



VCU

College of Engineering

CS25-310 Advancing High-Performance Computing Preliminary Design Report

Prepared for

Alberto Cano

VCU College of Engineering

By

Yunus Bidav, Steven Holcombe, James Jenkins, Amaka Odidika

Under the supervision of

Alberto Cano, Carlisle Childress, Romano Woodfolk

11 Oct 2024

Executive Summary

This document presents a comprehensive preliminary design report for the senior capstone project spearheaded by the High Performance Research Computing (*HPRC*) core laboratory at Virginia Commonwealth University (*VCU*). Central to this initiative is the establishment and configuration of a state-of-the-art high-performance computing cluster, equipped with updated software and hardware. The setup of this cluster will then be both demonstrated and evaluated for its potential integration into existing production clusters operated by the HPRC.

In this report, which forms the second major segment of the final report, several key sections outline the foundational planning stages and design decisions of the project. The problem statement provides essential background information and articulates the overarching purpose of the project, establishing a clear context for the proposed work. Following this, the engineering design requirements delineate the general goals, specific design objectives, technical specifications, relevant codes, and standards that will guide efforts. In addition, we elaborate on the scope of work, presenting more concrete indicators of progress, such as detailed deliverables and critical milestones. This clarity will facilitate effective tracking of our advancements throughout the project life-cycle. The concept generation and concept selection sections detail the narrowing down of design choices that lead to the final cluster design. The design methodology then provides the details of this design. Additionally, societal impact and cost analysis are performed, after which general conclusions about the project are drawn. Finally, the report includes Appendices 1 and 2, which feature a Gantt chart timeline outlining project phases and a team contract that underscores our collective commitments and responsibilities.

Table of Contents

Section A. Problem Statement	4
Section B. Engineering Design Requirements	5
B.1 Project Goals	5
B.2 Design Objectives	5
B.3 Design Specifications and Constraints	5
B.4 Codes and Standards	5
Section C. Scope of Work	7
C.1 Deliverables	7
C.2 Milestones	7
C.3 Resources	8
Section D. Concept Generation	9
D.1 Minimalist HPC Cluster with Manual Configuration	9
D.2 User-Centric HPC with Simplified Access Tools	9
D.3 LLM and AI Workload Integration	9
Section E. Concept Evaluation and Selection	11
Section F. Design Methodology	13
F.1 Validation Procedure	13
Section G. Results and Design Details	14
Section H. Societal Impacts of Design	15
H.1 Public Health, Safety, and Welfare	15
H.2. Economic Impacts	15
H.3 Environmental Impacts	15
H.4. Ethical Considerations	15
Section I. Cost Analysis	16
Section J. Conclusions and Recommendations	16
Appendix 1: Project Timeline	17
Appendix 2: Team Contract	
Contents	18
Step 1: Get to Know One Another. Gather Basic Information.	18
Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.	19
Step 3: Time Commitments, Meeting Structure, and Communication	20
Step 4: Determine Individual Roles and Responsibilities	20
Step 5: Agree to the above team contract	21
References	22

Section A. Problem Statement

High Performance Computing (*HPC*), as described by IBM, uses clusters of machines which work in parallel to process large data sets and solve complex problems at much higher speeds than other methods [1]. This is in contrast to mainframes, which are large-scale, high-performance single computers. Large HPC clusters are often referred to as supercomputers. The Virginia Commonwealth University High Performance Research Computing Center operates two production HPC clusters, Athena and Apollo [2].

Many researchers at VCU use the HPRC's resources to do intensive research. For example, rendering frames, statistical support, and fluid and molecular dynamics, to name a few [3]. The HPRC has been continuously operating and innovating since 2006 [4], and this year is no exception. The goal of our HPRC advancement project is to highlight the capabilities and benefits of our resources while supporting their integration for VCU students. The project at hand is unique and distinctly open-ended, allowing flexibility in implementation. Thus, we are to build a cluster from the ground up, covering aspects from hardware configuration to software setup. Our client is VCU, and the community of users using varying resources that the HPRC provides.

