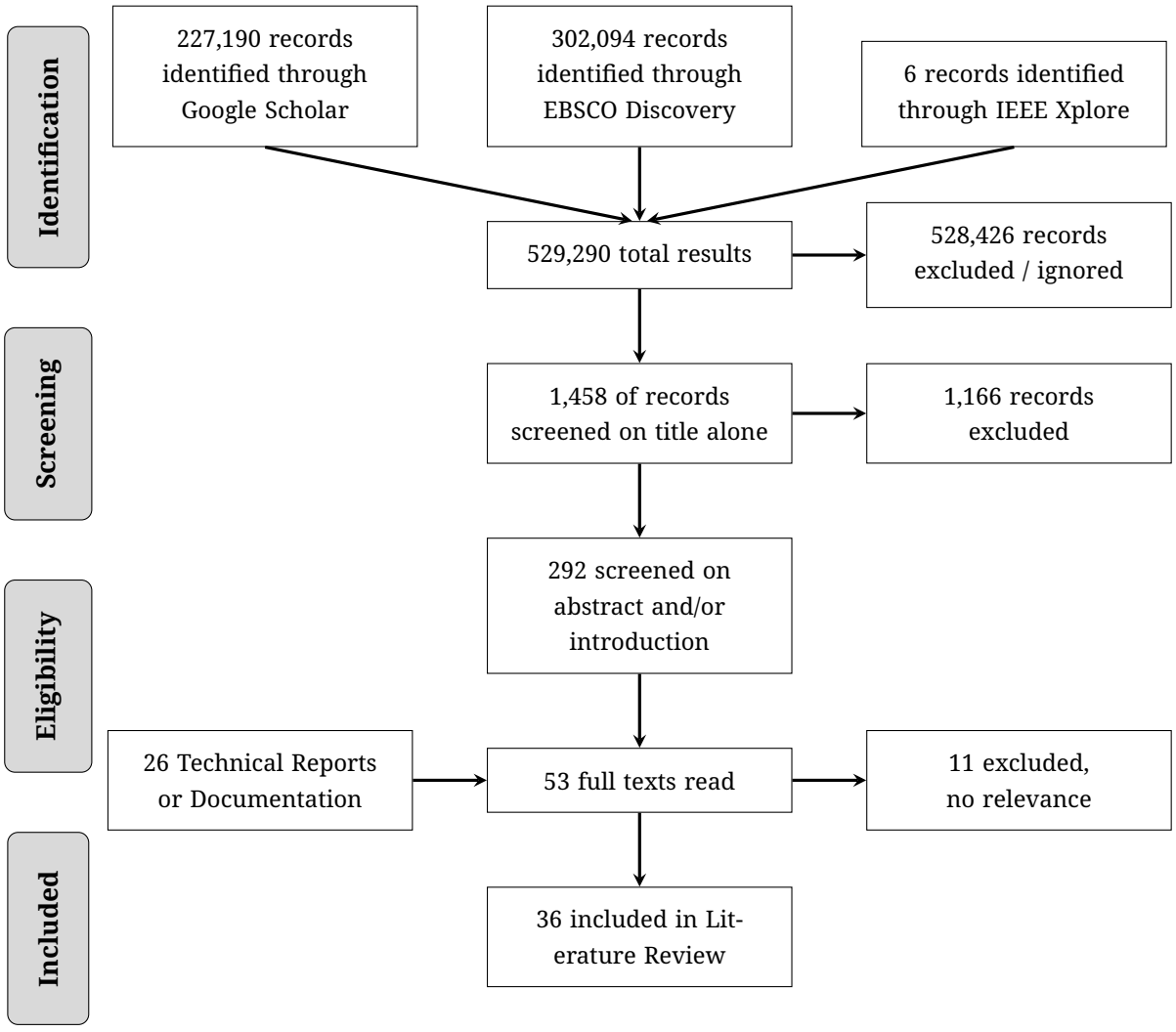


Figure E.1: PRISMA diagram showing literature review searches

Table E.1: Literature review searches

Search String	Search Engine	Criteria	Hits	Comments
software defined networking	Google Scholar	Published since 2020	18,000	Hard to discern relevant papers from volume returned; not much of direct relevance - mostly to do with DDoS within SDN
hypervisor network	Google Scholar	Published since 2020	17,500	Overwhelmed by volume of poor relevance paper
proxmox network	Google Scholar	Published since 2020	1,760	Lots of relevance
proxmox network*	EBSCO Discovery	Published in last 5 years	218	Some papers repeated from similar Google Scholar query. Not much new of relevance
proxmox network*	IEEE Xplore	Published between 2020 and 2025	6	1 of relevance already discovered from other site
proxmox AND network*	EBSCO Discovery	Published in the last 5 years and Full Text Online Only and Print	234	Lots of relevance; mostly IEEE papers
”computer network*” AND (emulation OR simulation)	EBSCO Discovery	Published in the last 5 years	77,065	A few of some relevance, lots of no relevance

continued on next page

Table E.1: Literature review searches (continued)

Search String	Search Engine	Criteria	Hits	Comments
computer network simulation	Google Scholar	Published since 2020	149,000	Not much of relevance, descends into neural networks by page 2
”computer network” simulation	Google Scholar	Published since 2020	17,400	Some of relevance on first few pages, then not much
”computer network” emulation	Google Scholar	Published since 2020	6,530	some of use
”linux” AND network*	EBSCO Discovery	Published in the last 5 years and Full Text Online Only and Print	174,898	Bad search; very broad with no discernably of relevance
“linux” (“virtual machine” or “vm”) network*	EBSCO Discovery	Published in the last 5 years and Full Text Online Only and Print	44,213	Bad search; very broad with no discernably of relevance
“linux” (“virtual machine” or “vm”) network*	Google Scholar	Published since 2020	17,000	A few of some relevance, lots of no relevance
”linux” ”network router”	EBSCO Discovery	Published in the last 5 years and Full Text Online Only and Print	5,466	Bad search; very broad with no discernably of relevance

