## III B.Tech I Sem CSE

# 23CS503 : DevOps

## Unit-V

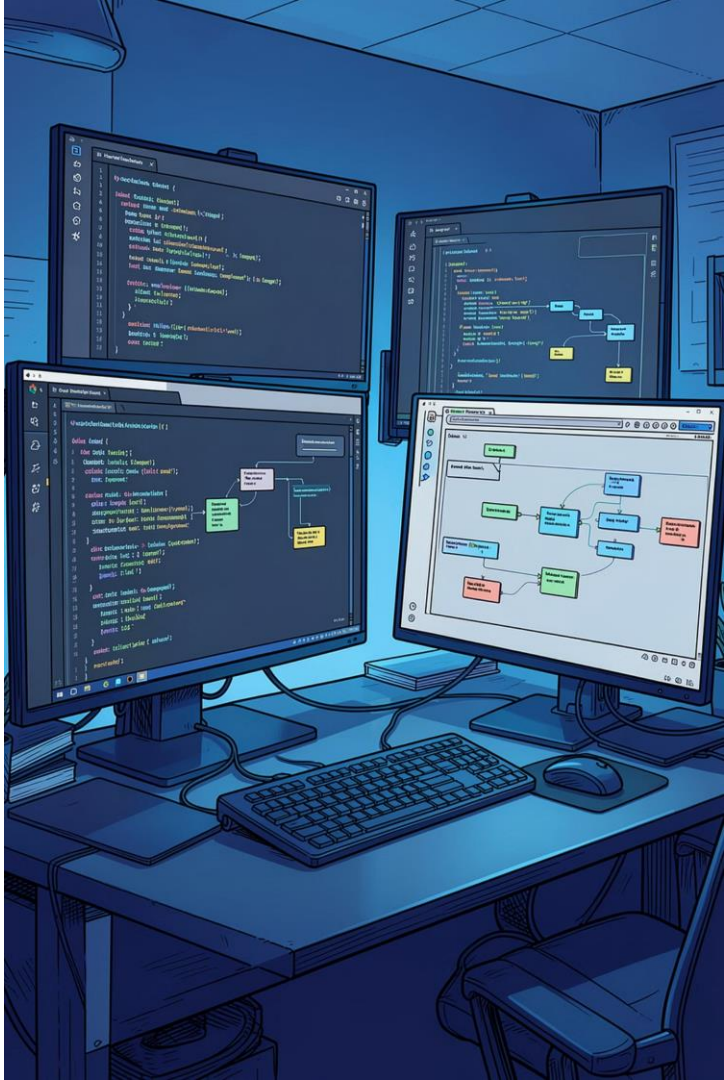
# Testing Tools and Deployment

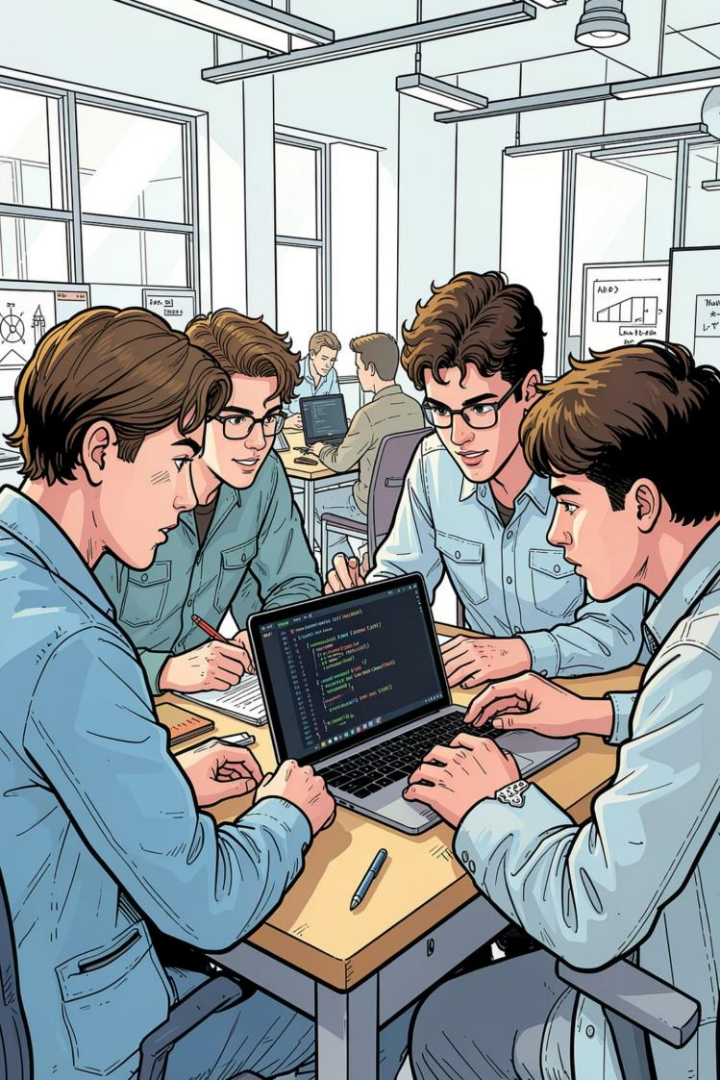
**Mr. P. Thirupathi**
**Assistant Professor, Department of CSE NRCM**

