Microsoft

# AZ-400.2 Module 02: Managing Code Quality and Security Policies

## Lesson 01: Managing Code Quality

## Lesson 1 Overview

- Code Quality Defined
- Sources and Impacts of Technical Debt
- Using Automated Testing to Measure and Monitor Technical Debt
- Configuring SonarCloud in a Build Pipeline
- Reviewing SonarCloud Results and Resolving Issues
- Integrating Other Code Quality Tools
- Code Quality Tooling
- Managing Technical Debt with Azure DevOps and SonarCloud

## Video: Code Quality Defined

Short deadlines, a lack of coding standards, and poor technical skills can lead to  code that is NOT:

- Clear and readable
- Documented
- Efficient
- Maintainable
- Extensible
- Secure

## Sources and Impacts of Technical Debt

- Technical Debt describes the future penalty that you incur today by making easy or quick choices in software development practices.
- Common sources of technical debt are:
  - Lack of coding style and standards
  - Lack of or poor design of unit test cases
  - Ignoring or not understanding object orient design principles
  - Monolithic classes and code libraries
  - Poorly envisioned use of technology, architecture and approach
  - Over-engineering code
  - Insufficient comments and documentation
  - Not writing self-documenting code
  - Taking shortcuts to meet deadlines
  - Leaving dead code in place

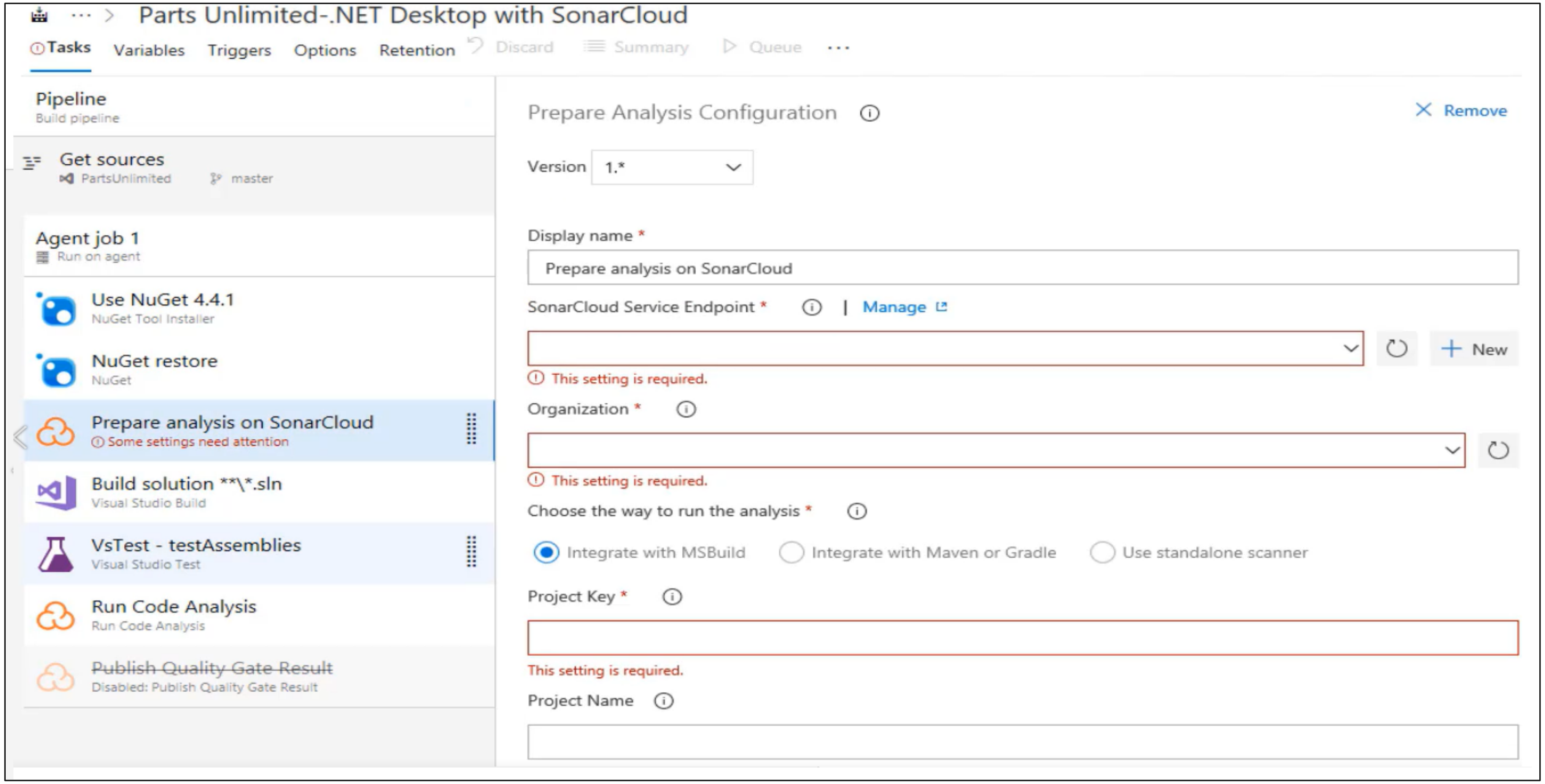
## Using Automated Testing to Measure Technical Debt

