QA Dashboard Automation – The DevOps Way

Problem Statement:

A 7-day agile sprint approach was resulting in production ready codes getting delivered only after the culmination of the sprint. It was a double whammy: besides the time that was consumed, the resulting financial impact was significant. A suggestion was made to embrace the DevOps approach to address this otherwise ubiquitous issue.

The Significance of the Approach:

DevOps makes it possible to increase the velocity at which companies can consistently deliver software applications and services. While different definitions exist for DevOps, the core idea is the same - combining elements of Software development and IT ops for faster and assured project delivery. The core idea is to manage end-to-end engineering processes.

Overview:

The goal of the project was to expedite the test cycles and in the end make deployment of the build faster to the end users. Earlier the release cycle of a client took more than a week which included testing of an application manually, updating report sheets and then finally sharing the results with the business stakeholders. The solution that was proposed and finally implemented at client end resulted in a daily release cycle.

Client Details:

Name: Confidential | **Type**: Social Media IT | **Location**: USA

Technologies:

Python, MongoDB, Jenkins CI/CD, Selenium WebDriver, Cucumber

mindfire
BURN IGNORANCE

QA Dashboard Automation – The DevOps Way

Project Description:

Technology is rapidly evolving and with it, Quality Control has become more complex and time consuming. Testing of every feature manually takes more than the benchmark time, resulting in delays in fixing of issues and in the end a delay in releases; which affects businesses adversely. To overcome this, the client for this project built an innovative solution with the Mindfire team to execute automated regression suite on a daily basis which resulted in improving the time estimation of the whole Quality assurance process. The solution included the use of Cucumber Selenium to execute the automation suite and the detailed reports that were generated for each test case were then crawled into the backend. It displayed the unified reports to stakeholders which enabled them to decide the health of a build in progress on a daily basis. It also allowed the managers to search, sort and filter the projects using unified reports. In the case of functionality failures, the manager and developers were alerted by email in real-time which gave visibility and increased efficiency.

The client thus follows an agile process and the solution gives the managers a real-time picture of the quality of a live code base. Whenever a build has to be deployed, instead of the traversing the long lists of entries in excel sheets to determine which functionalities are failing and which aren't, the client can quickly check the unified reports (as shown in Diagram 3**)** and decide if the build is stable enough to be deployed.

The solution developed by Mindfire benefited the client greatly in time management and made releases possible within a day. This solution is highly configurable as it is cloud-enabled and can easily accommodate new projects.

QA Dashboard Automation – The DevOps Way

Previous Workflow:

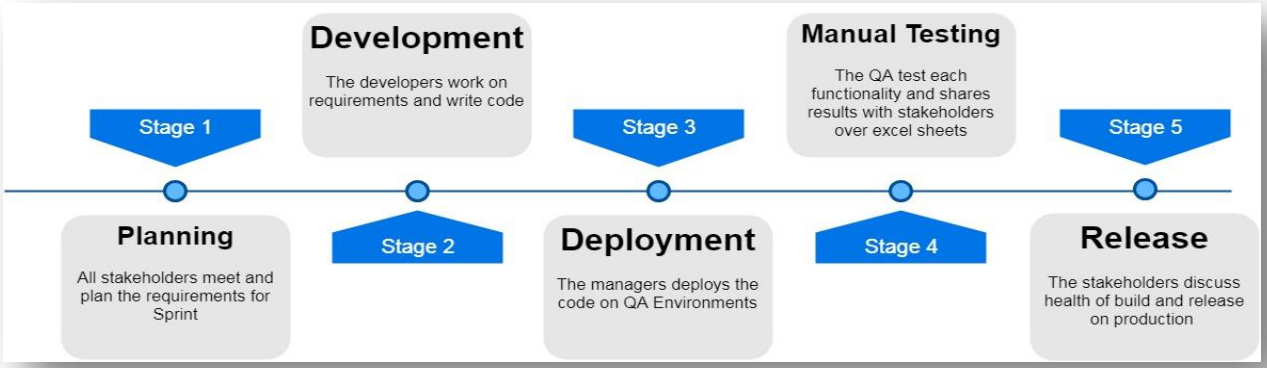


Diagram 1: Previous workflow

Challenges:

The major challenge with the previous workflow was the test results flow and quality delaying the release cycles. The QA was spending time creating and maintaining excel sheets and visibility was poor. Business stakeholders would know the results only when the regression cycles were over and if major issues were found, the release would often miss deadlines. Data representation was also a challenge; for instance, if an issue was found, it would take time to decide if it was a new issue or an existing issue. Also there was a need for the solution to not just present unified reports on a daily basis, but also provide quality data analysis for every issue with minimum effort.

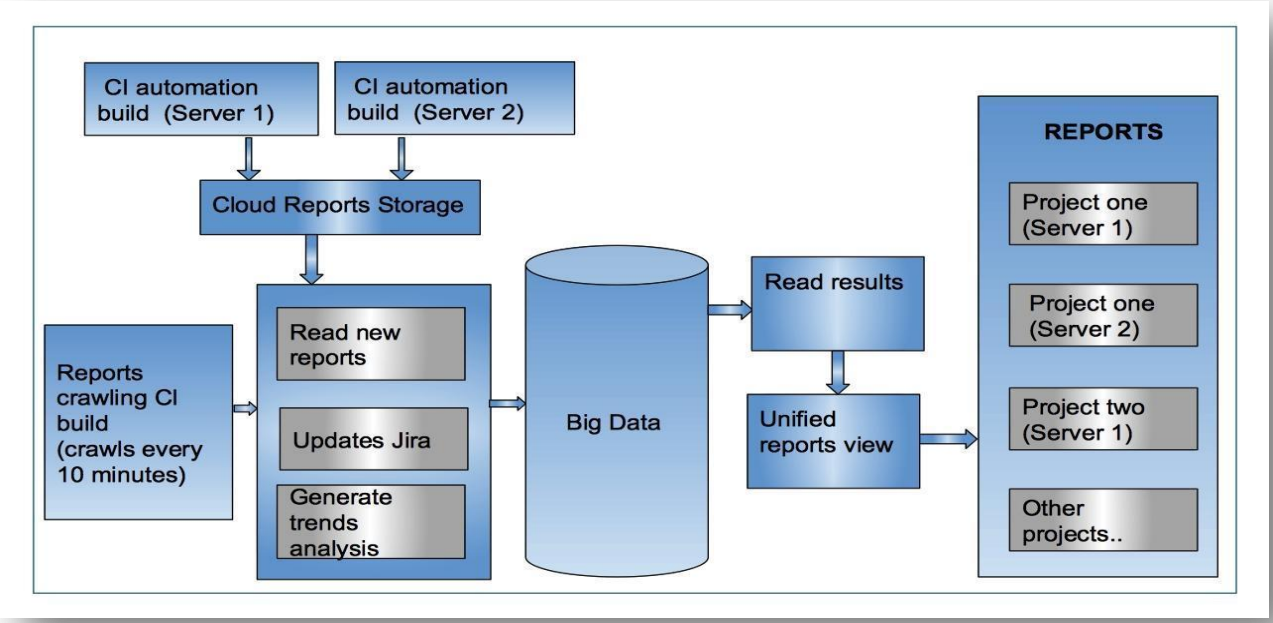
QA Dashboard Automation – The DevOps Way

Proposed Solution:

The proposed solution recommended the following features:

1. **Automatic deployments:** The code should be deployed on fixed intervals automatically using CI/CD tools.
2. **Automation testing:** Once a build is deployed, automation suite should be triggered to check the status of the build and push the results into unified reports.
3. **Unified reports:** Stakeholders should easily be able to access unified reports to monitor the performance of build daily.
4. **Data representation:** The data representation in solution should be granular so that interested parties can not only just view combined results but also every individual component being tested.
5. **Data analysis:** The solution should show historical results and daily trends in a meaningful representation.