# Testing Tools and Deployment

DEVOPS — 23CS503 | UNIT – V

Department of Computer Science and Engineering | Narsimha Reddy Engineering College (Autonomous)





# Learning Outcomes

After completing this unit, you will be able to:

01

## Software Testing Methodologies

Understand and apply various testing approaches in software development.

02

## Test Automation & Selenium

Explain automation concepts and use Selenium for web application testing.

03

## Test-Driven Development

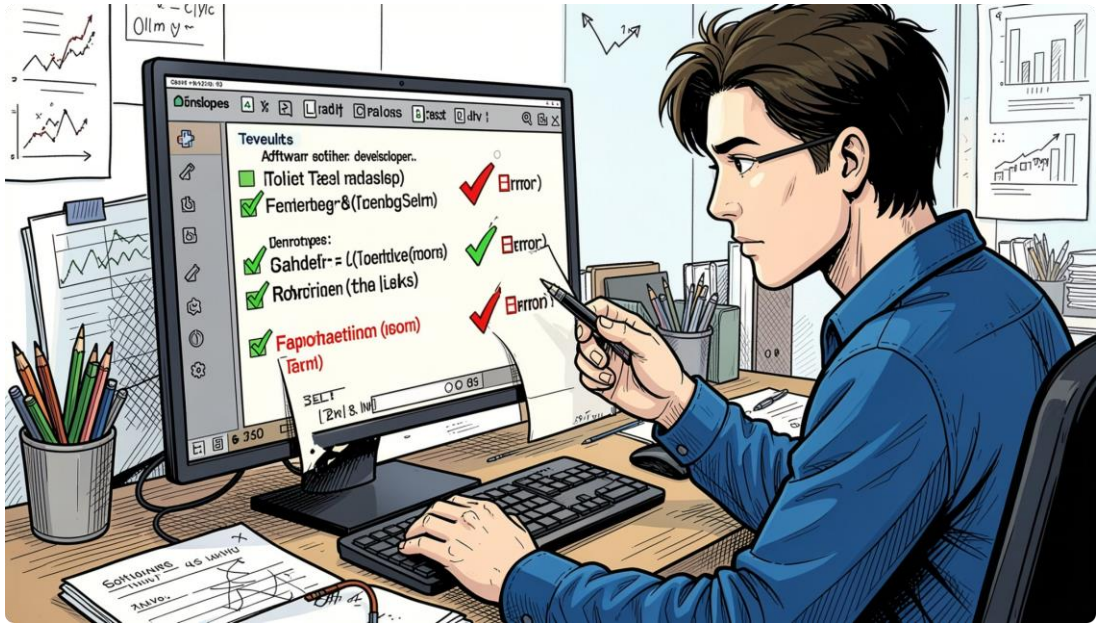
Apply TDD principles to write tests before production code.

04

## Deployment & Virtualization

Compare Puppet, Ansible, Chef, SaltStack, and Docker for deployment.

# Introduction to Software Testing



## What is Software Testing?

Software testing is the process of **evaluating software** to identify defects and ensure it meets specified requirements. It is a critical phase in the software development lifecycle.

