3. Scale Up – Web Infrastructure Design

Domain: www.foobar.com

Infrastructure Overview (6 Servers)

Server No.	Role	Components
Server 1	Load Balancer (Primary)	HAProxy (SSL termination, load balance)
Server 2	Load Balancer (Backup)	HAProxy (clustered with Server 1)
Server 3	Web Server	Nginx (serves static files, forwards to App)
Server 4	Application Server	App runtime (e.g., Node.js, Python, PHP)
Server 5	Database Server	MySQL (Primary DB, handles all writes)
Server 6	Replica DB Server (Optional)	MySQL Replica (read scaling, backup)

Component Splitting – Why?

Component	Reason for Separation
Web Server (Nginx)	Handles static content (images, JS, CSS) and forwards dynamic requests to the application server. Offloads work from app layer.
Application Server	Executes business logic. Keeping it separate allows for better CPU/memory management and scaling.
Database Server	Centralized and optimized for data storage and queries. Reduces data layer bottlenecks.
Load Balancer Cluster (HAProxy)	Provides high availability and distributes incoming traffic. If one fails, the other takes over (Active-Passive).

Load Balancer Cluster (HAProxy)

- **Active-Passive Setup**

    - Server 1: Active
    - Server 2: Passive (monitoring Server 1, ready to take over)
- **Why Clustered?**
    - Avoids Single Point of Failure (SPOF)
    - Ensures continuous service availability
- **Load Balancing Algorithm:**
    - *Round Robin* or *Least Connections*

Application Server vs Web Server – Key Differences

Feature	**Web Server (Nginx)**	**Application Server**
Function	Handles HTTP/S requests, static files	Runs application code, handles business logic
Examples	Nginx, Apache	Node.js, Django, Flask, PHP-FPM
Request Type	Static content, routing	Dynamic content (data processing, DB access)
Performance	Optimized for concurrency	CPU/memory intensive (logic processing)

Benefits of Scaling Up

- **Improved performance** through component isolation
- **High availability** with load balancer redundancy
- **Easier maintenance and scaling** (independent upgrades or horizontal scaling)

Potential Weaknesses (Still)

Issue	**Explanation**
Single MySQL write node	Can still be a bottleneck or SPOF unless replication/failover is configured

No mention of firewalls/security layers	Still vulnerable without WAFs, security groups, or HTTPS enforcement
No monitoring setup	Without monitoring tools (e.g., Prometheus, Datadog), performance or outages might go unnoticed

Scale Up – Web Infrastructure Design