📄 Custom Load Balancer

🔹 Project Goal

Build or extend a **Load Balancer (LB)** to distribute traffic across microservices deployed on Kubernetes or bare metal.

🔹 Basic Roadmap

Phase 1: Understanding & Setup

- Study the basics of Load Balancing and its role in real deployments.
- Explore existing open-source LB tools for reference.
- Choose a **microservice use-case** (e.g., Payment System, IoT, Analytics, Streaming).
- Decide on your **base open-source tool**.
- Plan deployment(Kubernetes or bare metal) (pods, services).

Phase 2: Core Implementation

- Implement essential LB features (routing, failover, etc.).
- Integrate with **databases + storage backend**.
- Deploy chosen microservices on Kubernetes with scaling.

Phase 3: Enhancement & Testing

- Add monitoring, logging, and dashboards.
- Run stress tests (latency, throughput, failover simulation).
- Optimize and demonstrate **autoscale up/down** (bonus points).

🔹 Target Features

👉 *You can use an existing Load Balancer and extend it by adding new features or improving existing ones.*

Default / Must-Have Features

- Routing strategies (round-robin, weighted, least-connections).
- Health checks & failover handling.
- TLS termination (basic SSL support).
- Traffic distribution across Kubernetes services.

Optional / Differentiating Features (choose a few to stand out)

- Content-aware routing (e.g., based on headers, URL paths).
- Sticky sessions (session affinity).
- Rate limiting or connection throttling.
- Multi-cluster routing (across different namespaces/clusters).
- Intelligent scaling (auto-adjusting replicas based on LB metrics).
- Observability hooks (export metrics to Prometheus/Grafana).

🔹 Storage & Database Options

Each project must integrate with **at least one database (SQL or NoSQL)** and **one distributed storage option(Open source only)**.

- **Relational DB (SQL):** PostgreSQL, MySQL, etc.
- **Non-Relational DB (NoSQL):** Cassandra, HBase, etc.
- **Distributed Storage:**
    - **Object:** Ceph RGW, MinIO, Ozone, SeaweedFS
    - **File:** CephFS, SeaweedFS.
    - **Block:** Ceph RBD, ZFS/LVM-backed volumes

🔹 Expectations

Mid Review (First Meet)

- Minimal working LB (basic routing + health check).
- Clear choice of **microservice use-case** with design diagram.
- Prototype running on Kubernetes (single-node acceptable).
- Connected to at least **one storage backend**.

Final Review

- Fully functional Load Balancer with target features implemented.
- Kubernetes deployment with **scaling + autoscaling** (bonus for scale-down).
- End-to-end demo:
- `[Client] → [Gateway] → [Custom LB] → [Microservices] → [DB + Storage]`
- Performance benchmarking & final documentation.