## Key Objectives

- **Detect Defects**
- **Improve Quality**
- **Verify Functionality**
- **Ensure Reliability**

NRCM

# Why Testing Matters

### Improves Quality

Catches bugs early, ensuring a polished final product.

### Reduces Cost

Fixing defects post-release is far more expensive.

### Enhances Security

Identifies vulnerabilities before attackers can exploit them.

### Customer Satisfaction

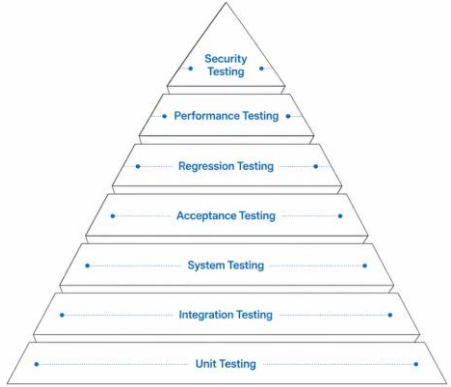
Reliable software builds trust and loyalty.

### Prevents Failures

Avoids costly downtime and reputational damage.

# Types of Software Testing

Software testing spans multiple levels — from individual components to the full system — each serving a distinct purpose in the quality assurance process.




| | |
|---|---|
| 1 | **Unit Testing**<br>Individual components |
| 2 | **Integration Testing**<br>Module interactions |
| 3 | **System Testing**<br>Complete system |
| 4 | **Acceptance Testing**<br>Business requirements |
| | **Regression Testing** |

# Unit Testing & Integration Testing

## Unit Testing

Tests **individual modules or components** in isolation to verify correctness.

- **JUnit** — Java
- **NUnit** — .NET
- **PyTest** — Python

*Benefits: Early defect detection, easier debugging*

## Integration Testing

Tests **interactions between integrated modules** to ensure they work together correctly.

### Big Bang

All modules integrated at once

### Top-Down

Tested from top level downward

### Bottom-Up

Tested from lower levels upward

### Incremental

Modules added one by one

# System Testing & Acceptance Testing

## System Testing

Tests the **complete integrated system** as a whole against specified requirements.

### Functional

Verifies all features work as expected.

### Performance

Measures response time, throughput, and load.

### Reliability & Security

Ensures system stability and data protection.

## Acceptance Testing

Validates whether the software **satisfies business requirements** and is ready for delivery.

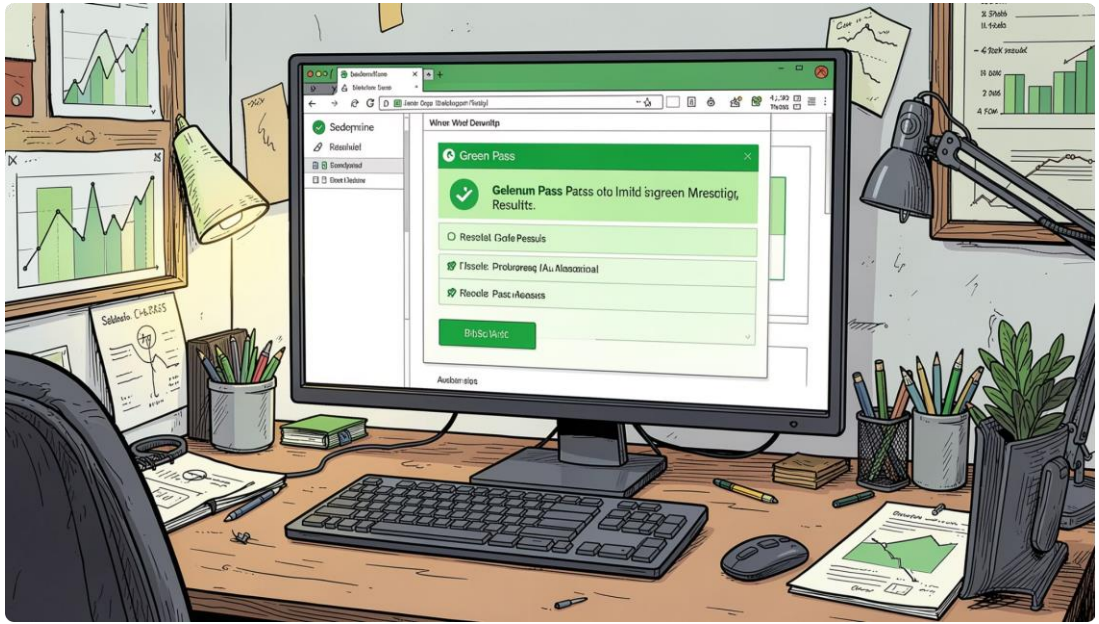
### Alpha Testing

Conducted internally by the development team in a controlled environment.

