1) Are you familiar with deploying applications (ReactJS / Golang / HTML) on Linux servers?

Yes, I have experience deploying applications on Linux servers. For ReactJS applications, I typically build the production-ready bundle and serve it either through Nginx or integrate it with a backend service. With Golang applications, I have compiled and run the executable directly on Linux and then configured it to run as a systemd service for reliability. For static HTML/CSS/JS projects, I've deployed them directly under Nginx's document root or set up a lightweight HTTP server. I also understand the importance of setting up environment variables, reverse proxies, and configuring SSL for production environments.

2) Do you have knowledge with GitHub for pulling or deploying code to servers?

Yes, I am comfortable using GitHub for deployment workflows. I have used git clone, pull, and fetch commands to bring code onto a server and integrate them with automated scripts for deployment. I also have experience setting up webhooks or using CI/CD pipelines (such as GitHub Actions) to automate the deployment process, which reduces manual errors. I understand the importance of SSH keys and secure authentication when connecting a Linux server with GitHub.

3) Do you have basic knowledge of web servers like Nginx or Apache2?

Yes, I have worked with both. I usually prefer Nginx for modern deployments because of its performance and reverse proxy features. I can configure Nginx to serve static files, proxy API requests to backend services (like Node.js or Golang), and enable HTTPS with Let's Encrypt. With Apache2, I've worked on virtual host configurations and basic rewrites. I also understand log analysis for debugging issues.

4) Are you aware of DNS management (e.g., pointing a domain to a server/IP)?

Yes, I have managed DNS settings through domain registrars and services like Route 53. I know how to configure A records, CNAMEs, and MX records to point a domain or subdomain to the server's public IP. I also understand the role of TTL values and propagation time, and I've configured domains with SSL certificates to ensure secure access.

5) Do you have basic understanding of AWS services such as EC2, S3, RDS, and IAM?

Yes. I have worked with EC2 for hosting applications, configured S3 for storing static assets (like images and files), and connected applications to RDS for database management. I also understand the use of IAM for controlling permissions and ensuring least-privilege access for different users or services. For example, I've set up IAM roles to allow EC2 instances to access S3 securely without hardcoding credentials.

6) Do you have knowledge with Linux command line for server configuration and maintenance?

Yes, I am very comfortable with the Linux command line. I can navigate the file system, manage users and permissions, configure environment variables, monitor system performance with tools like top or htop, and manage services with systemctl. I also know how to configure firewalls (UFW/iptables), set up cron jobs for automation, and perform log monitoring for debugging application or server errors.

7) Are you familiar with setting up security groups or firewall rules in AWS to control access to servers?

Yes, I understand how to configure AWS security groups to restrict access by IP and port, ensuring only required traffic (e.g., SSH on port 22, HTTP/HTTPS on 80/443) is allowed. I've also worked with network ACLs and Linux-based firewalls (UFW/iptables) to implement multiple layers of security. I always make sure to follow best practices such as disabling password-based SSH logins in favor of key-based authentication.

8) Have you previously troubleshooted errors during application deployments?

Yes, I have faced and resolved several issues during deployments. For example, I've troubleshooted Nginx misconfigurations, port conflicts, incorrect environment variables, and permission errors. I usually follow a structured approach: checking application logs, web server logs, and system logs to identify the root cause. In some cases, I have rolled back to a stable version using Git or deployment scripts. I also document recurring issues and fixes to make future deployments smoother.