Appendix F

Persistent Data Storage JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "Data Storage Schema for NetShed",
  "type": "object",
  "properties": {
    "nodeArray": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {
            "type": "string"
          },
          "niceName": {
            "type": "string"
          },
          "managementIP": {
            "type": "string"
          },
          "vmid": {
            "type": "string"
          },
          "active": {
            "type": "boolean"
          },
          "type": {
            "type": "string"
          },
          "interfaces": {
            "type": "array",
            "items": {
              "type": "object",
              "properties": {
                "management": {
                  "type": "boolean"
                },
                "name": {
                  "type": "string"
                },
```

```
40                "pveName": {
41                  "type": "string"
42                },
43                "ip": {
44                  "type": "string"
45                },
46                "netmask": {
47                  "type": "string"
48                },
49                "vlan": {
50                  "type": "string"
51                },
52                "routingProtocol": {
53                  "type": "string"
54                },
55                "linkUp": {
56                  "type": "boolean"
57                }
58              },
59              "required": [
60                "management",
61                "name",
62                "pveName",
63                "ip",
64                "netmask",
65                "vlan",
66                "routingProtocol",
67                "linkUp"
68              ]
69            }
70          },
71          "icon": {
72            "type": "object",
73            "properties": {
74              "xTL": {
75                "type": "number"
76              },
77              "yTL": {
78                "type": "number"
79              },
80              "width": {
81                "type": "number"
82              },
83              "height": {
84                "type": "number"
85              }
86            },
87            "required": [
88              "xTL",
89              "yTL",
90              "width",
```

