This README file that summarizes the architecture and process for the entire assessment:

**CI/CD Pipeline and Robotic Configuration Management**

This repository demonstrates an end-to-end system for code deployment, continuous delivery, and robotic configuration management using GitLab CI/CD pipelines, Ansible, and AWS. The system supports continuous security and compliance and provides a streamlined process for developers to push code all the way to deployment.

**Architecture Overview**

The system architecture consists of the following components:

* GitLab or GitHub: Version control system used for code repository and collaboration.
* GitLab CI/CD or GitHub Actions: Continuous integration and continuous delivery pipelines.
* Ubuntu 20.04 Development Environment: Individual laptops used for development.
* AWS Cloud: Target cloud infrastructure for deployment.
* Ansible: Configuration management tool for robotic configuration and infrastructure management.
* Gradle/Maven: Build tools for building the application.
* SonarQube and Snyk: Security scanning and vulnerability detection tools.

**Process Flow**

The process flow for the system is as follows:

1. Development: Developers work on code changes and push them to GitLab/GitHub repositories.
2. Continuous Integration: Code changes trigger the CI/CD pipeline.
3. Build Stage: The application is built using Gradle/Maven.
4. Test Stage: Automated tests are run to validate the application.
5. Deployment Stage:
   * Roaming/Mobile Robot: Ansible playbook (**ansible/robot\_playbook.yml**) is executed to configure the robot based on the desired state.
   * Cloud Infrastructure: Ansible playbook (**ansible/infrastructure\_playbook.yml**) is executed to deploy the infrastructure in AWS.
6. Rollback: In case of issues or bugs, a rollback can be performed by executing the Ansible playbook with the rollback tag.
7. Security and Compliance: Security scans and compliance checks are performed using SonarQube and Snyk.
8. Feedback: Pipeline status and notifications are provided to developers and stakeholders.

**Repository Structure**

The repository has the following structure:

* **.gitlab-ci.yml** or **.github/workflows/main.yml**: CI/CD pipeline configuration file defining the stages and jobs for the pipeline.
* **ansible/**: Directory containing Ansible playbooks and related files for robotic configuration management.
* **config/**: Directory containing configuration files used by Ansible playbooks.
* **src/**: Directory containing the application source code.
* **tests/**: Directory containing automated test scripts.
* **README.md**: Documentation file providing an overview of the system architecture and process.

**Getting Started**

To get started with the system, follow these steps:

1. Set up the development environment on Ubuntu 20.04.
2. Install GitLab or GitHub for version control.
3. Set up the CI/CD pipeline using GitLab CI/CD or GitHub Actions.
4. Customize the Ansible playbooks (**ansible/robot\_playbook.yml** and **ansible/infrastructure\_playbook.yml**) for your specific configuration requirements.
5. Configure the AWS environment and provide necessary credentials for deployment.
6. Customize the application source code and tests in the **src/** and **tests/** directories.
7. Run the CI/CD pipeline to trigger builds, tests, deployment, and security scans.
8. Monitor pipeline status and receive feedback on the deployment and security/compliance status.

**Conclusion**

This system provides a seamless process for developers to push code changes all the way to deployment while ensuring continuous security and compliance. The CI/CD pipeline, combined with Ansible for robotic configuration management, offers an efficient and reliable method for managing code and infrastructure across multiple environments.

Feel free to adapt and extend the system according to your specific requirements and leverage additional tools or services as needed.