### Beta Testing

Conducted by real users in a real-world environment before final release.

# Test Automation & Selenium




## Test Automation

Using tools and scripts to **execute test cases automatically**, reducing manual effort and increasing speed and coverage.

## Selenium

An open-source framework for **automating web browsers**. Supports Chrome, Firefox, Safari, and Edge.

- **Selenium IDE** — Record & playback
- **Selenium WebDriver** — Direct browser control
- **Selenium Grid** — Parallel test execution

### Faster Execution

Run thousands of tests in minutes.

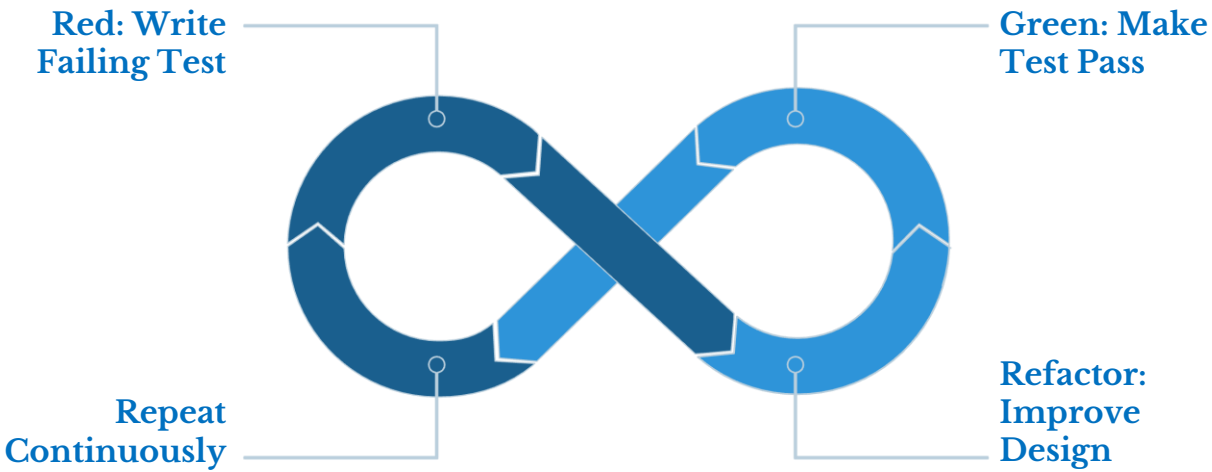
### Increased Coverage

Cover more scenarios than manual testing.

NRCM

# Test-Driven Development (TDD)

TDD is a development methodology where **tests are written before the actual code**. Developers follow a tight cycle to ensure code quality from the start.




This continuous Red → Green → Refactor loop ensures every feature is backed by a test from day one.

### Advantages

- Higher code quality and fewer defects
- Built-in regression test suite
- Encourages modular, clean design

### Disadvantages

- Steeper learning curve for teams
- Initial development time increases
- Not ideal for UI-heavy or legacy projects

# Deployment Tools & Virtualization

### Puppet

Automates infrastructure configuration using a declarative language. Ideal for large-scale environments.

### Ansible

Agentless automation using YAML playbooks. Simple setup, great for configuration management.

### Chef

Uses Ruby-based recipes to define infrastructure. Strong ecosystem for complex deployments.

### SaltStack

High-speed remote execution and configuration management. Scales to thousands of servers.

### Docker

Containerization platform that packages apps with dependencies, ensuring consistency across environments.

**Key Takeaway:** Each tool has strengths — Puppet/Chef for complex infra, Ansible for simplicity, Docker for portability, and SaltStack for speed at scale.