```
91                "height"
92              ]
93            }
94          },
95          "required": [
96            "id",
97            "niceName",
98            "managementIP",
99            "vmid",
100           "active",
101           "type",
102           "interfaces",
103           "icon"
104         ]
105       }
106     }
107   },
108   "required": [
109     "nodeArray"
110   ]
111 }
```

This was generated using transform.tools JSON to JSON Schema tool (https://transform.tools/json-to-json-schema).

Appendix G

Final Validation Test Plan & Outcome

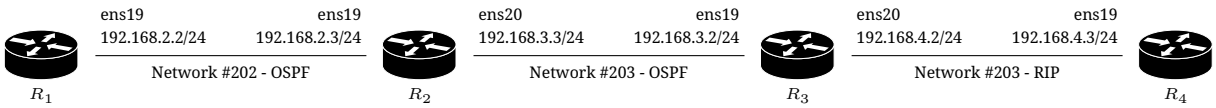


Figure G.1: Network diagram for final validation testing (icons: Cisco, n.d.)

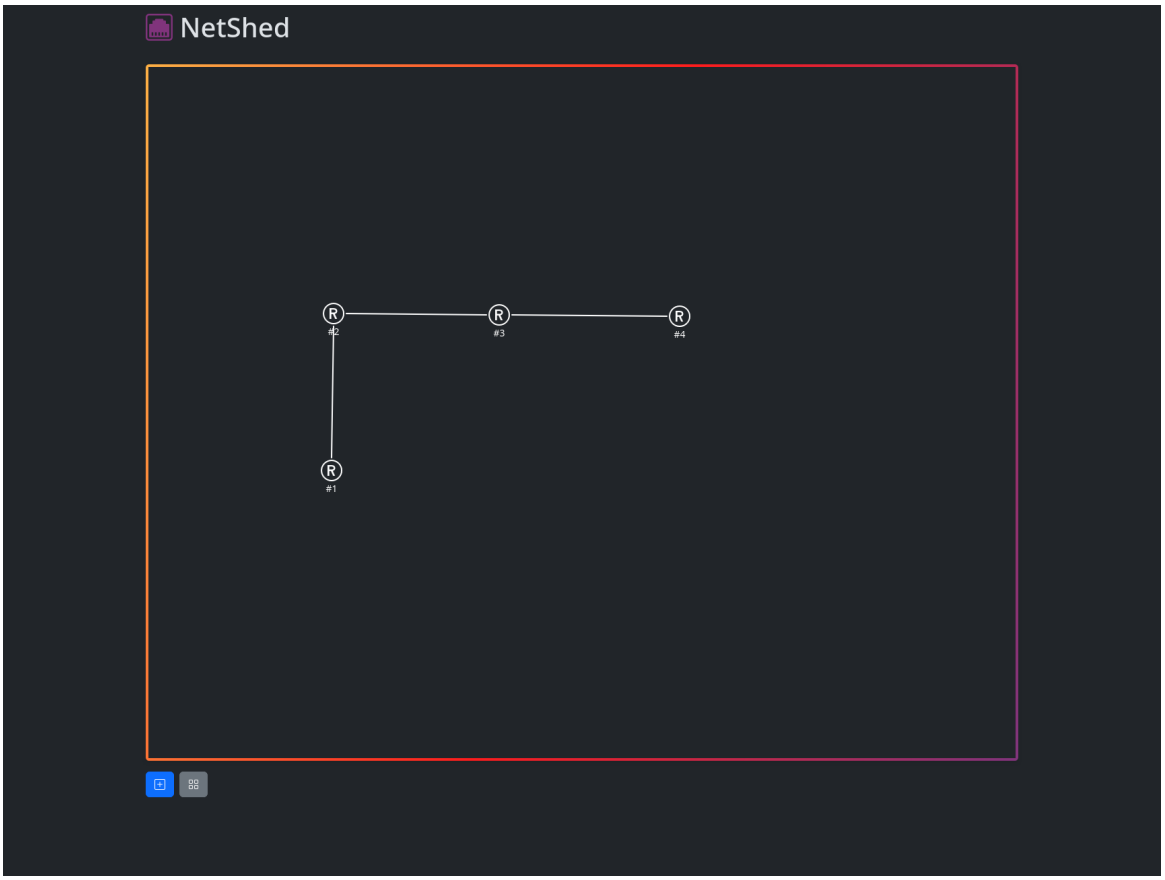


Figure G.2: In-App representation of network for final validation testing (step 9-10)

Table G.1: Application test cases

Case	Action	Possible Outcomes	Expected Outcomes
TC-01	Open app webpage	O1: App webpage opens, welcome modal opens O2: App webpage opens, welcome modal does not open O3: App webpage does not open	O1: App webpage opens, welcome modal opens
TC-02	Welcome Modal > Change tab	O1: Nav bar active tab changes to selected tab, body content changes O2: Nav bar active tab changes to selected tab, body content does not change O3: Nav bar active tab changes to another tab, body content changes O4: Nav bar active tab changes to another tab, body content changes O5: Nav bar active tab does not change, body content changes O6: Nav bar active tab does not change, body content does not change	O1: Nav bar active tab chances to selected tab, body content changes
TC-03	Welcome Modal > Press close button	O1: Welcome modal closes O2: Welcome modal does not close	O1: Welcome modal closes
TC-04	Click and drag mouse on canvas	O1: No nodes move O2: Node moves	O1: No nodes move

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-05	New Node button pressed	O1: New node appears on canvas O2: No new node appears on canvas	O1: New node appears on canvas
TC-06	Node selected	O1: Border appears around node, additional action buttons appear O2: Border appears around node, no additional action buttons appear O3: No change to node border O4: Node disappears from canvas	O1: Border appears around node, additional action buttons appear
TC-07	Node deselected	O1: Border disappears around node, additional action buttons disappears O2: Border disappears around node, additional action buttons remain O3: No change to node border O4: Node disappears from canvas	O1: Border disappears around node, additional action buttons disappears
TC-08	Node dragged across canvas	O1: Node follows mouse O2: Node stays where it was O3: Node moves inversely to mouse O4: Node disappears from canvas	O1: Node follows mouse

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-09	Node selected twice sequentially	O1: Border appears around node; border disappears around node O2: Border appears around node; border remains around node O3: No change to node border status O4: Border appears around node; node disappears from canvas O5: Node moves around canvas	O1: Border appears around node; border disappears around node
TC-10	Delete node button selected	O1: Node disappears from canvas O2: Node moves around canvas O3: Node multiplies on canvas	O1: Node disappears from canvas
TC-11	Node align to grid button selected	O1: Nodes align to grid O2: Nodes stack into a single pile O3: Nodes move randomly around canvas	O1: Nodes align to grid
TC-12	Configure node button selected	O1: Node configuration modal opens to interfaces tab O2: Node configuration modal opens to another tab O3: Node disappears from canvas O4: Node welcome modal opens	O1: Node configuration modal opens

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-13	Node Configuration Modal > Interfaces Tab > Add interface button	O1: New interface added to correct node O2: New interface added to incorrect node O3: No new interfaces added O4: All interfaces on node deleted	O1: New interface added to correct node
TC-14	Node Configuration Modal > Interfaces Tab > Save interface (valid data)	O1: Interface configured, user informed, form controls remain populated O2: Interface configured, user informed, form controls empty O3: Interface configured, user not informed, form controls empty, O4: Interface not configured O5: All configured interfaces deleted	O1: Interface configured, user informed, form controls remain populated
TC-15	Node Configuration Modal > Interfaces Tab > Save interface (invalid data)	O1: Error shown to user, no interface edit made O2: Error shown to user, interface ediTC made O3: Interface configured, user informed, form controls remain populated O4: Interfaces form controls empty	O1: Error shown to user, no interface edit made

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-16	Node Configuration Modal > Interfaces Tab > Delete Interface	O1: Interface deleted, user informed O2: Interface not deleted, user informed it was O3: Incorrect interface deleted	O1: Interface deleted, user informed
TC-17	Node Configuration Modal > Select another tab	O1: Nav bar active tab changes to selected tab, body content changes O2: Nav bar active tab changes to selected tab, body content does not change O3: Nav bar active tab changes to another tab, body content changes O4: Nav bar active tab changes to another tab, body content changes O5: Nav bar active tab does not change, body content changes O6: Nav bar active tab does not change, body content does not change	O1: Nav bar active tab chances to selected tab, body content changes
TC-18	Node Configuration Modal > Routes tab	O1: Routes shown in the summary box O2: No routes shown in the summary box	O1: Routes shown in the summary box
TC-19	Node Configuration Modal > Routes tab > select detailed routing table view	O1: Valid data shown in response box O2: No data shown in response box O3: Data entered into summary box	O1: Valid data shown in the response box

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-20	Node Configuration Modal > Actions tab > Enter IP and press go for Traceroute	O1: Traceroute run and output shown in response box O2: Traceroute run and garbage data shown in response box O3: Error shown in response box	O1: Traceroute run and output shown in response box
TC-21	Node Configuration Modal > Actions tab > Enter IP and press go for Ping	O1: Ping run and output shown in response box O2: Ping run and garbage data shown in response box O3: Error shown in response box	O1: Ping run and output shown in response box
TC-22	Node Configuration Modal > Status tab	O1: Status information for VM and Node shown O2: Status information for VM shown O3: Status information for Node shown O4: Incorrect node / VM status shown O5: Error	O1: Status information for VM and Node shown
TC-23	Node Configuration Modal > Help tab	O1: Help information displayed O2: Unhelpful information displayed O3: Error displayed O4: Modal crashes	O1: Help information displayed

continued on next page

Table G.1: Application test cases (continued)

Case	Action	Possible Outcomes	Expected Outcomes
TC-24	Node Configuration Modal > Press close button	O1: Modal closes, node remains selected O2: Modal closes, node deselected O3: Modal does not close	O1: Modal closes, node remains selected
TC-25	Refresh application	O1: App re-opens, remaining as it was prior to refresh O2: App resets to no nodes O3: App reverts to some other state	O1: App re-opens, remaining as it was prior to refresh

Table G.2: Final validation test plan & outcome

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
1-1	Open the NetShed Web Management Interface in a new tab in Firefox	TC-01	App opens; ‘Welcome’ modal displayed; ‘Welcome’ tab open	App opens; ‘Welcome’ modal displayed; ‘Welcome’ tab open	Pass
1-2	Select the ‘Help’ tab button	TC-02	‘Welcome’ tab content disappears; ‘Help’ tab content appears; ‘Help’ tab becomes selected in navigation bar	‘Welcome’ tab content disappears; ‘Help’ tab content appears; ‘Help’ tab becomes selected in navigation bar	Pass
1-3	Select the X button in the header row of the ‘Welcome’ modal	TC-03	‘Welcome’ modal closes; main app interface shown	‘Welcome’ modal closes; main app interface shown	Pass
2-1	Select the New Node button in the toolbar	TC-05	App loads; new router-type node icon generated on canvas	App loads for 20 seconds; new router-type node (#1) icon generated on canvas	Pass
2-2	Click and drag node #1 to desired location on canvas	TC-08	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