We are led by Alberto Cano and Carlisle Childress. They have been and will continue to be instrumental in our efforts for the capstone project. Alberto Cano is our faculty advisor and our sponsor. He has been our first point of contact during the beginning of the project and helped lay the foundation for what is to come throughout the year. Carlisle Childress is our acting technical advisor and manager. Carlisle is meeting with our group weekly, showing us current updates for Athena, and continuing the setup and timeline for our fall-winter semester goals.

We aim to install HPL, a ranking system for compute clusters to see where our system and configuration compare against other clusters around the globe. Another software that we are to install and set up for ease of use is Open OnDemand. This will allow a web interface for users who may not be familiar with the intricacies of connecting to a cluster and sending a compute workload using the terminal or other specialized software. This specific aspect of the project is one that we aim to have fully operational come spring for our demonstrations. Most students in computing-related programs at VCU have little to no familiarity with Athena. Installing Open OnDemand will be an effective introduction to interacting with the cluster for novice users. Furthermore, we are using recent versions of Rocky Linux 9 along with other core software. As these versions are recent, they may provide insight and showcase interesting capabilities during demonstration.

Previous HPRC capstones have followed guidelines similar to ours. However, a significant distinction lies in our aim to implement LLMs and other AI-related workloads. This introduces the challenge of running these tools on a cluster that may not have been specifically designed for them. Nonetheless, we will adhere to current guidelines and standards while referencing the work of previous teams to ensure consistency in the overall application.

Section B. Engineering Design Requirements

B.1 Project Goals

We aim for the following goals. These objectives should be completed and ready for display before the capstone project exposition.

- Working cluster hardware
- Software stack being utilized
- List of integrated software
- Ranked performance of the HPC
- Access to Open OnDemand for the project expo for student use
- Integration with LLMs and other AI workloads
- Educate the student body about Athena and its abilities

B.2 Design Objectives

- Support the execution of a standard HPL benchmark.
- Allow remote access within the VCU network.
- Enable scalability for future expansion of compute resources.
- Demonstrate improved performance over existing HPRC clusters on specific benchmark tests.
- Implement efficient resource allocation and scheduling using Slurm.
- Accommodate a variety of scientific and research applications commonly used by VCU researchers.

B.3 Design Specifications and Constraints

- Communication between compute nodes must use IEEE 802.3-compliant Ethernet cabling and the Message Passing Interface (*MPI*).
- Hardware must be mountable in a standard 19" wide rack space.
- Most utilized software should ideally be free and open source.
- Resource provisioning must utilize Slurm and Warewulf.
- Guidelines provided by OpenHPC documentation must be incorporated.

B.4 Codes and Standards

When working in a HPC environment the following standards should be applied to ensure security, performance, and modularity. These standards are applicable to all hardware and software setups we may use to integrate the above requirements and specifications.

- NIST SP 800-223: The design and access of HPC environments are advised to adhere to the best practices listed in the standard's write up.
- IEEE 802.3: Ethernet standard which is used throughout the hardware connections between hardware racks.
- MPI 4.1: MPI, the method of communication between the nodes of the cluster.

- IEEE P2416: Resource management standards for HPC workloads and service management.
- NIST SP 800-53: Provides best practices for security and privacy controls for cloud computing in general.
- IEEE 802.1: Standards for general local access networks and network management for servers on a hardware scale.
- IEEE 2937-2022: Benchmarking standards for intensive workloads related to AI workloads.

Section C. Scope of Work

C.1 Deliverables

The following are all of the deliverables we are expected to produce. All of the non-academic related deliverables were discussed with the team through various in person meetings along with our mentor, Carlisle Childress. These are to be completed in full, in a nearly sequential manner, as the project itself builds together one step after the other. Note, these ignore most academic deliverables.

- Team contract
- GitHub repo setup
- Project proposal
- Preliminary design report
- The first POC cluster is brought up and accessible
- Configure Slurm
- Fall poster
- Open HPC integration completed
- HPL ranking
- Install LLM and AI tools
- Final Design Report
- Fully working cluster with all software installed
 - Accessible through Open OnDemand
- Capstone Expo poster
- Capstone Expo presentation

C.2 Milestones

- Project Initiation (*Completed*)
 - Initial meetings with advisors.
 - Familiarization with project scope and HPC environment.
- Hardware Setup (*Ongoing*)
- Operating System Installation (*Prototype Completed*)
 - Install Rocky Linux 8 on a test cluster.
 - Migrate to Rocky Linux 9.
- Software Implementation (*Prototype Completed*)
 - Install and configure OpenHPC.
 - Set up Slurm workload manager.
 - Integrate Message Passing Interface (MPI).
- Performance Benchmarking (*Upcoming*)
 - Install HPL (High-Performance Linpack).
 - Rank cluster performance.