Technical debt:

- Adds problems during development that makes it more difficult to add customer value
- Saps productivity and frustrates development teams
- Makes code both hard to understand and fragile
- Increases the time to make changes, and to validate those changes
- Starts small and grows over time

✓ One way to minimize the accumulation of technical debt, is to use automated testing and assessment

# Demonstration: Configuring SonarCloud in a Build Pipeline



Parts Unlimited-.NET Desktop with SonarCloud

Pipeline
Build pipeline

Get sources
PartsUnlimited master

Agent job 1
Run on agent

Use NuGet 4.4.1
NuGet Tool Installer

NuGet restore
NuGet

Prepare analysis on SonarCloud
Some settings need attention

Build solution ***.sln
Visual Studio Build

VsTest - testAssemblies
Visual Studio Test

Run Code Analysis
Run Code Analysis

~~Publish Quality Gate Result~~
Disabled: Publish Quality Gate Result

Prepare Analysis Configuration

Version 1.*

Display name *
Prepare analysis on SonarCloud

SonarCloud Service Endpoint * | Manage

This setting is required.

Organization *

This setting is required.

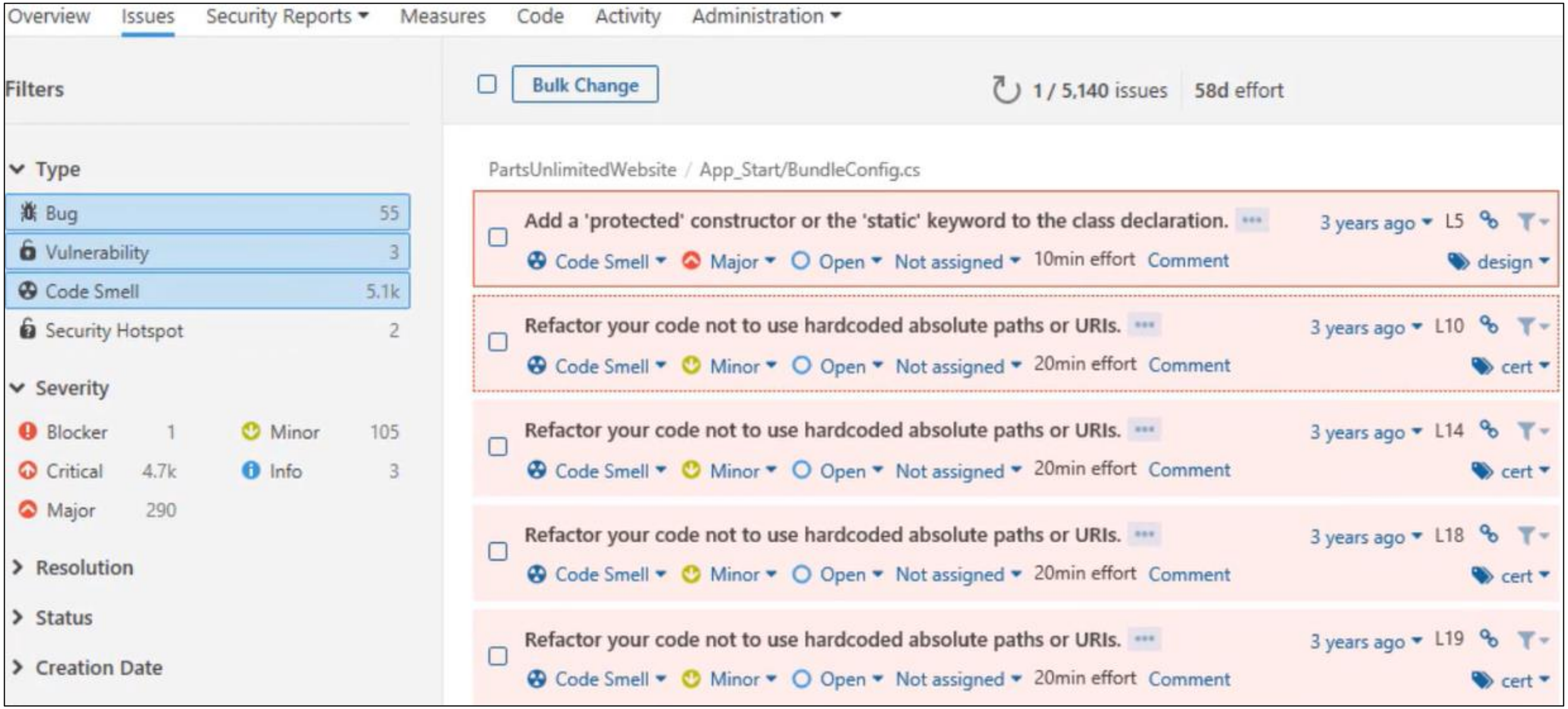
Choose the way to run the analysis *
[✓ Integrate with MSBuild] Integrate with Maven or Gradle Use standalone scanner

Project Key *

This setting is required.

Project Name

# Demonstration: Reviewing SonarCloud Results and Resolving Issues



Filters

Type

| | |
|---|---|
| Bug | 55 |
| Vulnerability | 3 |
| Code Smell | 5.1k |
| Security Hotspot | 2 |

Severity

| | | | |
|---|---|---|---|
| Blocker | 1 | Minor | 105 |
| Critical | 4.7k | Info | 3 |
| Major | 290 | | |

Resolution

Status

Creation Date

1 / 5,140 issues 58d effort

PartsUnlimitedWebsite / App_Start/BundleConfig.cs

Add a 'protected' constructor or the 'static' keyword to the class declaration. 3 years ago L5
Code Smell Major Open Not assigned 10min effort Comment design

Refactor your code not to use hardcoded absolute paths or URIs. 3 years ago L10
Code Smell Minor Open Not assigned 20min effort Comment cert

