Chapter 1: Introduction to Gerrit

What is Gerrit?

Gerrit is a **code review tool** built on top of Git that helps teams collaborate on software development projects. Think of it as a sophisticated gatekeeper that ensures all code changes are reviewed before they become part of your main codebase.

Why Code Review Matters

Imagine you're building a house. Would you want the inspector to check your work only after the entire house is built? Of course not! You'd want regular inspections at each stage. Code review works the same way - it catches issues early, improves code quality, and helps team members learn from each other.

Key Benefits of Gerrit

1. **Quality Assurance**

- Catches bugs before they reach production
- Ensures coding standards are followed
- Prevents problematic code from entering the main branch

2. **Knowledge Sharing**

- Team members learn from each other's code
- Best practices spread throughout the team
- New developers get mentoring through reviews

3. **Collaboration**

- Structured discussion around code changes
- Clear approval process
- Audit trail of all changes

4. **Integration**

- Works seamlessly with existing Git workflows
- Integrates with CI/CD pipelines
- Supports automated testing

How Gerrit Differs from Other Tools

Traditional Git Workflow

```
Developer → Git Repository → Production
```

GitHub/GitLab Pull Requests

```
Developer → Fork/Branch → Pull Request → Merge → Production
```

Gerrit Workflow

```
Developer → Gerrit Review → Approval → Git Repository → Production
```

Core Concepts (Simplified)

1. **Change**

A "change" in Gerrit is like a package containing your code modifications. It's similar to a Git commit but with additional review metadata.

2. **Review**

The process where other team members examine your change and provide feedback.

3. **Approval/Rejection**

Team members can approve (+1, +2) or reject (-1, -2) your changes based on their review.

4. **Submit**

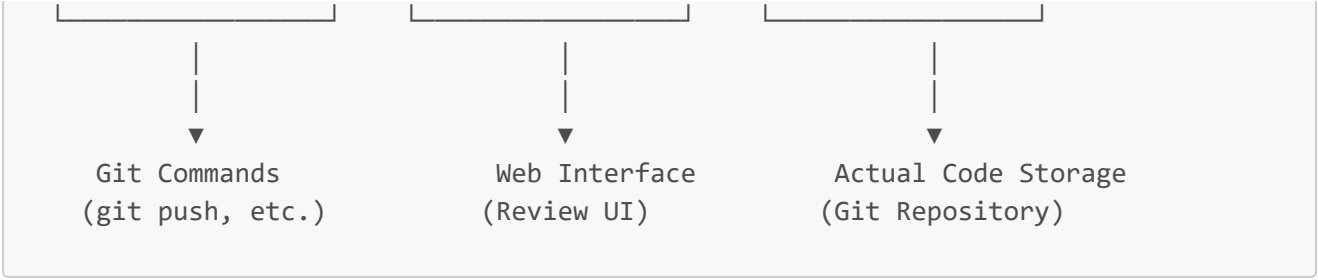
Once approved, the change is "submitted" (merged) into the target branch.

Real-World Analogy

Think of Gerrit like a **document editing process** in a professional environment:

1. **You write a document** (your code change)
2. **Submit it for review** (upload to Gerrit)
3. **Colleagues review and comment** (code review process)
4. **You make revisions** (address feedback)
5. **Final approval** (change gets approved)
6. **Document is published** (code is merged)

Gerrit Architecture Overview



Who Uses Gerrit?

- **Google**: Created Gerrit and uses it for Android and Chrome development
- **Android Open Source Project**: All Android code goes through Gerrit
- **Eclipse Foundation**: Uses Gerrit for all Eclipse projects
- **Many enterprise companies**: For internal software development

Sample Workflow Preview

Want to see how Gerrit works in practice? Check out our comprehensive Sample Workflow with Jenkins Integration that demonstrates:

- **Real team collaboration** with developers, reviewers, and DevOps
- **Complete CI/CD pipeline** with automated testing and verification
- **Human review process** with constructive feedback and iterations
- **Enterprise-grade quality gates** and approval workflows

This sample shows how all the concepts from this guide work together in a real business environment.

What You'll Learn in This Guide

By the end of this tutorial, you'll be able to:

1. **Install and configure** Gerrit from scratch
2. **Set up user accounts** and permissions
3. **Create and review** code changes
4. **Integrate Gerrit** with GitLab and GitHub
5. **Configure Jenkins** for automated verification
6. **Set up custom review labels** and submission requirements
7. **Configure enterprise-level** security and workflows
8. **Troubleshoot common issues**
9. **Implement best practices** for code review

Prerequisites Check

Before we continue, make sure you have:

- ☐ Basic understanding of Git (clone, commit, push, pull)
- ☐ Access to a computer where you can install software
- ☐ Basic command line knowledge
- ☐ Text editor or IDE of your choice

If you're missing any of these, don't worry! We'll provide quick refreshers as needed.

What's Next?

In the next chapter, we'll walk through installing Gerrit step by step. We'll start with a simple local installation and gradually build up to more complex configurations.

Ready to get started? Let's move to Chapter 2: Installation and Setup

Quick Quiz

Test your understanding:

1. What is the main purpose of Gerrit?
2. How does Gerrit fit into the software development workflow?
3. Name two benefits of code review.

Answers:

1. Gerrit is a code review tool that ensures all code changes are reviewed before being merged
2. It sits between developers and the main repository, requiring review and approval before changes are accepted
3. Quality assurance and knowledge sharing (among others)

Continue to Chapter 2: Basic Installation and Setup