2-3	Single click on node #1	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
2-4	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-201’ with ID #1; ‘Interface’ tab shown; one interface shown	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-201’ with ID #1; ‘Interface’ tab shown; one interface shown	Pass
2-5	Select ‘Help’ tab of Node Configuration Modal	TC-17, TC-23	Help information shown in the body of the modal	Help information shown in the body of the modal	Pass
2-6	Select ‘Status’ tab of Node Configuration Modal	TC-17, TC-22	App loads; two tables rendered - one showing ‘Node Status’ and one showing ‘VM Status’	App loads for < 1 second; two tables rendered - one showing ‘Node Status’ and one showing ‘VM Status’	Pass
2-7	Select ‘Interfaces’ tab of Node Configuration Modal	TC-17	‘Interfaces’ tab opens showing one interface	‘Interfaces’ tab opens showing one interface	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
2-8	Select 'New Interface' button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 2 seconds; renders a new interface card (ens19); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass
2-9	Configure interface ens19 correctly: IP 192.168.2.2; Subnet 24; Network ID 202; Routing Protocol OSPF; Link State up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 4 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass
2-10	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row	Partial Pass
2-11	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
3-1	Select the New Node button in the toolbar	TC-05	App loads; new router-type node icon generated on canvas	App loads for 17 seconds; new router-type node (#2) icon generated on canvas	Pass
3-2	Click and drag node #2 to desired location on canvas	TC-08	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Pass
3-3	Single click on node #2	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
3-4	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-202’ with ID #2; ‘Interface’ tab shown; one interface shown	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-202’ with ID #2; ‘Interface’ tab shown; one interface shown	Pass
3-5	Select ‘New Interface’ button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 2 seconds; renders a new interface card (ens19); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
3-6	Select ‘New Interface’ button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 2 seconds; renders a new interface card (20); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass
3-7	Configure interface ens19 correctly: IP 192.168.2.3; Subnet 24; Network ID 202; Routing protocol OSPF; Link State up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 5 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass
3-8	Configure interface ens20 correctly: IP 192.168.3.3; Subnet 24; Network ID 203; Routing Protocol OSPF; Link state up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 5 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
3-9	Select ‘Actions’ tab of node configuration modal	TC-17	Actions tab opens with two actions available: Ping and Traceroute	Actions tab opens with two actions available: Ping and Traceroute	Pass
3-10	Enter Node #1 ens19 IP into the Ping IP address field and press green go button	TC-21	App loads; results box renders within Ping card; results box populated with successful ping outcome	App loads; results box renders within Ping card; results box populated with summary: ‘2 packets transmitted, 2 received, 0% packet loss’	Pass
3-11	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row	Partial Pass
3-12	Click anywhere on the canvas where there is not a node	TC-07	Selected node’s purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
4-1	Select the New Node button in the toolbar	TC-05	App loads; new router-type node icon generated on canvas	App loads for 23 seconds; new router-type node (#3) icon generated on canvas	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
4-2	Click and drag node #3 to desired location on canvas	TC-08	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Pass
4-3	Single click on node #3	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
4-4	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Pass
4-5	Select 'New Interface' button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 2 seconds; renders a new interface card (ens19); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
4-6	Partially correctly configure interface ens19: IP 1991.168.3.2; Subnet 24; Network ID 203; Routing Protocol OSPF; link state up. Press green save button	TC-15	Warning toast displayed to user saying IP invalid; all fields retain entered data; toast auto-closes after a few seconds	Warning toast displayed to user saying IP invalid; all fields retain entered data; toast auto-closes after a few seconds	Pass
4-7	Partially correctly configure interface ens19: IP 192.168.3.2; Subnet 246; Network ID 203; Routing Protocol OSPF; link state up. Press green save button	TC-15	Warning toast displayed to user saying subnet invalid; all fields retain entered data; toast auto-closes after a few seconds	Warning toast displayed to user saying netmask invalid; all fields retain entered data; toast auto-closes after a few seconds	Partial Pass
4-8	Partially correctly configure interface ens19: IP 192.168.3.2; Subnet 24; Network ID 204; Routing Protocol OSPF; link state up. Press green save button	TC-15	Warning toast displayed to user saying subnet used does not match network ID; all fields retain entered data; toast auto-closes after a few seconds	App loads for 3 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Fail

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
4-9	Partially correctly configure interface ens19: IP 192.168.3.2; Subnet 24; Network ID 399; Routing Protocol OSPF; link state up. Press green save button	TC-15	Warning toast displayed to user saying network ID out of range; all fields retain entered data; toast auto-closes after a few seconds	Warning toast displayed to user saying VLAN invalid; all fields retain entered data; toast auto-closes after a few seconds	Partial Pass
4-10	Partially correctly configure interface ens19: IP 192.168.3.2; Subnet 24; Network ID 203; Routing Protocol unset; link state up. Press green save button	TC-15	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 2 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass
4-11	Correctly configure interface ens19 but don't bring up: IP 192.168.3.2; Subnet 24; Network ID 203; Routing Protocol OSPF; link state down. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 5 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
4-12	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Fail
4-13	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Pass
4-14	Correctly configure interface ens19: IP 192.168.3.2; Subnet 24; Network ID 203; Routing Protocol OSPF; link state up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 3 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
4-15	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Partial Pass
4-16	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
5-1	Single click on node #1	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
5-2	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-201' with ID #1; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-201' with ID #1; 'Interface' tab shown; one interface shown	Pass
5-3	Select 'Actions' tab of node configuration modal	TC-17	Actions tab opens with two actions available: Ping and Traceroute	Actions tab opens with two actions available: Ping and Traceroute	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
5-4	Enter node #3 ens19 IP address into the Ping box and press the go button	TC-21	App loads; results box renders within Ping card; results box populated with successful ping outcome	App loads; results box renders within Ping card; results box populated with summary: ‘2 packets transmitted, 2 received, 0% packet loss’	Pass
5-5	Enter node #3 ens19 IP address into the Traceroute box and press the go button	TC-20	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome containing two hops (192.168.2.3, and 192.168.3.2)	Pass
5-6	Select ‘Routes’ tab of the node configuration modal	TC-17, TC-18	App loads; routes tab opens with Route Summary box rendered and populated with basic routing table; Detailed Routing Tables card present with no data populated	App loads for <1 second; routes tab opens with Route Summary box rendered and populated with basic routing table (including 192.168.3.0/24 using OSPF); Detailed Routing Tables card present with no data populated	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