- User Interface Development (*Upcoming*)
 - Implement OpenHPC web interface.
- LLM and AI tool integration (*Upcoming*)
- Documentation and Testing (*Upcoming*)
 - Develop user guides and documentation.
- Final Deliverables (*Upcoming*)
 - Prepare the final design report.
 - Create capstone expo posters and presentations.
- Project Closure (*Upcoming*)
 - The final demonstration of the working cluster.
 - Handover to HPRC for potential integration.

C.3 Resources

Hardware is typically the most expensive piece aspect of clusters. Fortunately, we are able to develop and test using pre-existing clusters at the HPRC. In fact, several servers, which are currently not integrated into publicly available services, are poised to serve as the head and compute nodes for the capstone cluster. This pre-existing infrastructure will significantly streamline our efforts and enhance our operational capabilities. Furthermore, proficiency in Linux systems administration will be paramount for the successful setup and configuration of the cluster. This expertise will ensure that we can effectively manage the hardware resources, ensure optimal performance, and facilitate a seamless integration of the necessary software tools. Lastly, all efforts will aim to use mostly free and open-source software. This means that this project will likely proceed without the need for much funding, relying mostly on pre-existing resources and knowledge at the HPRC.

Section D. Concept Generation

To address the challenges of building a high-performance computing (*HPC*) cluster that balances usability, functionality, and modern computational workloads, we propose the following three design concepts. Each concept explores a different focus area that aligns with the project goals. Furthermore, each concept emphasizes feasibility, ensuring that the proposed solutions can be implemented within the project timeline.

D.1 Minimalist HPC Cluster with Manual Configuration

Here, we prioritize a fundamental approach to set up the HPC cluster. We focus on a manual setup from the ground up, including configuring the head node, the compute nodes, and the software environment. We aim to use Rocky Linux 9 as the operating system, with Slurm for workload management, as well as other tools such as OpenHPC and Warewulf to support the cluster setup. By following a manual setup with outlined step-by-step instructions, we aim to build a functional and stable cluster without relying on automated tools.

General Benefits:

- Allows the team to gain experience with cluster setup and administration.
- Ensures a simple, well-documented system using widely adopted tools.

General Challenges:

- Manual setup is time consuming and requires troubleshooting.
- Configuration errors may impact cluster performance if not addressed carefully.

D.2 User-Centric HPC with Simplified Access Tools

Here, we focus on improving accessibility for users without prior experience with HPC systems. By incorporating tools such as Open OnDemand, we aim to allow users to interact through simple graphical interfaces. These tools will allow basic workflows to be managed without an explicit understanding of terminal-based commands.

General Benefits:

- Enhances usability and accessibility
- Encourages more users to explore and experiment with HPC capabilities.

General Challenges:

- Web-based tools require additional configuration
- Determining which tools and features to include can be difficult

D.3 LLM and AI Workload Integration

Here we focus on integrating support for AI workloads, including large language models (*LLMs*), into the HPC cluster. We aim to provide users with a platform to experiment with modern AI tasks. With this, we hope to broaden the scope of the cluster's applications and encourage exploration, research, and hands-on learning in areas such as machine learning, natural language processing, and other computational AI projects.

General Benefits:

- Encourages exploration and research into AI applications by users.
- Maintains cluster relevance by supporting modern workloads.

General Challenges:

- Integrating AI workloads adds complexity to the cluster setup.
- AI workloads require significant resources.

Each design concept focuses on a specific goal, with an emphasis on system setup, user accessibility, and AI experimentation, while maintaining realistic expectations. These concepts will be evaluated in Section E to determine relevance and significance.

Section E. Concept Evaluation and Selection

Although all design concepts are significant, ordering is necessary to prioritize implementation and attention. To determine the significance of the design concept, we use a systematic approach. Each concept was evaluated against each other on the basis of specific criteria, taking into account objectives, constraints, and expectations.