Architecture:



QA Dashboard Automation – The DevOps Way

Diagram 2: Solution Architecture

This architecture shows end-to-end solution. Test Environments of Server 1 & 2 are hosted on Cloud, Cloud reports storage is on shared storage, Crawlers and Big Data is hosted on separate Cloud Servers which display the unified reports of all projects

Screenshots:

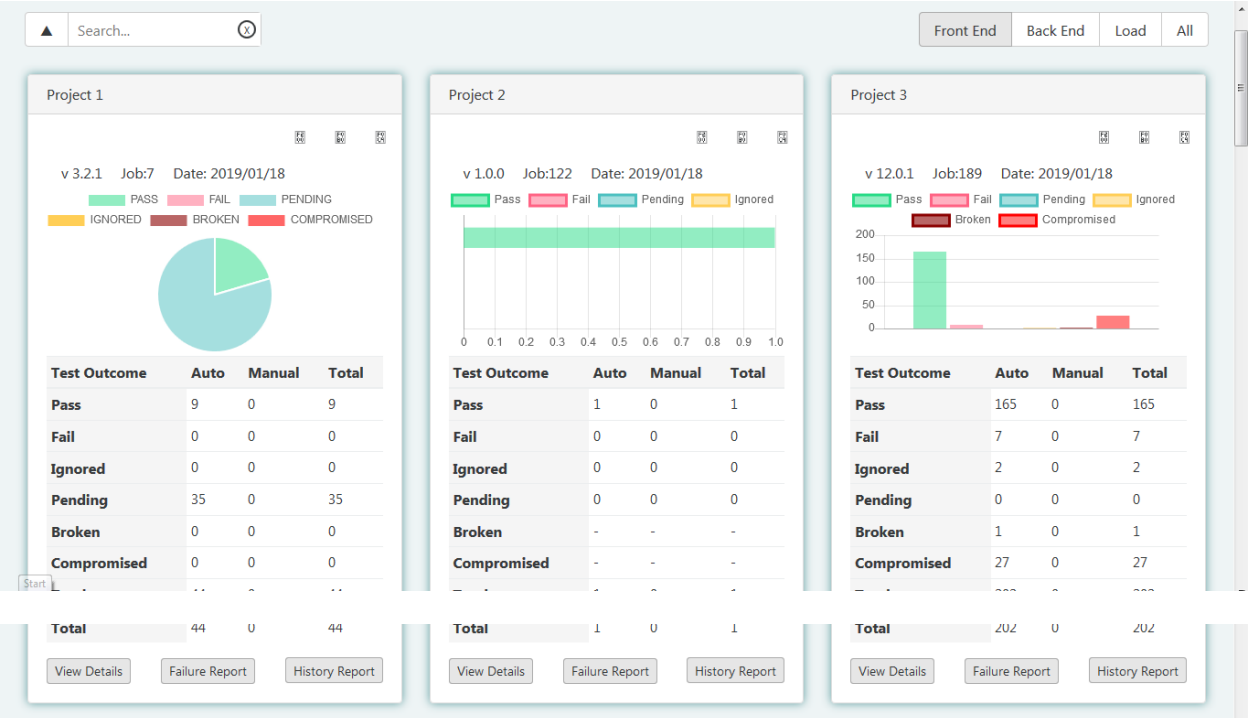


Figure 3: Landing Page

The landing page contains all project cards with current build results. This includes search, sort and filter functionalities present in the top menu. Every project card has
“View Details” button to check detailed build results.
“Failure report” to check individual test results.
“History report” to check the trends of every test case from past 20 days.