5-7	Use the ‘Select a view’ dropdown to select ‘OSPF Routes’ and press the green go button	TC-19	App loads; renders results box within Detailed Routing Tables card; populates results box with OSPF routing information	App loads for <1 second; renders results box within Detailed Routing Tables card; populates results box with OSPF routing information including ‘OSPF network routing table’ showing one directly connected network, and one connected via #2; and ‘OSPF router routing table’ showing two other OSPF routers (#2 and #3)	Pass
5-8	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Partial Pass
5-9	Click anywhere on the canvas where there is not a node	TC-07	Selected node’s purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
6-1	Select the New Node button in the toolbar	TC-05	App loads; new router-type node icon generated on canvas	App loads for 23 seconds; new router-type node (#4) icon generated on canvas	Pass
6-2	Click and drag node #4 to desired location on canvas	TC-08	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Mousedown: purple border appears around node; mousemove: node follows mouse; mouseup: purple border around node disappears	Pass
6-3	Single click on node #4	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
6-4	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-204’ with ID #4; ‘Interface’ tab shown; one interface shown	Node configuration modal opens; header of modal shows ‘netshed-app-rtr-204’ with ID #4; ‘Interface’ tab shown; one interface shown	Pass
6-5	Select ‘New Interface’ button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 2 seconds; renders a new interface card (ens19); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
6-6	Correctly configure interface ens19: IP 192.168.4.3; Subnet 24; Network ID 204; Routing Protocol RIP; link state up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 3 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass
6-7	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Partial Pass
6-8	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
7-1	Single click on node #3	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
7-2	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Pass
7-3	Select 'New Interface' button	TC-13	App loads; renders a new interface card; success toast displayed confirming new interface added; toast auto-closes after a few seconds	App loads for 3 seconds; renders a new interface card (ens20); success toast displayed confirming new interface added; toast auto-closes after a few seconds	Pass
7-4	Correctly configure interface ens20: IP 192.168.4.2; Subnet 24; Network ID 204; Routing Protocol RIP; link state up. Press green save button	TC-14	App loads; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 2 seconds; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds; all form controls values unchanged	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
7-5	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Partial Pass
7-6	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
8-1	Single click on node #1	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
8-2	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-201' with ID #1; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-201' with ID #1; 'Interface' tab shown; one interface shown	Pass
8-3	Select 'Actions' tab of node configuration modal	TC-17	Actions tab opens with two actions available: Ping and Traceroute	Actions tab opens with two actions available: Ping and Traceroute	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
8-4	Enter node #4 ens19 IP address into the Ping box and press the go button	TC-21	App loads; results box renders within Ping card; results box populated with successful ping outcome	App loads; results box renders within Ping card; results box populated with summary: ‘2 packets transmitted, 2 received, 0% packet loss’	Pass
8-5	Enter node #4 ens19 IP address into the Traceroute box and press the go button	TC-20	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome containing three hops (192.168.2.3, 192.168.3.2, and 192.168.4.3)	Pass
8-6	Select ‘Routes’ tab of the node configuration modal	TC-17, TC-18	App loads; routes tab opens with Route Summary box rendered and populated with basic routing table; Detailed Routing Tables card present with no data populated	App loads for <1 second; routes tab opens with Route Summary box rendered and populated with basic routing table (including 192.168.3.0/24 using OSPF and 192.168.4.0/24 using OSPF); Detailed Routing Tables card present with no data populated	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
8-7	Use the ‘Select a view’ dropdown to select ‘OSPF’ and press the green go button	TC-19	App loads; renders results box within Detailed Routing Tables card; populates results box with OSPF configuration information	App loads; renders results box within Detailed Routing Tables card; populates results box with OSPF configuration information	Pass
8-8	Use the ‘Select a view’ dropdown to select ‘OSPF Routes’ and press the green go button	TC-19	App loads; populates results box with OSPF routes information	App loads for <1 second; populates results box with OSPF routes information including ‘OSPF network routing table’ showing one directly connected network, and one connected via #2; ‘OSPF router routing table’ showing two other OSPF routers (#2 and #3); ‘OSPF external routing table’ showing management network (10.10.2.0/24) and Network 204 (via 192.168.2.3)	Pass
8-9	Use the ‘Select a view’ dropdown to select ‘OSPF Interfaces’ and press the green go button	TC-19	App loads; populates results box with OSPF interfaces information	App loads for <1 second; populates results box with OSPF interfaces information about the interface using OSPF (ens19)	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
8-10	Use the ‘Select a view’ dropdown to select ‘OSPF Neighbours’ and press the green go button	TC-19	App loads; populates results box with OSPF neighbours information	App loads for <1 second; populates results box with slightly illegible table of neighbouring OSPF routers	Pass
8-11	Use the ‘Select a view’ dropdown to select ‘OSPF Database’ and press the green go button	TC-19	App loads; populates results box with OSPF database contents	App loads for <1 second; populates results box with OSPF Database showing ‘Router Link States’, ‘Net Link States’ and ‘AS External Link States’	Pass
8-12	Use the ‘Select a view’ dropdown to select ‘RIP’ and press the green go button	TC-19	App loads; populates results box with ‘% RIP instance not found’	App loads for <1 second; populates results box with ‘% RIP instance not found’	Pass
8-13	Use the ‘Select a view’ dropdown to select ‘RIP Status’ and press the green go button	TC-19	App loads; populates results box with ‘% RIP instance not found’	App loads for <1 second; populates results box with ‘% RIP instance not found’	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
8-14	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Partial Pass
8-15	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
9-1	Single click on node #4	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
9-2	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-204' with ID #4; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-204' with ID #4; 'Interface' tab shown; one interface shown	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
9-3	Select ‘Routes’ tab of the node configuration modal	TC-17, TC-18	App loads; routes tab opens with Route Summary box rendered and populated with basic routing table; Detailed Routing Tables card present with no data populated	App loads for <1 second; routes tab opens with Route Summary box rendered and populated with basic routing table (including 192.168.2.0/24 using RIP and 192.168.3.0/24 using RIP); Detailed Routing Tables card present with no data populated	Pass
9-4	Use the ‘Select a view’ dropdown to select ‘RIP’ and press the green go button	TC-19	App loads; renders results box within Detailed Routing Tables card; populates results box with RIP routing table	App loads for <1 second; renders results box within Detailed Routing Tables card; populates results box with RIP routing table	Pass
9-5	Use the ‘Select a view’ dropdown to select ‘RIP Status’ and press the green go button	TC-19	App loads; populates results box with RIP status	App loads for <1 second; populates results box with RIP status	Pass