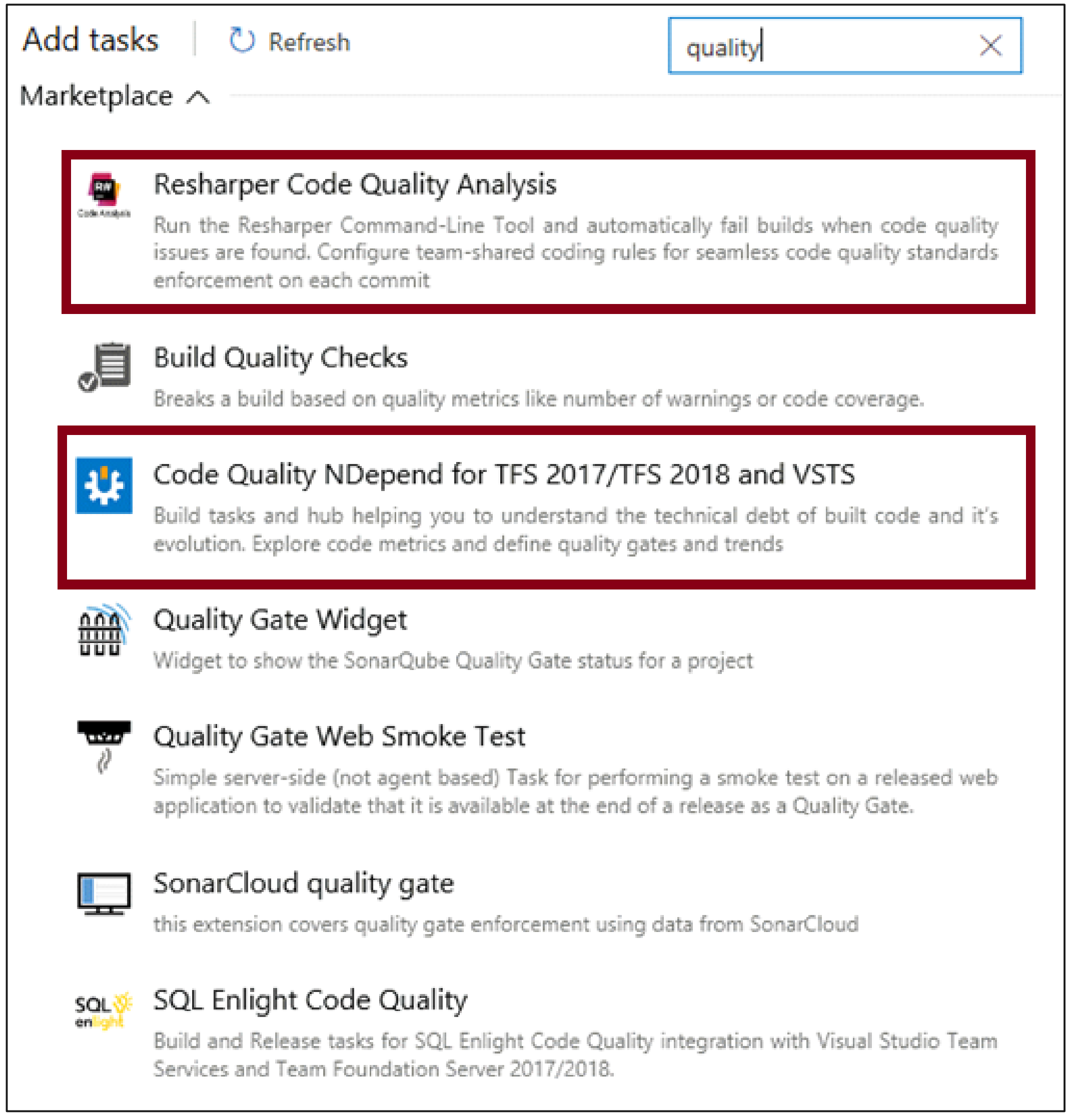
Refactor your code not to use hardcoded absolute paths or URIs. 3 years ago L14
Code Smell Minor Open Not assigned 20min effort Comment cert

Refactor your code not to use hardcoded absolute paths or URIs. 3 years ago L18
Code Smell Minor Open Not assigned 20min effort Comment cert

Refactor your code not to use hardcoded absolute paths or URIs. 3 years ago L19
Code Smell Minor Open Not assigned 20min effort Comment cert

# Integrating Other Code Quality Tools

- NDepend is a Visual Studio extension that assesses the amount of technical debt that a developer has added during a recent development period, typically in the last hour
- Resharper Code Quality Analysis is a command line tool and can be set to automatically fail builds when code quality issues are found



Resharper Code Quality Analysis
Run the Resharper Command-Line Tool and automatically fail builds when code quality issues are found. Configure team-shared coding rules for seamless code quality standards enforcement on each commit

Build Quality Checks
Breaks a build based on quality metrics like number of warnings or code coverage.

