Site Reliability Engineering (SRE)

What is SRE?

Site Reliability Engineering (SRE) is an engineering approach to IT operations where software, automation, and coding practices are used to manage systems instead of manual work. SRE focuses on building systems that are reliable, scalable, and easy to operate. It treats operations as an engineering problem and solves it using software-based solutions.

SRE was introduced by Google to manage large-scale systems efficiently and reliably.

Why SRE?

SRE exists to solve common operational problems such as:

- Frequent system failures
- Manual operational work
- Slow recovery from incidents
- Poor system monitoring
- Difficulty in scaling systems

SRE helps organizations to:

- Improve system reliability
- Reduce downtime
- Automate repetitive tasks
- Scale systems smoothly
- Balance innovation and stability
- Improve user experience

What Can SRE Do?

SRE teams are responsible for:

- System deployment and configuration
- Monitoring and alerting
- Incident and outage management
- Performance optimization
- Capacity planning
- Automation of operations
- Reliability improvement
- Production system stability

SRE ensures that systems are stable for users while still allowing teams to release new features.

DevOps vs SRE

DevOps and SRE both aim to improve collaboration between development and operations teams, but their focus is different.

DevOps:

• Focuses on culture and collaboration
• Emphasizes fast delivery
• Strong focus on CI/CD pipelines
• Speed and automation driven

SRE:

• Focuses on system reliability
• Emphasizes stability and uptime
• Uses engineering principles for operations
• Balances innovation with reliability

SRE can be seen as a practical implementation of DevOps with a strong focus on reliability. DevOps mainly focuses on delivery speed, while SRE focuses on long-term service reliability and stability.

SRE Principles

These are the **foundational concepts** that define how Site Reliability Engineering works.
They describe the **core thinking and philosophy** behind SRE.

1. Engineering Approach to Operations

SRE treats IT operations as an engineering problem. Operational challenges are solved using software, automation, and structured engineering practices instead of manual effort.

2. Automation-First Mindset

Manual and repetitive tasks are minimized through automation. This improves efficiency, reduces human error, and increases system consistency.

3. Reliability by Design

Systems are designed with reliability as a core requirement from the beginning, not added later as a fix.

4. Measurable Reliability

System reliability must be defined using clear metrics and measurable objectives instead of assumptions.

5. Scalability

Systems must continue to perform reliably as users, traffic, and data grow.

6. Data-Driven Operations

Operational decisions are based on monitoring data, metrics, and system performance indicators rather than guesswork.

7. Balance Between Speed and Stability

Innovation and development speed must not compromise long-term system reliability and stability.

SRE Golden Rules

These are the practical rules used in daily SRE work.
They define how SRE principles are applied in real systems.

1. Error Budgets (Embracing Risk)

Failure is accepted as part of system operation. Acceptable downtime is defined using error budgets, allowing teams to balance feature releases with system reliability.

2. SLO-Based Reliability Management

Reliability is managed using Service Level Objectives (SLOs), which define clear reliability targets based on user experience.

3. Elimination of Toil

Repeated manual operational tasks must be automated to improve efficiency and reduce operational load.

4. User-Centric Reliability

System health is evaluated from the user's perspective, not just from server or infrastructure metrics.

5. Engineering Solutions Over Manual Fixes

Problems are solved using engineering solutions instead of repeated manual interventions.

6. Reliability Before Feature Releases

System stability and reliability take priority over releasing new features.

7. The 4 Golden Signals of Monitoring

System health is monitored using four key signals:

- **Latency** – Response time of requests
- **Traffic** – Demand/load on the system
- **Errors** – Failed or incorrect requests
- **Saturation** – Resource usage and capacity limits