9-6	Select ‘Actions’ tab of node configuration modal	TC-17	Actions tab opens with two actions available: Ping and Traceroute	Actions tab opens with two actions available: Ping and Traceroute	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
9-7	Enter node #1 ens19 IP address into the Ping box and press the go button	TC-21	App loads; results box renders within Ping card; results box populated with successful ping outcome	App loads; results box renders within Ping card; results box populated with summary: '2 packets transmitted, 2 received, 0% packet loss'	Pass
9-8	Enter node #1 ens19 IP address into the Traceroute box and press the go button	TC-20	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome	App loads; results box renders within Traceroute card; results box populated with successful traceroute outcome containing three hops (192.168.4.2, 192.168.3.3, 192.168.2.2)	Pass
9-9	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2, #2 & #3, and #3 & #4 present	Pass
9-10	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	Selected node's purple border removed; selected node control buttons hidden from action row	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
10-1	Single click on node #3	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
10-2	Select node configuration button	TC-12	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Node configuration modal opens; header of modal shows 'netshed-app-rtr-203' with ID #3; 'Interface' tab shown; one interface shown	Pass
10-3	Select 'Routes' tab of the node configuration modal	TC-17, TC-18	App loads; routes tab opens with Route Summary box rendered and populated with basic routing table; Detailed Routing Tables card present with no data populated	App loads for <1 second; routes tab opens with Route Summary box rendered and populated with basic routing table (including 192.168.2.0/24 using OSPF); Detailed Routing Tables card present with no data populated	Pass
10-4	Use the 'Select a view' dropdown to select 'OSPF Routes' and press the green go button	TC-19	App loads; renders results box within Detailed Routing Tables card; populates results box with OSPF routes	App loads for <1 second; renders results box within Detailed Routing Tables card; populates results box with OSPF routes including 192.168.2.0/24 and 192.168.3.0/24	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
10-5	Use the ‘Select a view’ dropdown to select ‘RIP’ and press the green go button	TC-19	App loads; populates results box with RIP Routing table	App loads for <1 second; populates results box with RIP routing table which shows 192.168.2.0/24 via OSPF; 192.168.3.0/24 directly connected, and 192.168.4.0/24 directly connected	Pass
10-6	Select the ‘Interfaces’ tab of the Node Configuration Modal	TC-17	Interfaces’ tab opens showing three interfaces	Interfaces’ tab opens showing three interfaces	Pass
10-7	Select the ‘Delete’ button on ens20	TC-16	App loads; interface card removed from ‘Interfaces’ tab in Node Configuration modal; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	App loads for 3 seconds; interface (ens20) card removed from ‘Interfaces’ tab in Node Configuration modal; success toast displayed confirming interface has been updated; toast auto-closes after a few seconds	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
10-8	Select the X button in the header row of the Node Configuration modal	TC-24	Modal closes; main app interface shown; selected node still has purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Modal closes; main app interface shown; selected node does not have purple border; selected node control buttons still present below canvas in action row; line connecting nodes #1 & #2 and #2 & #3 present	Partial Pass
10-9	Click anywhere on the canvas where there is not a node	TC-07	Selected node's purple border removed; selected node control buttons hidden from action row	No change to node border status; selected node control buttons hidden from action row	Partial Pass
11-1	Select the 'Node align to grid' button	TC-11	Nodes grid-align in top left of canvas	Nodes grid-align in top left of canvas; error toast containing message 'Critical Error' thrown; toast disappears after a few seconds	Partial Pass
11-2	Press Ctrl + F5 to hard refresh application	TC-25	Web page refreshes; nodes remain in the same location; 'Welcome' modal not rendered	Web page refreshes; nodes remain in the same location; 'Welcome' modal not rendered	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
12-1	Single click on node #4	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
12-2	Select the ‘Node Delete’ button	TC-10	App loads; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	App loads for 3 seconds; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	Pass
12-3	Single click on node #3	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
12-4	Select the ‘Node Delete’ button	TC-10	App loads; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	App loads for 2 seconds; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	Pass

continued on next page

Table G.2: Final validation test plan & outcome (continued)

Step	Action	Test Case	Expected Outcomes	Actual Outcomes	Result
12-5	Single click on node #2	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
12-6	Select the ‘Node Delete’ button	TC-10	App loads; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	App loads for 3 seconds; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	Pass
12-7	Single click on node #1	TC-06	Purple border appears around node; selected node control buttons appear below canvas in action row	Purple border appears around node; selected node control buttons appear below canvas in action row	Pass
12-8	Select the ‘Node Delete’ button	TC-12	App loads; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	App loads for 2 seconds; node disappears from canvas; success toast displayed confirming node has been deleted; toast auto-closes after a few seconds	Pass
13-1	Refresh Application	TC-25	Web page refreshes; no nodes appear; ‘Welcome’ modal displayed open on ‘Welcome’ tab	Web page refreshes; no nodes appear; ‘Welcome’ modal displayed open on ‘Welcome’ tab	Pass

Appendix H

Deployment Playbook

NetShed Deployment Instructions

The process to deploy NetShed is quite hands on. There are lots of different components which have to be provisioned and configured in a specific order. The components have broadly been split into parts.

There are a number of variables used throughout this process. They are denoted with a dollar sign (`$`) prepended to their name, for example `$APPSERVER_VM_IP`. For reference, these variables are listed with an explanation and example below

Variable	Use	Example
`$PVE_HOST_IP`	The IP address of the Proxmox VE Machine.	`10.0.0.2`
`$APPSERVER_VM_IP`	The IP address of the App Server VM	`10.0.0.3`
`$RESOURCE_SERVER`	The Protocol, IP & Port of the Resource Server Docker container	`http://10.0.0.3:8081`

Part 0: Preparation

> Objective: Locate resources and make decisions in preparation for deploying NetShed and begin preparing app files.

Things to Find

- A PC capable of running NetShed (see *README/System Requirements* for more information) - referred to during these instructions as "PVE Host Machine"
- A Laptop / PC to open these instructions on and connect to the NetShed Host with - referred to during these instructions as "Your PC"
- A USB Stick which can be used as a bootable USB Stick
- Monitor, network cables, peripherals

Things to Decide On

- IP Addresses - see *README/Network Requirements* for more information

On Your PC

1. Download the Netshed Application files from Moodle / clone from GitHub (depending on where you've found this document). Unzip and put somewhere sensible on your PC where you can open with File Explorer, a terminal of your choosing, and edit files within it.

Part 1: Proxmox VE Installation & Configuration

> Objective: Install Proxmox VE onto the host machine and complete configuration to create API Keys to be used by NetShed Web Management Interface

On Your PC

1. Download the Proxmox VE 9.1 Installer ISO
2. Burn this to a USB stick to make it bootable - using a tool such as Balena Etcher or add it to a Ventoy enabled USB drive.

On the PVE Host Machine

1. Install Proxmox VE onto the host machine. The Proxmox Wiki contains installation instructions.

From the Proxmox VE Shell

1. Create the new user:

```
pveum user add netshedapi@pam
```

2. Apply the Administrator role to the user:

```
pveum acl modify / -user netshedapi@pam -role Administrator
```

3. Generate an API token for the user

```
pveum user token add netshedapi@pam nshdtokn --privsep=0
```

4. From the output - copy the `full-tokenid` and `value` and save these somewhere safe.

Part 2: Deploy the App Server

> Objective: To provision and configure a VM to act as the 'App Server' ready to deploy Docker containers to in a later Part.

This is a very convoluted process as the later way to deploy a VM is considerably shorter as it utilises the App Server. However the App Server has to be deployed for that to work, and the App Server can't be used to deploy itself.

On Your PC

1. SCP the `scripts/provision-base-vm.sh` file to the Proxmox VE Host into the `/root` directory.

```
scp scripts/provision-base-vm.sh root@$PVE_HOST_IP:/root
```

From the Proxmox VE Shell

1. Open the `provision-base-vm.sh` file using `nano` for editing:

```
nano provision-base-vm.sh
```

2. Replace the placeholder values assigned to the four variables at the top of the page with production values.
   - `VM_ID`: The Proxmox ID to be assigned to the App Server VM
   - `VM_NAME`: The hostname of the App Server
   - `IP_ADDR`: The IP address to assign to the App Server VM in the form IPADDR/CIDR, for example `10.0.0.2/24`
   - `GATEWAY`: The gateway of the network
3. Exit out of the nano editor using `Ctrl + X` then `y` to save.
4. Use `chmod` to make the script executable

```
chmod +x provision-base-vm.sh
```

5. Run the script:

```
./provision-base-vm.sh
```

6. Wait for the script to complete. It will output a 'Complete' message on termination.

From the Proxmox VE Web UI

1. Open the App Sever VM's console (either the NoVNC or Xterm.js consoles work).

From the App Server Console

1. Confirm the VM has fully booted. You should be presented with a login prompt and the 'SSH Key image' should have scrolled past.
2. Login using the credentials `root` with the password `tempP@ssword123!`
3. Update package databases

```
apt update
```

4. Install the QEMU Guest Agent

```
apt install qemu-guest-agent
```

5. Enable the QEMU Guest Agent service and instruct it to start now (NB. this will return some errors, these can be safely ignored as the service will start regardless)

```
systemctl enable qemu-guest-agent --now
```

On Your PC

1. SCP the `scripts/appserver-init-pt1.sh` file to the Proxmox VE Host into the `/root` directory.