Code Quality NDepend for TFS 2017/TFS 2018 and VSTS
Build tasks and hub helping you to understand the technical debt of built code and it's evolution. Explore code metrics and define quality gates and trends

Quality Gate Widget
Widget to show the SonarQube Quality Gate status for a project

Quality Gate Web Smoke Test
Simple server-side (not agent based) Task for performing a smoke test on a released web application to validate that it is available at the end of a release as a Quality Gate.

SonarCloud quality gate
this extension covers quality gate enforcement using data from SonarCloud

SQL Enlight Code Quality
Build and Release tasks for SQL Enlight Code Quality integration with Visual Studio Team Services and Team Foundation Server 2017/2018.

## Discussion: Code Quality Tooling

Azure DevOps can be integrated with a wide range of existing tooling that is used for checking code quality during builds.

- Which code quality tools do you currently use (if any)?
- What do you like or don't like about the tools?

## Lab: Managing Technical Debt with Azure DevOps and SonarCloud

In this hands-on lab, you will learn how to manage and report on technical debt using SonarCloud integration with Azure DevOps. You will perform the following tasks:

- Integrate SonarCloud with Azure DevOps and run an analysis
- Analyze the results
- Configure a quality profile to control the rule set used for analyzing your project

✓ Note that you must have already completed the prerequisite labs in the Welcome section.

## Lesson 02: Managing Security Policies

## Lesson 2 Overview

- Open Source Licensing Challenges
- Avoiding the OWASP Top Ten
- Detecting Open Source Issues with WhiteSource Bolt
- Integrating Other Security Policy Tooling
- Security Policy Tooling
- Checking Vulnerabilities using WhiteSource Bolt and Azure DevOps

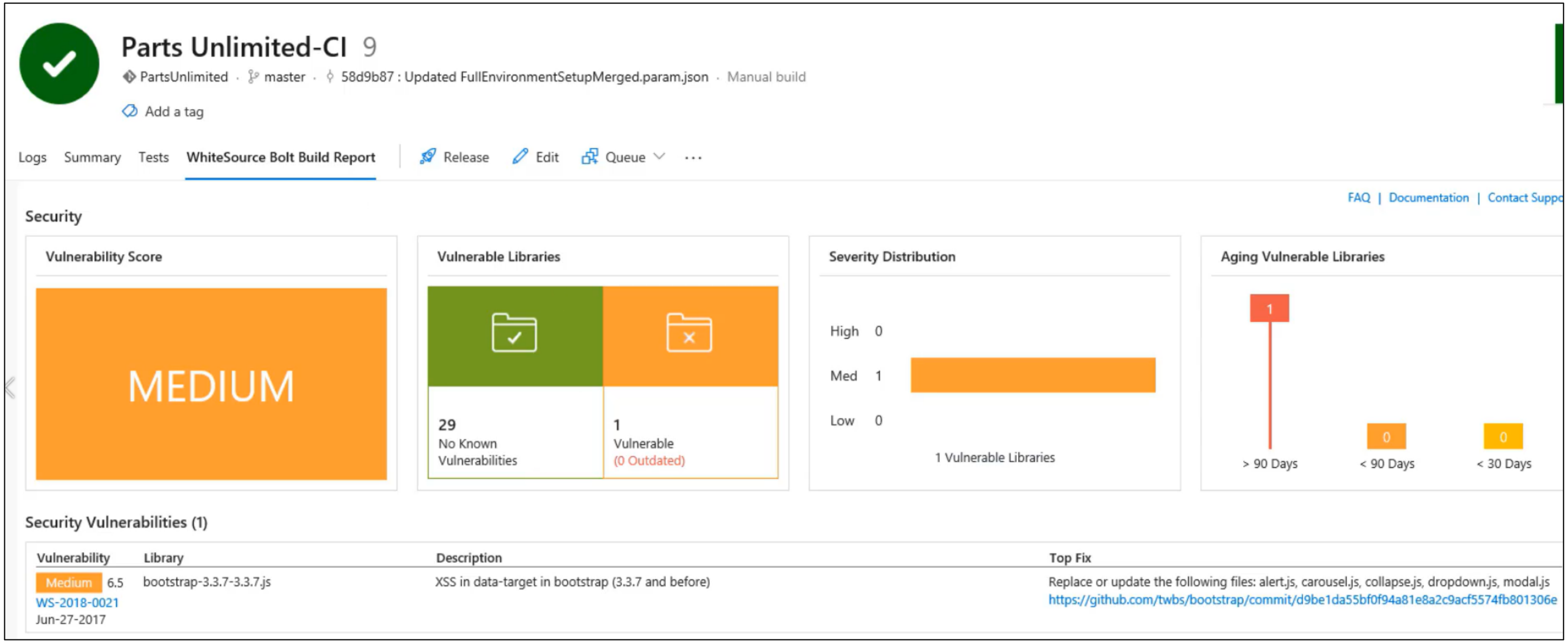
## Video: Open Source Licensing Challenges

