DEVOPS Challenges

To make it clear that the **Infrastructure** challenge is **required**, but candidates should choose only **one** from either **Automation & Development** or **System Design & Scalability**, you can revise the **Challenge Selection & Purpose** section as follows:

Challenge Selection & Purpose

As a candidate, you must complete the **Infrastructure** challenge to demonstrate your ability to provision and secure cloud environments using Infrastructure as Code (IaC).

Additionally, you must select **one** challenge from either the **Automation & Development** section or the **System Design & Scalability** section. These challenges assess complementary skills in scripting, API interactions, CI/CD pipeline design, or resource allocation for scalable systems.

Your task:

1. **Complete the Infrastructure challenge (mandatory).**
2. **Choose ONE challenge** from either:
   - **Automation & Development**
   - **System Design & Scalability**

Whichever challenge you choose, focus on **automation, efficiency, security, and scalability**—core principles of DevOps engineering. Be prepared to explain your **design choices, trade-offs, and potential improvements** in a technical discussion.

Infrastructure

Architect a **scalable, secure, and highly available web application** using Infrastructure as Code (IaC). The solution should be deployable on **Azure** (preferred) or any cloud provider of your choice.

Requirements:

- Deploy a **static web application** that serves a simple HTML page:

```
<html>
<head><title>Hello World</title></head>
<body><h1>Hello World!</h1></body>
</html>
```

- Use a **configuration management tool** (Terraform, Ansible, or equivalent) to provision and configure the infrastructure.
- Secure the web application:
  - **Restrict public access** to only the necessary ports.
  - **Enforce HTTPS** by redirecting HTTP traffic.
  - **Implement TLS/SSL certificates** (self-signed or managed).
- Ensure **scalability** by designing for high availability and auto-scaling.
- Provide **observability** with monitoring, logging, and alerting.
- Include **automated tests** to validate server configuration and security.

Deliverables:

- **Source Code**: Hosted in a public GitHub repository (`<FIRSTNAME>_DevOps_Challenge`).
- **Documentation**:
  - Overview of your design choices.
  - Deployment instructions.
  - Monitoring and scaling strategies.
- **Demo**: Be prepared to walk through your solution.

Automation & Development

Choose **one** of the following automation challenges:

1. API Automation

- Build an **API client** that iterates over a web API and retrieves paginated data.
- Implement **error handling and retries** for API failures.
- Support **rate limiting** to prevent overloading the API.
- Store the retrieved data in a structured format (JSON, CSV, or database).

2. Command Line Utility

- Develop a CLI tool that:

    - Accepts user input.
    - Stores and retrieves input data.
    - Outputs the stored data in a meaningful way.
- Apply a **real-world use case**, such as a task manager, log parser, or configuration manager.

Deliverables:

- **Source Code**: Commit to the GitHub repository (`<FIRSTNAME>_DevOps_Challenge`).
- **Documentation**:
    - Explanation of the approach.
    - Edge cases considered.
    - Any additional enhancements or optimizations.

System Design & Scalability

Choose **one** of the following system design challenges:

1. Continuous Delivery Pipeline

- Design a **CI/CD pipeline** that includes:
    - **Automated builds** and **unit testing**.
    - **Security checks and quality gates**.
    - **Multi-environment deployments** (e.g., Dev, QA, Prod).
- Provide **diagrams and documentation** outlining the architecture.

2. Scaling Model

- Perform **capacity planning** for a chat application that supports **40,000 concurrent users**.
- Estimate the **network, storage, and compute** requirements.
- Use **simple, back-of-the-napkin math** to justify the scaling model.

Deliverables:

- **Diagrams & Documentation**: Present findings in a markdown file, spreadsheet, or slide deck.

Submission Guidelines

- Host all code in a public **GitHub repository** named `<FIRSTNAME>_DevOps_Challenge`.
- Include **README documentation** with setup instructions and explanations.
- Prepare to **demo and discuss** your work.

This challenge ensures a **real-world DevOps focus** with hands-on automation, system design, and coding aspects. Let me know if you'd like any refinements!