INSTRUCTURE

QUALITY ASSURANCE PROGRAM

Engineering, Security, and Operations

May 2024

Table of Contents

INSTRUCTURE

Introduction

Overview

At Instructure, we care deeply about creating quality products and delivering them to our customers with the smoothest of experiences humanly possible. This is why when it comes to the Software Development Life Cycle (SDLC), we apply Agile principles and methodologies with an integrated Quality Assurance (QA) process to the design, development, and maintenance of our products.

Our developers operate in either Scrum or Kanban teams organized around product components and features, with dedicated QA processes and personnel integral to each team. All functional code changes are required to be thoroughly reviewed and tested by one or more engineers other than the author. All code changes are also required to have appropriate automated tests written before the change is accepted, both for new features and bug fixes.

Once new code has passed peer review, the code is incorporated into the code base and submitted to testing and quality assurance. The new code is deployed to a continuous integration server where it is immediately tested. Instructure's testing team runs the following types of tests:

- Unit tests (testing code with code)

- Integration tests (testing code with integrations with other code)

- Browser tests (testing how code works in the browser) on all the different environments and across different databases.

After passing these tests, the code is incorporated in the master code branch for formal quality assurance. The QA team tests the new code on all supported platforms and browsers. Any bugs or defects that are found must be fixed during this formal QA period, otherwise the change is pulled for further development.

Integrated QA Process

Integrated with the Agile development methodology, Instructure's QA process tests and validates the functionality, reliability, and performance of individual features. Typically, new development will undergo four stages of the test process.

- Test Approach Definition (per project/epic) - to establish the overall strategy and scope of testing. Artifacts include the project Test Plan.

- Test Requirements & Planning (per feature/user story) - to identify the specific activities and resources required to apply the defined test approach. Artifacts include test-specific development stories and acceptance criteria.

- Test Development and Execution (per story or code change) - to manage and conduct the testing including. Artifacts include automated tests, manual test cases, test plans, defects, and documentation.

- Regression Testing. Artifacts include Test Run results and defect tickets.

The QA process is a dynamic process and not a strict workflow. As requirements, schedules, or resources change, the development teams regularly adjust test plans.

Testing

Testing Approach

Some teams approach large new development projects, while others operate on maintaining and iteratively improving existing features. All teams determine, either on a per-project or ongoing basis, their strategy and overall approach for ensuring the quality of their work. While some processes are common across all development teams, individual teams are responsible for determining and implementing the best processes for their individual previews.

Teams are expected to document their approach in a Project Test Plan which may include the specifics of:

- Automation technologies

- Automation methodologies (unit, integration, UI, etc.)

- Performance testing requirements

- Manual testing requirements

- Code review standards

- Definition-of-done (for both the project and guidelines for individual stories within a project)

Test Requirements and Planning

As requirements are defined more granularly in preparation for inclusion in a Scrum sprint or placement on the Kanban board, specific Acceptance Criteria are developed. The criteria represent the team's agreement on what must be accomplished before a ticket is considered "done" and will often include an outline of what types of testing must be performed and what automated tests must be written and integrated into automated testing suites in conjunction with the ticket. These criteria are developed as part of Backlog Grooming, Sprint Planning, and Defect Triage meetings with involvement by representatives of all disciplines.

Regression Testing

Regression testing is the ongoing, regular testing of developed features to ensure that bugs are not introduced over time. The automated tests built during the development cycle are run regularly and applicable tests related to application functionality are run any time new changes are proposed. Manual regression testing is performed as needed for cases that are difficult or impossible to automate.

Test Development and Execution

Manual testing is performed and automated tests are written as functional code changes are being made. The specific test activities required will vary based on the code in development, but may include:

- Designing and writing automated tests - unit, integration, UI, etc.

- Generating data and environments required for testing

- Executing performance / load tests

- Executing manual tests and exploratory testing

- Sign-off when acceptance criteria have been met

Testing Scope

Definitions

The following defines our testing scope:

- Unit Testing - Validation of individual units of product code.

- Integration Testing - Verifies functionality and reliability of the product code as it relates to other modules as well as the database and the API layer.

- UI Automation - Automation of end-to-end tests at the UI layer, conducted in a web browser.

- Functional Testing - Verifies that each feature of the application meets its requirements. For example, a "Create" button should trigger the appropriate event when pressed.

- User Acceptance Testing - Verifies that the product/project works in the customer expected manner. End-to-end testing from the UI layer.

- Performance Testing - Ensures that the system meets defined response times.

- Load Testing - Creates a load greater than the expected customer demand on the system.

- Ad-Hoc or Exploratory Testing - Exploratory, unstructured testing that attempts to find flaws and break the system.

- Multi-User Testing - Similar to load testing but assures that an acceptable number of users can work within the system simultaneously.

- Regression Testing - Regular, repetitive testing (both automated and manual) of existing features.

