Web Infrastructure Design: Application Server vs Web Server

Overview:

This project aims to distinguish and separate the roles of web servers and application servers in a scalable web infrastructure. The components include one server, a load balancer (HAProxy), and a split configuration of web server, application server, and database, each on its own server.

Components:

1. **Web Server (WebServer):**
   - Purpose: Handles static content, serves HTML, CSS, and other client-side assets.
   - Rationale: Isolating the web server allows for efficient processing of static content, enabling quicker response times for clients.
2. **Application Server (AppServer):**
   - Purpose: Executes application logic, processes dynamic content, and interacts with the database.
   - Rationale: Separating the application layer ensures scalability, as multiple application servers can serve requests concurrently, improving overall system performance.
3. **Database Server (DBServer):**
   - Purpose: Stores and manages the application's data.
   - Rationale: Isolating the database allows for optimized performance and scalability. It also facilitates better security and maintenance practices.
4. **Load Balancer (HAProxy):**
   - Purpose: Distributes incoming traffic across web servers to ensure even load distribution.
   - Rationale: Load balancing improves fault tolerance, minimizes downtime, and enhances overall system performance by efficiently managing incoming requests.

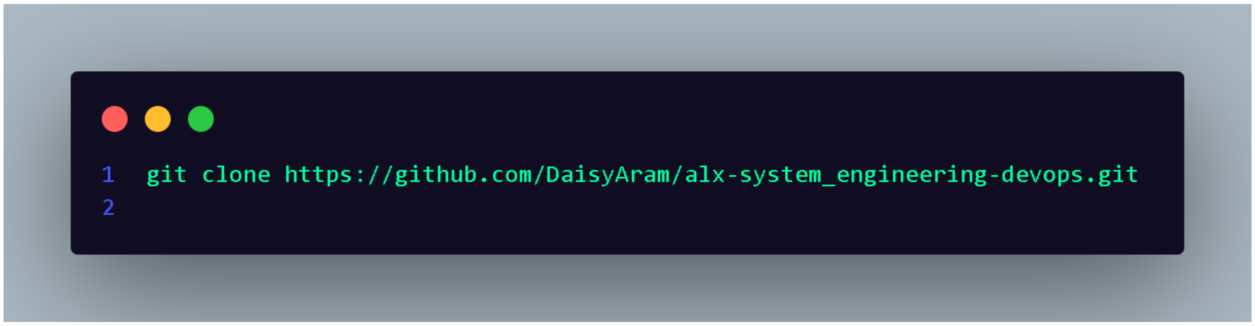
Specifics and Explanation:

- **Web Server vs Application Server:**
  - Web servers handle static content, responding to client requests for files like HTML and images. Application servers process dynamic content, executing code and interacting with databases to generate responses tailored to user requests.
- **Load Balancer Configuration:**
  - HAProxy is configured as a cluster to provide High Availability (HA). This ensures continuous service even if one load balancer node fails, preventing disruptions in traffic distribution.

- **Split Components:**
  - Splitting components onto separate servers enables scalability and maintainability. It allows for the independent scaling of web servers, application servers, and databases based on specific needs.

How to Run:

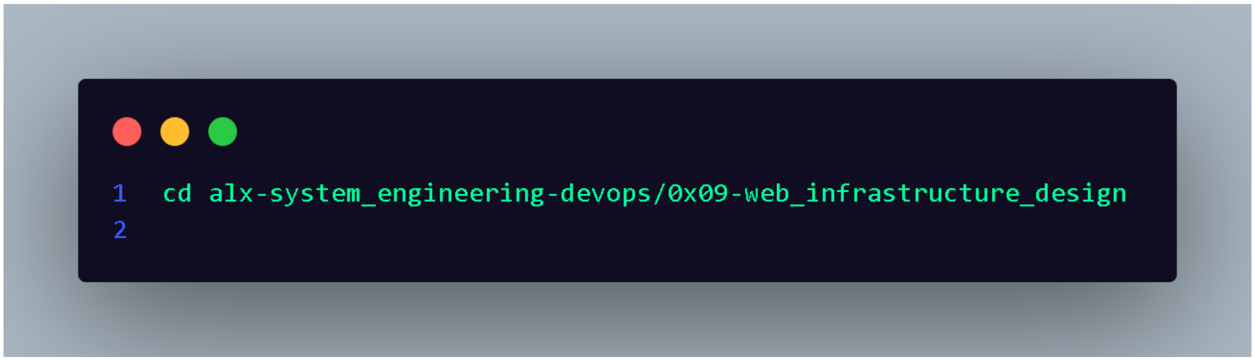
1. Clone the repository:



```
git clone https://github.com/DaisyAram/alx-system_engineering-devops.git

```

2. Navigate to the directory:



```
cd alx-system_engineering-devops/0x09-web_infrastructure_design

```

3. Open the file **3-scale_up** for detailed configurations and explanations.

Conclusion:

By adopting a split architecture with dedicated servers for web, application, and database components, and utilizing HAProxy as a load balancer, this infrastructure achieves improved scalability, reliability, and performance. Understanding the distinctions between web and application servers is crucial for designing robust and efficient web applications.