- Open source software is code that everyone can read, modify, enhance, and share
- Incorporating open source code is convenient but can cause issues:
    - Security
    - Quality
    - Old versions
    - Licensing
- Minimize risk by implementing automated systems to manage the code

## Video: Avoiding OWASP Top Ten

1. Injection Attacks
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities
5. Broken Access Control

# Demonstration: Detecting Open Source Issues with WhiteSource Bolt



Parts Unlimited-CI 9

PartsUnlimited · master · 58d9b87 : Updated FullEnvironmentSetupMerged.param.json · Manual build

Add a tag

Logs Summary Tests **WhiteSource Bolt Build Report** | Release Edit Queue ···

FAQ | Documentation | Contact Suppo

Security

Vulnerability Score: MEDIUM

Vulnerable Libraries: 29 No Known Vulnerabilities; 1 Vulnerable (0 Outdated)

Severity Distribution

| | |
|---|---|
| High | 0 |
| Med | 1 |
| Low | 0 |

1 Vulnerable Libraries

Aging Vulnerable Libraries

| > 90 Days | < 90 Days | < 30 Days |
|---|---|---|
| 1 | 0 | 0 |

Security Vulnerabilities (1)

| Vulnerability | Library | Description | Top Fix |
|---|---|---|---|
| Medium 6.5 WS-2018-0021 Jun-27-2017 | bootstrap-3.3.7-3.3.7.js | XSS in data-target in bootstrap (3.3.7 and before) | Replace or update the following files: alert.js, carousel.js, collapse.js, dropdown.js, modal.js https://github.com/twbs/bootstrap/commit/d9be1da55bf0f94a81e8a2c9acf5574fb801306e |

## Integrating Other Security Policy Tooling

- Micro Focus Fortify searches for violations of security-specific coding rules and guidelines
- Checkmarx CxSAST is designed for identifying, tracking and fixing technical and logical security flaws
- BinSkim is a static analysis tool that scans binary files
- OWASP Zed Attack Proxy Scan is an open-source web application for professional penetration testers
- Kasun Kodagoda can run an active scan against a target with security risk thresholds and can generate scan reports

## Discussion: Security Policy Tooling

Azure DevOps can be integrated with a wide range of existing tooling that is used for checking security policy during builds.

- Which security policy tools do you currently use?
- What do you like or don't like about the tools?

## Lab: Checking Vulnerabilities using WhiteSource Bolt with Visual Studio Team Services

In this hands-on lab, you will learn how to check for open source vulnerabilities using WhiteSource Bolt in conjunction with Azure DevOps. You will learn how to:

- Integrate WhiteSource Bolt with you Azure DevOps build process
- Detect and remedy vulnerable open source components
- Generate comprehensive open source inventory reports per project or build
- Enforce open source license compliance, including licenses for dependencies
- Identify outdated open source libraries with recommendations to update

✓ Note that you must have already completed the prerequisite labs in the Welcome section.

## Module 2: Review Questions

1. You want to run a penetration test against your application. Which tool could you use?
2. What is code smells? Give an example of a code smell.
3. You are using Azure Repos for your application source code repository. You want to create an audit of open source libraries that you have used. Which tool could you use?
4. Name three attributes of high-quality code.
5. You are using Azure Repos for your application source code repository. You want to perform code quality checks. Which tool could you use?