The following criteria, along with their respective weights, were used to evaluate the design concepts. The weights were determined based on the perception of the relative importance of each criterion in the achievement of the project goals.

Usability: (0.25)

- Accessibility of the system for users with varying levels of technical experience.

Complexity: (0.2)

- Level of difficulty in setup, integration, and maintenance.

Scalability: (0.15)

- Ability to expand the system to support future workloads.

Resource Requirements: (0.2)

- Reasonable amount of computational and hardware resources required to implement and sustain the design.

Educational Value: (0.2)

- The extent to which the design provides opportunities for experimentation, and exploration by users.

The following table presents the evaluation of each design concept based on the defined criteria and weighting factors. Scores range from 1 to 5, with higher scores indicating a better alignment with the criteria. Furthermore, a final weighted total score was calculated to be used for the concept ranking.

Criteria	Weight	Minimalist HPC	User-Centric HPC	LLM/AI Integration
Usability	0.25	3.5 (0.875)	4.5 (1.125)	3.5 (0.875)
Complexity	0.20	3.5 (0.7)	3.0 (0.6)	2.0 (0.4)
Scalability	0.15	4.0 (0.6)	3.5 (0.525)	2.5 (0.375)
Resource Requirements	0.20	4.5 (0.9)	3.5 (0.7)	2.0 (0.4)
Educational Value	0.20	3.5 (0.7)	4.0 (0.8)	4.5 (0.9)
Total Score		3.775	3.75	2.95

Based on the evaluation, the minimalist HPC cluster with manual configuration is the most significant design concept, as it provides the essential foundation for all other work. Manual configuration of the cluster is necessary to establish a functional, stable, and well-documented system. Although the user-centric HPC with simplified access tools and LLM and AI workload integration are important for usability and expanded capabilities, they depend on the successful configuration of the cluster. Even as we move toward improving accessibility and integrating AI workloads, the core setup remains central to the project's success. In the future, we will prioritize the foundational set-up of the cluster.

Section F. Design Methodology

Firstly, we opt for a manual build of our in-progress production system. In particular, we follow the installation guides for OpenHPC and Rocky Linux 9. By following the guides and documenting intermediate issues, we gain a significant understanding of the system design. With that in mind, after our manual process, we opt to create a script to automatically build the entire cluster. FTP servers and Slurm configurations were some of the early challenges we faced. Without manually building out the cluster step by step and being able to interact with the systems at each failing step, it would not be feasible to understand and rectify issues as readily. Once the shell script has been completed, we intend to design a one-click solution to deploy a fully available production cluster. This has been the goal from the start from a redundancy and replicability perspective.

In terms of cluster design, apart from deployment, a fairly standard implementation of a main head node and some compute nodes were used. A spoke architecture is what was outlined in the install guides. Since the documentation was followed closely, the current install closely parallels the limited install documentation. This ensures not only robustness, but also fallback in the case of outages. Because we do not have a bespoke deployment, the process of debugging and error resolution is very smooth compared to a completely custom solution. We aim for ease of access and learning for the design goals, and that is what we believe we are on track to achieve.

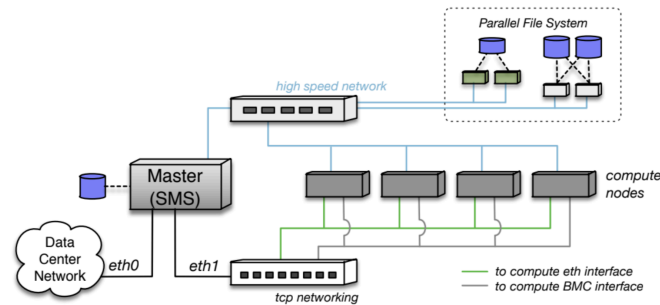
F.1 Validation Procedure

For testing we use a server called Bender, which has virtualization capabilities useful for development. For our final installation, we intend to migrate to a cluster called Scylla. Doing development in this environment isolates us from the outside world so that we are less prone to both user error and external attacks. To measure the integrity of the cluster and our work, we have our mentor Carlisle verifying the state of each cluster and authorizing us to continue to sequential steps.

