Definition of Ready & Done Examples

For Scrum Teams

Definition of Ready (User Story)

A user story is ready for sprint planning when:

- ☐ **User story format**: Written as "As a [role], I want [feature], so that [benefit]"
- ☐ **Acceptance criteria**: Clear, testable criteria defined (Given/When/Then format)
- ☐ **Dependencies identified**: Any blockers or prerequisite work documented
- ☐ **Estimated by team**: Story points assigned through team consensus
- ☐ **Sized appropriately**: Can be completed within one sprint
- ☐ **Testable**: Team understands how to verify completion
- ☐ **Refined**: Discussed and understood by development team
- ☐ **Priority set**: Product Owner has assigned priority
- ☐ **No blockers**: Required resources/access available

Example Ready Story:

> As a customer, I want to reset my password via email, so that I can regain access if I forget it.
>
> **AC1:** Given I'm on the login page, when I click "Forgot Password" and enter my email, then I receive a password reset link within 5 minutes.
>
> **AC2:** Given I click the reset link, when I create a new password meeting security requirements, then my password is updated and I'm redirected to login.
>
> **Estimated:** 5 points | **Dependencies:** Email service configured | **Priority:** High

---

Definition of Done (User Story)

A user story is done when:

- ☐ **Code complete**: All code written and committed to version control
- ☐ **Code reviewed**: Peer review completed and approved
- ☐ **Unit tests**: Written and passing (>80% coverage)
- ☐ **Integration tests**: Written and passing
- ☐ **Documentation**: README, API docs, inline comments updated
- ☐ **Deployed to staging**: Successfully deployed to staging environment
- ☐ **Manual testing**: QA/exploratory testing completed
- ☐ **Acceptance criteria met**: All ACs verified and signed off
- ☐ **No critical bugs**: P0/P1 issues resolved
- ☐ **Product Owner accepted**: PO has reviewed and approved
- ☐ **Performance validated**: Meets performance requirements
- ☐ **Security reviewed**: No new security vulnerabilities introduced

---

For Kanban Teams

Definition of Ready (Work Item)

- [ ] **Clear objective**: What needs to be accomplished
- [ ] **Value articulated**: Why this work matters
- [ ] **Size estimated**: Rough effort estimate (T-shirt sizing)
- [ ] **Dependencies mapped**: Upstream/downstream dependencies known
- [ ] **Resources available**: Tools, access, information ready

Definition of Done (Work Item)

- [ ] **Acceptance criteria met**: Original requirements satisfied
- [ ] **Tested**: Appropriate level of testing completed
- [ ] **Documented**: Necessary docs updated
- [ ] **Deployed/Delivered**: In production or delivered to customer
- [ ] **Validated**: Stakeholder acceptance received

For Platform/DevOps Teams

Definition of Ready (Platform Work)

- [ ] **Problem statement**: Clear description of issue or need
- [ ] **Users identified**: Who will benefit from this work
- [ ] **Success metrics**: How we'll measure success
- [ ] **Architecture reviewed**: High-level design approved
- [ ] **Security considerations**: Security requirements identified
- [ ] **Cost estimated**: Infrastructure and operational costs known
- [ ] **Runbook exists**: Operational procedures documented

Definition of Done (Platform Work)

- [ ] **Infrastructure as Code**: All changes codified (Terraform, ARM, etc.)
- [ ] **Automated deployment**: CI/CD pipeline updated
- [ ] **Monitoring configured**: Metrics, logs, alerts in place
- [ ] **Documentation complete**: Runbooks, architecture diagrams updated
- [ ] **Security validated**: Security scan passed, secrets managed
- [ ] **Disaster recovery tested**: Backup/restore procedures verified
- [ ] **Cost monitored**: Budget alerts configured
- [ ] **Team trained**: Knowledge transfer completed
- [ ] **Production deployed**: Live in production environment
- [ ] **Post-deployment validation**: Health checks passed

Tips for Success

Making DoR Effective

1. **Keep it minimal**: 5-7 criteria maximum
2. **Make it binary**: Each item is clearly yes/no
3. **Review regularly**: Adjust based on team learnings
4. **Enforce consistently**: Don't pull unready work into sprint

Making DoD Effective

1. **Automate checks**: Use CI/CD to enforce technical DoD items
2. **Make it visible**: Post on team board
3. **No exceptions**: Done means DONE
4. **Include non-functional**: Don't forget performance, security, docs

Common Mistakes

- ❌ Too many criteria (creates friction)
- ❌ Ambiguous language ("adequate testing")
- ❌ Never reviewing/updating
- ❌ Skipping DoD under pressure
- ❌ Different definitions for different work types (unless intentional)

Customization Guide

Adapt these templates to your context:

Add for regulated industries:

- Compliance verification
- Audit trail documentation
- Regulatory approval

Add for customer-facing work:

- UX review completed
- Accessibility validated
- Localization checked

Remove if not applicable:

- Items your team can't control
- Criteria that slow you down without adding value
- Redundant checks covered elsewhere