- Smoke Testing - Validation of major feature functionality after a deployment to a new environment.

- Product Certification - Certification by the Product team that the product/project meets objectives and requirements of the Test Plan.

- Accessibility (a11y) Testing - Ensures that product features and functionality are available to users with visual, physical, cognitive, or other disabilities.
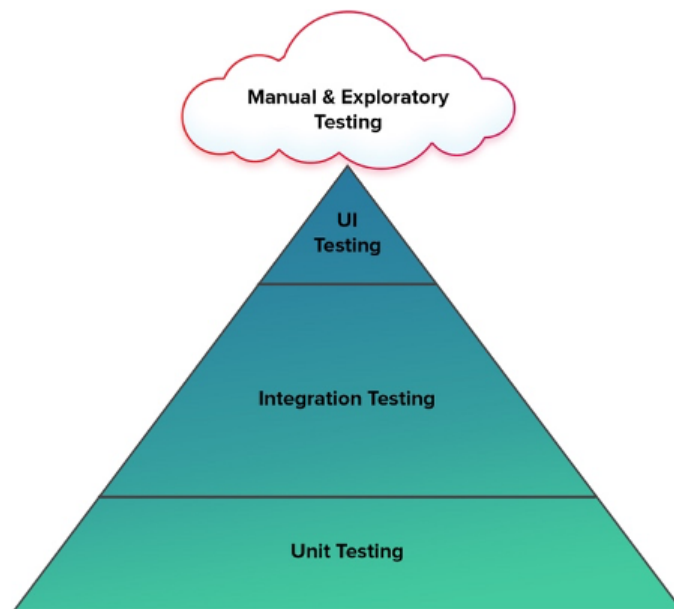
Automated Testing

Automated testing is a critical part of the Instructure quality plan. Instructure prioritizes tests that can be executed quickly and reliably, according to the traditional automation pyramid.

Unit tests should represent the vast majority of automated tests, as they run the most quickly and efficiently. Integration tests may include tests of large subsystems or service endpoints and should represent the next largest body of automated tests. UI Automation tests are run using applicable testing technologies and should be limited to tests of main "Happy Path" functionality.

Additional automation may be created for performance and load testing.

Manual Testing

Manual Testing

Manual testing will be done for all test cases that cannot be automated in addition to user interface (UI) test cases. Manual exploratory testing is conducted throughout new feature development.

Product Development Testing Checklist

The following are taken into consideration as features and products are developed:

- **FERPA** issues

- **Accessibility (WCAG 2.1 Standards, screen readers, color contrast, etc.)**

- **Checking features from different roles (e.g., Admin, Teacher, Student)**

- **Monitoring Permission issues (i.e., a user seeing too much or too little)**

- **Internationalization**

- **XSS**

- **SQL injection**

- **Documentation**

- **Pagination**

- **Mobile browsers / mobile impact**

- **Old data**

- **Sharding**

- **API**

- **Course copy/export**

- **Browser dimensions**

- **Performance tested**

Accessibility Testing

Ensuring an accessible and pleasant experience to all users, regardless of disability, is a key focus of Instructure. Our programs are built using the most modern HTML and CSS technologies and we are committed to W3C's Web Accessibility Initiative and Section 508 guidelines.

Accessibility training is conducted for all engineering staff. Instructure is committed to all new features being fully accessible before being deployed to production. Current commitments to Accessibility standards are documented publicly on our website.

User Acceptance Testing (UAT)

Traditionally, User Acceptance Testing (UAT) is the last milestone before a system is placed in production for its users, where end-users and stakeholders verify and sign off on the product being a suitable fit to their specifications. As cloud-based SaaS solutions, and as outlined in this paper, Instructure ensures a continuous testing program for its products to meet the needs of our customers - especially when it comes to software quality and fit for purpose. During a product implementation, a customer may wish to run their own UAT to ensure their specifications are met. On an ongoing basis, through a customer's non-production instance (if applicable), users can also conduct UAT on new features before they reach production.

Web Browser Support

Our products support the last two versions of major web browsers, with further details regularly updated in our public documentation.

Limitations in automation technology prevent automation from running on older browser versions. As such, test coverage against supported browser versions comes via manual exploratory and regression testing.

Conclusion

We recognize mistakes can happen in the Software Development Life Cycle. After all, we're only human. If a defect accidentally sneaks into our code, we know it can cause trivial or serious consequences. This is why we place great importance on QA Testing. We want our code to run as smoothly as possible for our customers, and that's why we take extreme care to implement both preventative and detective mechanisms throughout the SDLC, with an integrated QA process to the design, development, and maintenance of our products. The bottom line for our customers: all code changes run through our full QA test suite before they can be accepted into the relevant product to ensure secure code, consistent performance, and a great all-round experience to keep on learning.

INSTRUCTURE