```
scp scripts/appserver-init-pt1.sh root@$PVE_HOST_IP:/root
```

From the Proxmox VE Shell

1. Open the `appserver-init-pt1.sh` file using `nano` for editing:

```
nano appserver-init-pt1.sh
```

2. Replace the placeholder values assigned to the four variables at the top of the page with production values.
   - `VM_ID`: The Proxmox ID assigned to the App Server VM
3. Exit out of the nano editor using `Ctrl + X` then `y` to save.
4. Use `chmod` to make the script executable

```
chmod +x appserver-init-pt1.sh
```

5. Run the script:

```
./appserver-init-pt1.sh
```

6. Wait for the script to complete. It will output a 'Complete' message on termination. This script may output lots of 'questionable, garbled nonsense' - this is fine and to be expected.

On Your PC

1. SCP the `scripts/appserver-init-pt2.sh` file to the App Server VM into the `/root` directory.

```
scp scripts/appserver-init-pt2.sh root@$APPSERVER_VM_IP:/root
```

2. SSH into the App Server VM using the root user. Type yes when prompted to trust the SSH keys.

```
ssh root@$APPSERVER_VM_IP
```

From the SSH Session with the App Server

1. Use `chmod` to make the script executable

```
chmod +x appserver-init-pt2.sh
```

2. Run the script

```
./appserver-init-pt2.sh
```

3. Wait for the script to complete. This will output Complete when it has terminated.

Part 3: Deploy the Resource Server Docker Container

Objective: Deploy the Resource Server Docker container to the App Server and bring it online

On Your PC

1. SCP the `resource-server/` directory to the App Server into the `/root/docker/` directory

```
scp -r resource-server/ root@$APPSERVER_VM_IP:/root/docker/
```

2. SSH into the App Server using the root user.

```
ssh root@$APPSERVER_VM_IP
```

From the SSH Session with the App Server

1. Navigate into the `resource-server` folder we have just uploaded

```
cd /root/docker/resource-server
```

2. Start the container in detached mode

```
docker compose up -d
```

On Your PC

1. Open a web browser and navigate to the Resource Server's location (`http://$APPSERVER_VM_IP:8081`). Confirm a web page is presented to you.

Part 4: Router VM Template Creation

Objective: To create the template which NetShed Web Management will use to create new VMs from

From the Proxmox VE Shell

1. Modify the `provision-base-vm.sh` script to update the variables at the top of the file:
   - `VM_ID`: The Proxmox ID to be assigned to the NetShed Template Router VM
   - `VM_NAME`: The hostname of the NetShed Template Router
   - `IP_ADDR`: The IP address to assign to the NetShed Template Router VM in the form IPADDR/CIDR, for example `10.0.0.2/24`
   - `GATEWAY`: The gateway of the network
2. Run the `provision-base-vm.sh` script:

```
./provision-base-vm.sh
```

3. Wait for the script to execute. It will exit automatically on completion.

From the Proxmox VE Web UI

1. Open the VM Console (either the NoVNC or Xterm.js Consoles work)

From the VM Console

1. Logon using the username `root` and the password `tempP@ssword123!`
2. Download the NetShed Router Template configuration file (`router-template-init.sh`):

```
wget -O $RESOURCE_SERVER/router-template-init.sh ~/router-template-init.sh
```

3. Edit the downloaded script using the `nano` editor to set the `RESOURCE_SERVER` variable near the top of the file. This should be in the format `http://1.2.3.4:8081/`
4. Make the script executable:

```
chmod +x ~/router-template-init.sh
```

5. Run the script. This will ask for a the Enter key to be pressed twice to acknowledge the completion of the main script and again before the VM automatically shuts down. Once the script has finished executing - the VM will shutdown.

```
~/router-template-init.sh
```

From the Proxmox VE Web UI

1. Right click on the VM, which should now be shut down, and select "Convert To Template".

Part 5: Prepare the Environment Configuration File

Objective: To create the `.env` file used by the NetShed Web Management Interface and populate it with the requisite configuration information

On Your PC

1. Navigate to the `netshed-web-management-interface` directory within the NetShed Application files you downloaded in Part 0.
2. Make a copy of the `.env.sample` file and call it `.env`.
3. Replace the placeholder content as follows:
   - `API_PVE_API_KEY`: Proxmox API Key generated in Part 1. This should be inserted in the format `$FULL_TOKEN=$VALUE` for example `netshedapi@pam!nshdtokn=aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa`
   - `API_PVE_NODE_NAME`: The hostname of your PVE node
   - `API_PVE_TEMPLATE_ID`: The numerical ID number for the NetShed Router VM Template (variable `$TEMPL_VM_ID` from earlier in these instructions)
   - `API_PVE_API_LOC`: The IP address & port of your PVE node. For example `10.10.0.10:8006`
   - `API_CFG_INIT_NEWNODE_PVEID`: The first VM ID to use in sequence when generating new VMs. For example `100`
   - `API_CGF_INIT_NEWNODE_IPADDR`: The first IP address to use in sequence when generating new VMs. For example `10.10.10.2`
   - `API_CGF_INIT_NEWNODE_APPID`: The first NetShed Internal ID to assign to new VMs. For example `1`.

Part 6: Deploy the Web App Docker Container

Objective: Deploy the Web App Docker container to the App Server and bring it online

On Your PC

1. SCP the `netshed-web-management-interface/` directory to the App Server into the `/root/docker/` directory

```
scp -r netshed-web-management-interface/ root@$APPSERVER_VM_IP:/root/docker/
```

2. SSH into the App Server using the root user.

```
ssh root@$APPSERVER_VM_IP
```

From the SSH Session with the App Server

1. Navigate into the `netshed-web-management-interface` folder we have just uploaded

```
cd /root/docker/netshed-web-management-interface
```

2. Start the container in detached mode

```
docker compose up -d
```

On Your PC

1. Open a web browser and navigate to the Web management interfaces' location (`http://$APPSERVER_VM_IP`). Confirm a web page is presented to you.
2. Navigate to the debug page (`http://$APPSERVER_VM_IP/api/debug.php`). Confirm no errors are shown. If any errors are shown - they will need to be resolved before NetShed will be fully operational