Apart from checking the architecture itself, the critical services side of the cluster is something that can easily cause a non-working condition of the whole cluster. We start by checking the head node to ensure that every HPC related system service is healthy. Using a manual method of checking each integral service such as Warewulf control, Slurm, and Slurm database, we are able to get a clear picture of the health and functionality of the head node. For the worker nodes, since they are built off of a pre-built image that is downloaded from a repository, not much checking is involved. Checking only one worker node is sufficient, and if the services within a worker node are healthy, we are able to check off worker node functionality. Finally, running jobs on the OHPC platform will be the final test of our design and its ability to perform as we have expected.

Section G. Results and Design Details

Currently, we have developed a virtualized prototype HPC cluster, completing key phases of the design methodology. The prototype demonstrates basic functionality of foundational tools and provides a platform upon which the later iterations can be based. However, implementations of simplified access tools, AI workflows, and benchmarking are pending. However, this prototype validates the feasibility of our design and ensures a stable foundation for further development.



[5]

Technologies Used

- Virtualization: A virtual machine serves as a head node, enabling sandboxed development and testing.
- Operating System: Rocky Linux 9, ensuring compatibility with modern HPC tools.
- Job Scheduling: Slurm, allowing resource allocation and basic job execution.
- Provisioning: Warewulf, delivers prebuilt compute node images to hardware.
- Build Systems: Spack and Easybuild are selected for allowing researchers to access scientific software easily.
- Containerization: Apptainer allows for the use of software containers without root permissions, making it the best choice for running containers on a multi-user HPC system.
- Web Access: OpenOnDemand provides an easy-to-use web interface through which to access HPC clusters.

Key Achievements

- Prototype Virtual Head Node: Successful implementation of a virtualized head node as working prototype.
- Documentation of Installation Issues: Identified and documented challenges encountered while following install guides, ensuring smoother deployment in later iterations.
- Verification of Basic Functionality: Confirmed operation of basic core services validating the initial setup.

Pending Developments

- Simplified Access Tools: Integration of Open OnDemand for improved user accessibility
- AI Workflows: Configuration for running AI and LLM workloads.
- Performance Benchmarking: Testing with High-Performance Linpack (*HPL*) to evaluate cluster performance.

The completed prototype serves as a milestone, laying the groundwork for a fully functional HPC cluster that meets our project objectives.

Section H. Societal Impacts of Design

H.1 Public Health, Safety, and Welfare

As it stands, no critical research is dependent upon this project. However, the design realizes safety and reliability to ensure a secure and accessible HPC environment for demonstration purposes. Thus, we focus on showcasing modern HPC workflows and technologies with potential broader impacts. For example,

- Educational Potential: Demonstration may encourage researchers already conducting large-scale data analysis to explore HPC systems for improving computational workflows.

In general, to achieve such potential, it is important the cluster functions reliably and securely. This means special attention should be paid to,

- System Security: Adhering to common standards establishes secure access and operation, safeguarding against vulnerabilities.
- Reliability: By validating the head node and documenting installation challenges, we ensure stability in demonstrating new technologies and workflows to users.

H.2. Economic Impacts

We emphasize cost-effective design and resource utilization. This includes,

- Hardware Efficiency: Reusing existing equipment minimizes costs while extending the lifecycle of hardware components.
- Long-Term Potential: Showcasing the prototype may encourage future investment in HPC infrastructure, benefiting both the university and its research ecosystem.

H.3 Environmental Impacts

The project adopts environmentally conscious practices during the prototype development:

- Reuse of Hardware: Similar to economic impact, by using existing servers and components, the project minimizes electronic waste and reduces the environmental impact associated with purchasing new hardware.
- Energy-Efficient Prototyping: Developing the prototype in a virtualized environment lowers power consumption during testing and validation phases.

H.4. Ethical Considerations

The project promotes ethical and responsible use of HPC resources. These include,

- Accessibility: Using open-source tools and workflows ensures that technologies demonstrated in this project remain accessible to a broad audience.
- Transparency: Documenting challenges and installation issues allows for clear knowledge transfer.
- Responsible Use: By showcasing the cluster for educational purposes, the project avoids misuse of computational resources.

Section I. Cost Analysis

Currently, we have not incurred any costs. Most hardware is recycled from different parts of retired HPRC components or surrounding VCU services. We plan on adding InfiniBand support, but costs will vary and likely will not fall on our team. Nevertheless, we have had a special emphasis on being resourceful when it comes to hardware.

