1. Distributed web infrastructure

For every additional element, why are you adding it:

Web Server (Nginx): The web server handles incoming HTTP requests, serves static content, and may also perform SSL termination. It helps offload some processing from the application server.
Application Server: This is where the application logic is executed. Separating it from the web server allows for better scalability and maintenance.
Load Balancer (HAProxy): Distributes incoming traffic across multiple servers to ensure optimal resource utilization, prevent server overload, and improve availability.
Database (MySQL): Stores and manages application data.

Distribution algorithm your load balancer is configured with and how it works:

Common distribution algorithms include Round Robin, Least Connections, and IP Hash. The choice depends on the specific requirements of the application. Round Robin, for instance, distributes requests equally among servers, while Least Connections sends traffic to the server with the fewest active connections.

Load-balancer enabling an Active-Active or Active-Passive setup, explain the difference:

Active-Active: Both load balancers are actively distributing traffic. This setup is more scalable and fault-tolerant but may require additional configuration to avoid conflicts.
Active-Passive: One load balancer is actively handling traffic, and the other is in standby. The passive one takes over if the active fails. It's simpler but may have higher downtime during failover.

How a database Primary-Replica (Master-Slave) cluster works:

The primary (master) node handles write operations and replicates changes to one or more replica (slave) nodes. Read operations can be distributed among replicas, improving read scalability and fault tolerance.

Difference between the Primary node and the Replica node in regard to the application:

The primary node is responsible for handling write operations and is the authoritative source for the data. The replica nodes replicate data from the primary and can be used for read operations to scale read-intensive workloads.

Issues with this infrastructure:

Single Point of Failure (SPOF):

The absence of redundancy for critical components like the load balancer, application server, or database can lead to a single point of failure, impacting overall system reliability.

Security issues (no firewall, no HTTPS):

Lack of a firewall exposes servers to potential unauthorized access. Additionally, not using HTTPS leaves communications vulnerable to interception. Implementing a firewall and enabling HTTPS is crucial for securing the infrastructure.

No monitoring:

Without monitoring, it's challenging to identify performance bottlenecks, potential security threats, or impending failures. Implementing monitoring tools allows proactive management and ensures the health and performance of the system.