mindfire
BURN IGNORANCE

QA Dashboard Automation – The DevOps Way

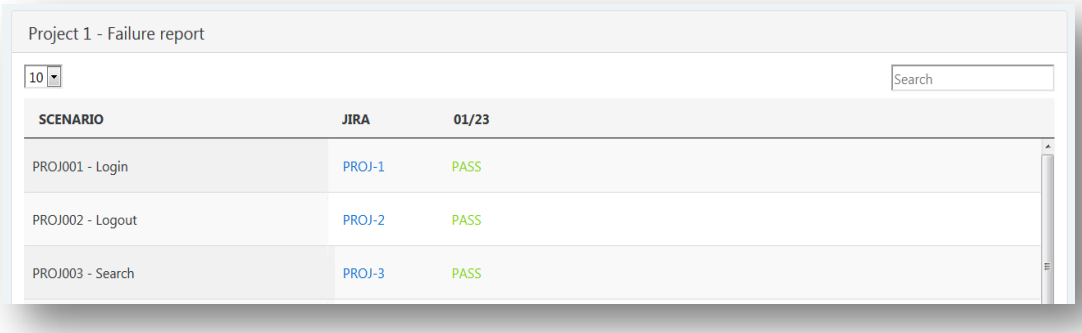


Figure 4: Failure Report

This page contains all individual tests executed with results.

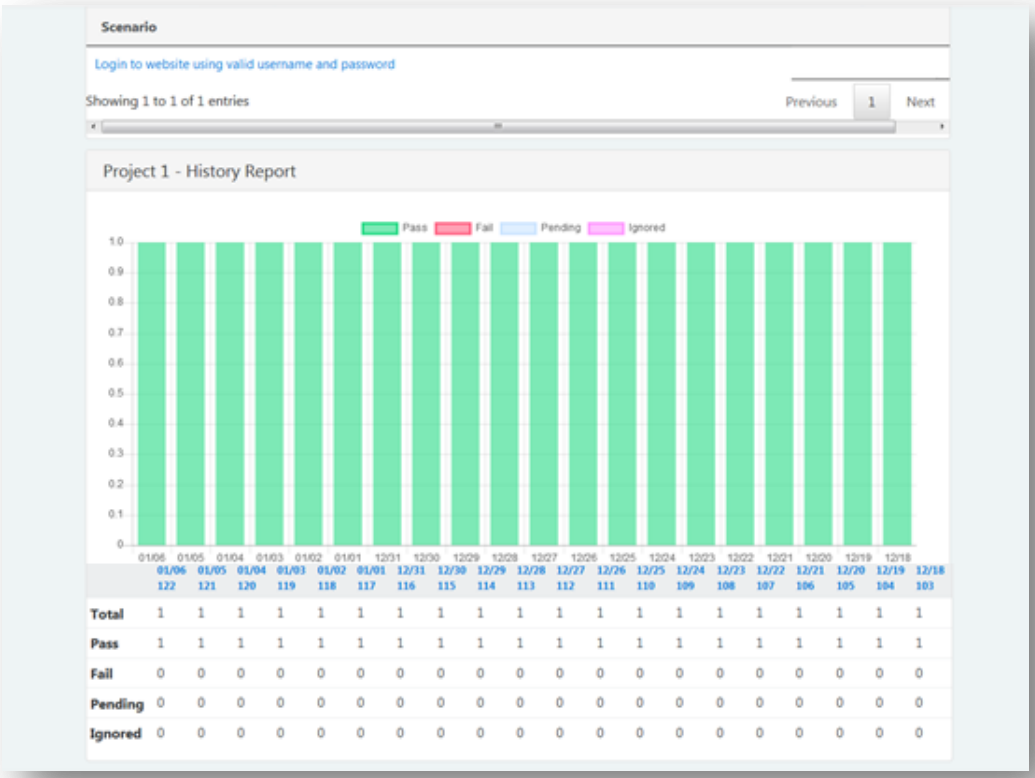


Figure 5: History Report

This shows trends and pattern of previous execution of test cases.