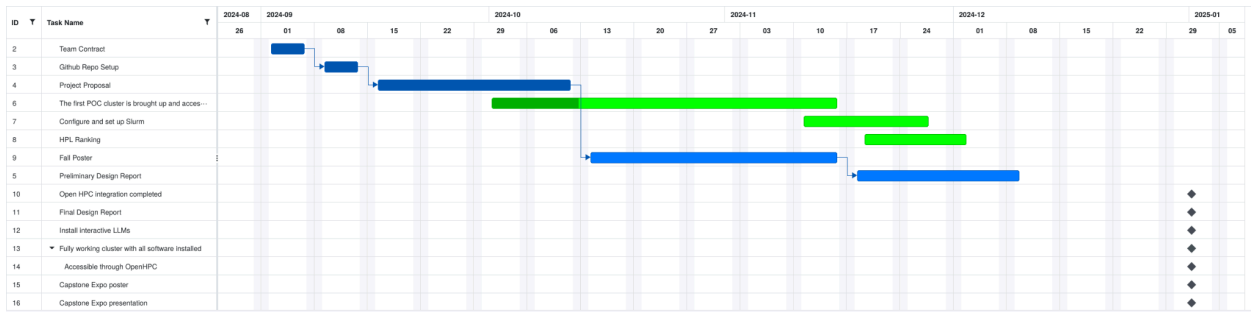
Furthermore, it is exceedingly difficult to determine the cost impact of running the production cluster at this time, as not all machines have been measured and the final configuration of our cluster is uncertain.

Section J. Conclusions and Recommendations

By this time, it has been demonstrated why, for the purposes of our tasks and as a result of our research, a minimalist HPC cluster with manual configuration presents the ideal technology group for significant progress in the advancement of HPC at VCU within the span of the academic year.

As of yet, no significant obstacles have presented themselves in such a way that our belief in our design methodology has wavered, but in the future an alternative may be considered. Furthermore, areas of improvement are difficult to identify early in the implementation process, and as such no further recommendations can be made at this time.

Appendix 1: Project Timeline



Appendix 2: Team Contract (i.e. Team Organization)

Contents

Step 1: Get to Know Another	3
Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.	5
Step 3: Time Commitments, Meeting Structure, and Communication	7
Step 4: Determine Individual Roles and Responsibilities	8
Step 5: Agree to the above team contract	9

Step 1: Get to Know One Another. Gather Basic Information.

<i>Team Member Name</i>	<i>Strengths each member bring to the group</i>	<i>Other Info</i>	<i>Contact Info</i>
Yunus Bidav	Team collaboration, project management, creative problem solving, and technical skills	Experience with linux on server environments. Experience managing users/clients. Experience with neural networks and interpretable ML models. In depth knowledge of discrete math and its applications in computer science. ACO (Algorithms, Combinatorics, and Optimizations) Enthusiast.	Bidavye@vcu.edu 929-330-7737
Steven Holcombe	Experience with linux systems and reliability and maintenance of large multi tenant applications hosted on aws and azure (for my job). I like to learn more about most topics in computer science so I am always up for a challenge.	Reachable by discord I never check my email. I work at my job MWF 1-5 so I may be unresponsive at those times but for the most part I am highly available. Familiar with AWS, myriad of SAP products and I am pretty proficient with github. I would be able to learn more about clustering and setup/maintenance of the software side of things.	holcombes@vcu.edu 571-224-4308
James Jenkins	Enthusiasm for and experience with Linux and server operations, more free time than I ought have.	Reachable by email or Discord, will likely not respond from 10pm-7am.	jenkinsjb2@vcu.edu (540) 621-6313
Amaka Odidika	The ability to be open-minded and learn from others, computer science has also taught me to pay attention to details.	Sometimes actual text message works better for me depending on the times, usually after 9pm. I also like organization and agendas.	odidikaa@vcu.edu 240-917-6910

<i>Other Stakeholders</i>	<i>Notes</i>	<i>Contact Info</i>
Alberto Cano	<i>Faculty Advisor</i>	acano@vcu.edu
Carlisle Childress	<i>Project Advisor</i>	cgchildr@vcu.edu
Romano Woodfolk	<i>Project Advisor</i>	rmwoodfolk@vcu.edu
James Davis	<i>Project Technical Advisor</i>	jmdavis1@vcu.edu

Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.

