Workflow Proof Document

Jira Bug Tracking Lifecycle – EverShop Demo Application

1. Introduction

This document provides a comprehensive workflow proof of defect lifecycle management conducted using the **Jira Bug Tracking Template** for the **EverShop Demo Web Application** (https://demo.evershop.io).

The primary objective is to illustrate a structured, real-world Quality Assurance (QA) workflow, demonstrating how defects are identified, logged, prioritized, assigned, reviewed, and resolved following industry-standard practices.

All activities documented in this workflow were performed manually within Jira, simulating a realistic QA–Developer collaboration scenario commonly adopted in software development organizations.

2. Application Under Test (AUT)

Attribute	Details
Application Name	EverShop Demo Web Application
Application Type	Web-based E-commerce System
URL	https://demo.evershop.io
Testing Type	Manual Functional & Exploratory Testing
Test Management Tool	Jira (Bug Tracking Template)

The AUT is a functional e-commerce platform with core features including product browsing, cart management, checkout processes, and user account management. Testing focused on functional correctness, usability, and responsiveness across different modules.

3. Objective of Workflow Proof

The key objectives of this workflow proof are as follows:

1. To demonstrate **proper defect lifecycle management** using Jira.
2. To show **priority-driven defect handling** aligned with business impact.
3. To simulate **collaboration between QA and Development teams**.
4. To provide **documented evidence of structured testing and reporting** in a professional QA environment.

This proof establishes adherence to best practices in defect tracking suitable for organizational-level QA processes.

---

4. Overview of Defect Lifecycle

The defect lifecycle implemented for this workflow follows a **linear, status-driven approach**:

TO DO → IN PROGRESS → IN REVIEW → DONE

Each status reflects a distinct stage in the defect management process, with detailed comments, priority assignments, and QA validation incorporated at each step.

---

5. Phase 1: Defect Identification and Logging (TO DO)

5.1 Description

During the exploratory and functional testing phase, the application was thoroughly evaluated for inconsistencies, functional failures, and UI issues.

A total of **10 defects** were identified across multiple functional areas, including:

- Product listing and categorization
- Cart management and checkout workflow
- UI responsiveness on various screen sizes
- Search and filter functionality

5.2 Jira Actions Performed

Each identified defect was logged as a **Bug** issue type in Jira, including:

- **Clear and concise summary**

- **Environment details** (browser, OS, device)
- **Steps to reproduce**
- **Expected vs. actual behavior**
- **Severity and priority classification**

All defects were initially configured as:

- **Status:** TO DO
- **Assignee:** Unassigned

5.3 Purpose

This phase represents the **QA responsibility for defect discovery and documentation**, ensuring that all issues are clearly captured before assignment to developers.

---

6. Phase 2: Priority-Based Assignment (IN PROGRESS)

6.1 Description

Logged defects were reviewed and **prioritized based on business impact and severity**:

- **High Priority:** Critical defects impacting core functionality, requiring immediate resolution.
- **Medium Priority:** Non-critical defects affecting user experience but not core operations.
- **Low Priority:** Minor cosmetic or edge-case issues deferred for later resolution.

6.2 Jira Actions Performed

High-priority defects were transitioned to:

- **Status:** IN PROGRESS

Developers were assigned (simulated) to these defects, and comments were added to reflect work commencement:

“Developer has started working on the issue. Investigation and fix are in progress.”

Medium and low-priority defects remained in TO DO, representing backlog items.

6.3 Purpose

This phase demonstrates **realistic sprint and capacity planning**, where high-priority issues are addressed first while less critical defects remain deferred.

---

7. Phase 3: Review and Parallel Development (IN REVIEW & IN PROGRESS)

7.1 Description

As development progressed:

- **High-priority defects** were moved to **IN REVIEW** after fixes were applied.
- **Medium-priority defects** transitioned to **IN PROGRESS**.
- **Low-priority defects** remained in **TO DO**.

This mirrors Agile development environments, where multiple defects are handled in parallel based on priority.

7.2 Jira Actions Performed

Status updates and comments were recorded:

- **High → IN REVIEW**
- **Medium → IN PROGRESS**
- **Low → TO DO**

QA validation readiness comments were added:

“Fix has been completed by developer and is ready for QA review.”

7.3 Purpose

This phase reflects **parallel development and review processes**, common in continuous delivery and Agile workflows, ensuring that QA can validate fixes while medium-priority defects are concurrently addressed.

---

8. Phase 4: Fix Completion and Validation (DONE)

8.1 Description

Upon QA verification:

- **High-priority defects** were marked **DONE** after validation.
- **Medium-priority defects** were moved to **IN REVIEW** for QA verification.
- **Low-priority defects** remained deferred for future sprints.

8.2 Jira Actions Performed

Status updates were applied to reflect completion:

- **High → DONE**
- **Medium → IN REVIEW**
- **Low → TO DO**

QA validation comments confirmed issue resolution:

“Fix completed and validated by QA. Issue behaves as expected.”

8.3 Purpose

This phase demonstrates **QA responsibility in confirming defect resolution**, ensuring that all completed fixes meet expected standards before closure.

---

9. Evidence and Workflow Screenshots

Screenshots were captured at every stage to provide evidence of a structured defect lifecycle:

1. **Defects logged in TO DO**
2. **High-priority defects moved to IN PROGRESS**
3. **High-priority defects in IN REVIEW, medium-priority in progress**
4. **High-priority defects marked DONE, medium-priority under review**

These visuals collectively validate the complete lifecycle and adherence to company-level QA standards.

---

10. Key Learnings and Best Practices

Through this workflow, the following QA best practices were applied:

- **Comprehensive defect documentation** for reproducibility
- **Priority-based defect handling** to align with business impact
- **Clear communication** between QA and Development via Jira comments
- **Sprint-aligned defect management**, simulating real-world capacity planning
- **End-to-end tracking of defects** from identification to closure

---

11. Conclusion

This workflow proof demonstrates a **professional Jira-based defect management process** for a real-world e-commerce application.

The exercise highlights the ability to:

- Identify and document defects clearly and thoroughly
- Manage defect priorities and transitions effectively
- Collaborate efficiently with development teams
- Perform QA validation and ensure proper closure of issues