0. Simple web stack

We start with a scenario where a user wants to access the website hosted at www.foobar.com. They enter the URL into their web browser, and their computer sends a request over the internet to the server hosting the website.

1. Specifics of the Infrastructure:

Server: A server is a computer system that provides resources, services, or programs to other computers, known as clients, over a network.

Domain Name: The domain name (e.g., foobar.com) is a human-readable address that points to the server's IP address (e.g., 8.8.8.8). It allows users to access websites using memorable names instead of IP addresses.

DNS Record: The "www" in www.foobar.com is typically a CNAME (Canonical Name) record that points to the main domain (foobar.com) or directly to the server's IP address.

Web Server (Nginx): The web server handles HTTP requests from clients (such as web browsers) and serves web pages, static content, or routes requests to the application server based on configurations.

Application Server: The application server executes the application code, processes dynamic content, interacts with the database, and generates web pages dynamically.

Database (MySQL): The database stores and manages the website's data, such as user information, content, configurations, etc. MySQL is a relational database management system used to manage structured data.

Communication with User's Computer: The server communicates with the user's computer using the HTTP protocol. When a user requests a webpage, their browser sends an HTTP request to the server, which responds with the requested webpage.

2. Issues with the Infrastructure:

Single Point of Failure (SPOF): Since there's only one server, if it goes down due to hardware failure, software issues, or any other reason, the website becomes inaccessible until the server is restored. To mitigate this, redundancy measures such as backups, load balancing, and failover mechanisms should be implemented.

Downtime during Maintenance: When maintenance tasks like deploying new code require the web server to be restarted, the website may experience downtime. To minimize this, maintenance should be scheduled during low-traffic periods, and techniques like rolling updates can be employed to minimize downtime.

Limited Scalability: With only one server, the infrastructure cannot efficiently handle a significant increase in incoming traffic. To scale the infrastructure, horizontal scaling (adding more servers) and vertical scaling (increasing the resources of existing servers) strategies should be considered. Additionally, implementing load balancers can distribute incoming traffic across multiple servers.

To address these issues, a more robust infrastructure design with redundancy, scalability, and fault tolerance measures should be considered. This might involve adding multiple servers, load balancers, redundant databases, and employing techniques like containerization and automation for efficient management and scaling.