<i>Culture Goals</i>	<i>Actions</i>	<i>Warning Signs</i>
Submit deliverables before the night they are due.	<ul style="list-style-type: none"> - Set internal deadlines - Communicate frequently progress and expectations 	<ul style="list-style-type: none"> - Unresponsiveness from team members - Infrequent progress updates on approaching deadlines
Making the effort to inform the team members if you will be unable to attend a meeting	<ul style="list-style-type: none"> - Communicate clearly what meeting dates the member will not be able to attend - Making an effort to modify the meeting time so all members may be present 	<ul style="list-style-type: none"> - Missing meetings with little to no notice - Not making an effort to modify the meeting within a reasonable window of time before the meeting - Repeated failure to attend meetings with no notice will be brought up with advisor
Every member should do their part pertaining to their role(s) and responsibilities	<ul style="list-style-type: none"> - Every member will achieve their goals in a timely manner in order to complete all milestones on time - Making sure other members are offered help incase others are falling behind 	<ul style="list-style-type: none"> - Parts of the project assigned to a member are incomplete or missing - If there are multiple missing parts pertaining to a member the advisor will be informed

Each team member should provide support while also holding one another accountable.	<ul style="list-style-type: none"> - If a team member is struggling there is an effort to assist or guide. - Ensuring each team member does their part and if otherwise accountability is taken to ensure proper support can be given. 	<ul style="list-style-type: none"> - Responsibilities are not completed and there are no reasons given
---	--	---

Step 3: Time Commitments, Meeting Structure, and Communication

<i>Meeting Participants</i>	<i>Frequency Dates and Times / Locations</i>	<i>Meeting Goals Responsible Party</i>
Students Only	Whenever needed within discord chat or voice if members are available for a short voice chat.	Simple day to day responsibility updates sent in the updates text channel of our discord. <i>Managed by Steven Holcombe</i>
Students Only	Voice call through Discord, in-person if necessary. Thursdays 5-6pm	Set responsibilities and goals for the week, provide updates to team on progress and setbacks, work on project. <i>James will manage the flow of the call while Steven will scribe.</i>
<i>Students + Faculty advisor</i>	Weekly Meeting with Faculty Advisor 6-6:45pm Thursdays	Weekly check in for questions on work being performed or next steps as needed. <i>Steven Will scribe and Yunus will manage the meeting times and changes</i>

Step 4: Determine Individual Roles and Responsibilities

<i>Team Member</i>	<i>Role(s)</i>	<i>Responsibilities</i>
Steven Holcombe	<i>Reports Manager</i>	Take meeting notes, Update github weekly with status reports.
Yunus Bidav	<i>Logistics Manager</i>	Handle meeting times. Follow up on commitments.
Amaka Odidika	<i>Financial Manager</i>	Research technical purchases and justifications for the allocated budget .
James Jenkins	<i>Project Manager</i>	Coordinate communication with Faculty Advisors/Sponsor, plan meetings and establish goals for the project. Submit deliverables.

Step 5: Agree to the above team contract

Team Member: Steven Holcombe

Signature: 

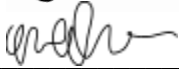
Team Member: James Jenkins

Signature: 

Team Member: Yunus Bidav

Signature: 

Team Member: Amaka Odidika

Signature: 

References

- [1] Susnjara, Susan. (2024, July 9). *What is High-Performance Computing (HPC)?*. Retrieved October 1, 2024.
- [2] VCU Research and Innovation. (2022). *HPRC Services*. Retrieved October 1, 2024.
research.vcu.edu/cores/hprc/facilities
- [3] VCU Tech Services. (2022, April 26). *High Performance Computing: IT Catalog: VCU*. Retrieved September 29, 2024. catalog.ts.vcu.edu/service/high-performance-computing
- [4] VCU Research and Innovation. (2022). *HPRC Services*. Retrieved September 29, 2024.
research.vcu.edu/cores/hprc/services
- [5] OpenHPC Project. (2024, November 7). *Install Guide (v3.2): Rocky 9.4/x86_64 + Warewulf4 + SLURM*. Retrieved June 9, 2024.