Defect Lifecycle

1. **Detection** – Found during testing or by users.
2. **Logging** – Documented in a defect tracking tool with detailed info.
3. **Triage** – Assessed for severity, priority, and impact.
4. **Assignment** – Assigned to a developer or team.
5. **Resolution** – Fixed by developers.
6. **Verification** – Tested again by QA to ensure the fix works.
7. **Closure** – Officially marked as closed after successful testing.
8. **Reopening (optional)** – If the issue resurfaces or isn't properly fixed.

Importance of Early Defect Detection

Early detection = lower cost + higher quality.

- Reduces the risk of major system failures later on.
- Saves time and effort in debugging complex issues.
- Prevents negative user experiences and reputation damage.
- Streamlines development by minimizing rework.

Methods of Identifying Defects

Method	Description
Manual Testing	Human testers follow test cases to uncover defects.
Automated Testing	Scripts and tools catch regressions and common issues.
Code Reviews	Peer review uncovers logical flaws before testing.
Static Code Analysis	Tools scan code for vulnerabilities and errors without executing it.
User Feedback	Real-world usage reveals unexpected bugs.

Logging Defects Effectively

- **Title**: Clear and descriptive
- **Steps to Reproduce**: So others can verify it easily
- **Expected vs. Actual Results**: Highlight the deviation
- **Severity & Priority**: Indicate impact and urgency
- **Screenshots or Logs**: Provide visual/contextual evidence