Feasibility Study Report
Project Name: Cloud Odyssey
Date: 28/03/2025
Version: 1.0

21116034	Vaishnavi Virat Dave
21116048	Kaashvi Jain
23114043	Kajal
23114052	Kumud
21323003	Abhijna Raghavendra
23114008	Anushka Jangid

1. Target Audience

Cloud Odyssey is engineered for computational researchers, academic institutions, and organizations requiring high-performance distributed computing while minimizing reliance on costly cloud-based infrastructures. The system is particularly beneficial for users seeking an adaptable and scalable computing framework that leverages underutilized computational resources.

2. Communication and Engagement Strategy

- **Stakeholder Updates:** Periodic progress reports will be disseminated via formal documentation, email briefings, and scheduled virtual meetings with the prospective users of our system. Nowadays, ML/RL/DL model development is very common and large datasets are generally required to train the neural networks or the algorithms. The very basic use case of our system is to help these programmers train their systems much more efficiently and fast, given the limited compute of their systems. As we move ahead with the project, we plan to do some industry-level analysis as well!

- **Internal Coordination:** All 6 of us will maintain synchronized workflows through **Jira**, GitHub project boards, and weekly sprint meetings.

- **Community Involvement:** The system will be maintained as an **open-source initiative**, with ongoing engagement facilitated through GitHub discussions and dedicated technical forums.

3. Project Scope and Objectives

Cloud Odyssey aims to establish a robust Beowulf cluster that enables users to:

- Establish secure SSH connections to a Master Node for executing parallel workloads using OpenMPI and SLURM.
- Dynamically adjust worker node participation based on real-time availability.
- Implement comprehensive system monitoring via Prometheus and Grafana.
- Enable remote SSH access behind NAT and firewall restrictions using LocalTunnel/ngrok.

4. Initial Requirements Assessment

- **Functional Requirements:** Secure remote access, optimized workload distribution, adaptive worker node scaling, real-time system monitoring, and cross-network accessibility.

- **Non-Functional Requirements:** Secure authentication mechanisms, scalable architecture, efficient resource utilization, structured logging, and low-latency computational processing.

5. Deliverable Roadmap

- **Phase 1:** Establishment of Master Node with SSH access and initial task distribution capabilities.
- **Phase 2:** Implementation of dynamic worker node scaling and remote accessibility protocols.
- **Phase 3:** Integration of performance monitoring and system optimization components.
- **Final Deliverable:** A fully operational distributed computing system, supplemented with comprehensive documentation and validation test cases.

6. Development Methodology

The entire software can be mainly divided into 5 microservices which are as follows:

- auth-service: Authentication for the non-master nodes before they join the cluster
- master-service: The compute usage methodology implementation after the cluster formation
- monitoring-service: The resource utilization surveillance of both master and non-master nodes
- task-queue-service: Scheduling and allotment of tasks to the available service nodes (real-time)
- worker-service: Executing the allotted tasks on the service node's compute

All these micro-services are essentially independent. Hence, their development will be carried out independently using the **iterative waterfall model**. But we will try to implement the Agile practices like pair programming, TDD and time-box based development of features.

- **Tooling:** GitHub for repository management, Docker for containerized application deployment, and Jira for project management.

7. Project Timeline

Milestone	Target Completion

Infrastructure Setup	15/03/2025
Core System Development	10/04/2025
Feature Enhancement	17/04/2025
Testing & Optimization	24/04/2025
Deployment & Documentation	01/05/2025

8. Risk Assessment and Mitigation Strategies

Identified Risk	Proposed Mitigation Strategy
Potential security vulnerabilities in SSH access	Implement robust authentication, firewall configurations, and encrypted communication channels.
Network performance degradation due to task scheduling inefficiencies	Optimize parallel processing frameworks and implement adaptive load balancing strategies.
Fluctuating worker node availability	Deploy redundancy mechanisms and fault-tolerant task execution models.
Scalability limitations in long-term adoption	Incorporate Kubernetes-based orchestration for dynamic resource allocation.

9. Technical Requirements

- **Software:** OpenMPI, SLURM, OpenSSH, NFS (Network File System), FastAPI (for high performance HTTP based service APIs), Fiber (Golang) for low memory footprint, PostgreSQL as it supports both relational and non-relational queries, Redis Streams for dynamic data analysis, Prometheus for event monitoring, Grafana for data visualization, Docker for OS-level virtualization and Docker Compose for multi-container applications.

- **Hardware:** Multi-core processors, a minimum of 8GB RAM per node, and high-speed network connectivity.

Conclusion: This feasibility study substantiates that Cloud Odyssey is a technically viable and strategically advantageous solution for distributed high-performance computing. By leveraging open-source technologies and dynamic resource utilization strategies, the system provides a cost-effective alternative to traditional cloud-based computational models.