UTD

THE UNIVERSITY OF TEXAS AT DALLAS

Coding Metrics for United Airlines

Team 4

Pallavi Vayalali

Prajakta Bhavsar

Mujahed Ali Khan

William Wade Mullican

AGENDA

- Executive Summary
- Problem Statement
- Key Objectives
- Content and Explanation with a Demo
- Improvement Recommendations
- Closing Summary
- Exhibit

EXECUTIVE SUMMARY

United Airlines currently lacks the means to accurately measure and analyze developer productivity, code quality, and the success of software deployments.

Goal

- Design and implement a data-driven, AI-supported framework
- Interactive dashboard integrated into Azure DevOps
- Enabling real-time visibility

Solution Intention

- Optimize sprint planning and resource management
- Track key performance indicators
- Integrate AI tools to automate testing
- Help stakeholders make data-driven decisions

PROBLEM STATEMENT ADDRESSING CURRENT CHALLENGES AND ROADBLOCKS

- **Lack of Structured Metrics**
  There is no standardized framework to measure individual or team productivity, making it difficult to evaluate performance.

- **Poor Visibility into Code Quality**
  No consistent metrics are in place to assess the quality of third-party or in-house developed code.

- **Resource Planning Challenges**
  The current system does not allow managers to accurately predict the resources needed to move a user story from backlog to deployment.

- **Untracked Failures**
  There is no mechanism to track the failure rate of user stories or production releases, leading to inefficiencies and increased rework.

OBJECTIVES TO ENHANCE RELIABILITY AND IMPROVE METRICS

- **Track Developer Metrics Visually**
  Build a dashboard that shows key metrics like how fast code is written, how often bugs happen, and how long tasks take.

- **Make Sprint Planning Smarter**
  Keep an eye on team workload to avoid overloading developers and plan sprints more effectively.

- **Boost Deployment Reliability**
  Keep track of how often deployments fail and quickly spot any rollbacks.

- **Use AI for Smarter Testing**
  Use AI tools like GitHub Copilot/ SonarQube to automatically suggest code and create test cases, saving time and effort.

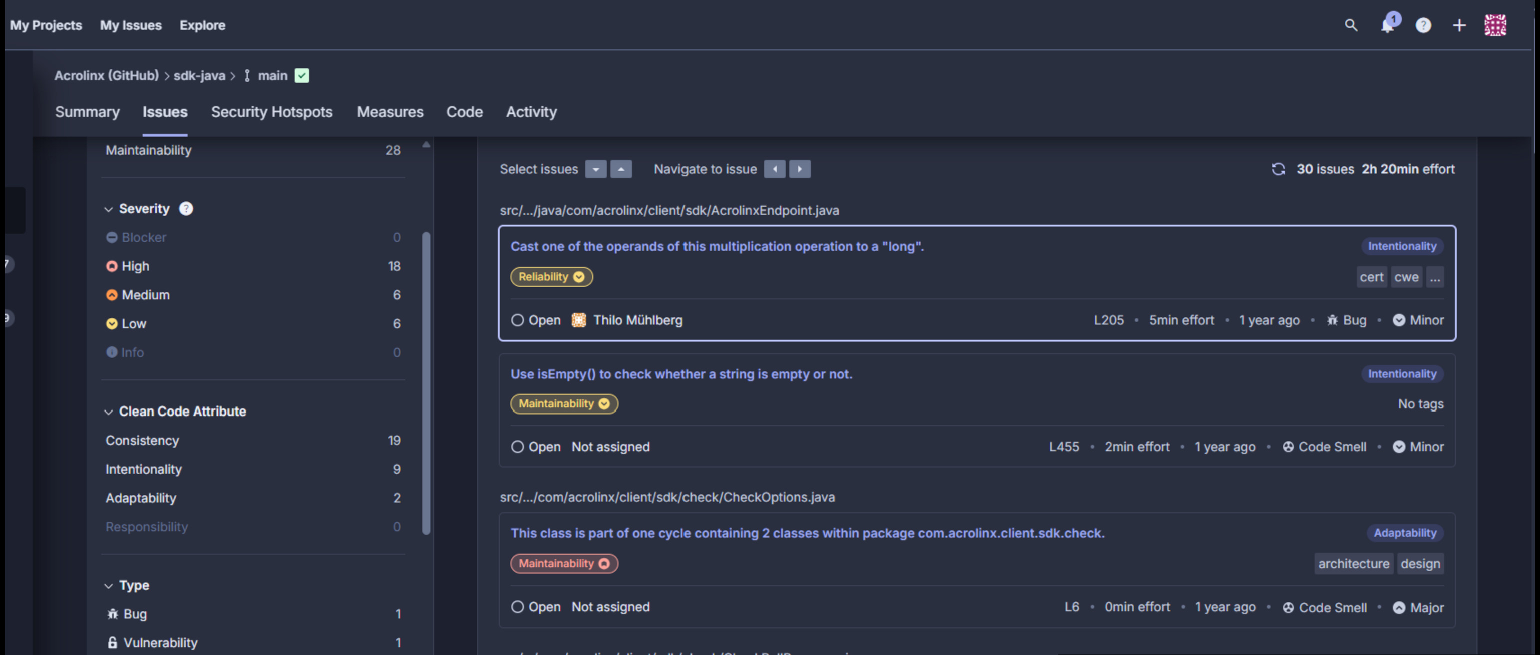
AI TOOLS THAT FIT BEST FOR UNITED AIRLINES

Tool	Purpose	Why It's a fit
Azure OpenAI	Custom AI apps, sprint summaries, NLP features.	Seamless with Azure ecosystem, flexible for DevOps use cases.
CodeAnt AI	Automated code reviews, bug detection.	Easy Azure DevOps integration, speeds up feedback cycle.
SonarQube	*Continuous code quality & vulnerability detection.*	*Already integrated, enforces coding standards and quality.*

CONTENT / ANALYSIS / EXPLANATION

- Live Demo

Azure DevOps Backlog Link:
https://dev.azure.com/wwm170000/UnitedWorkFlow/_dashboards/dashboard/91e3de14-3d21-48fc-8de3-2bf4cc205d2c



Acrolinx (GitHub) > sdk-java > main

Maintainability	28

Severity

Blocker	0
High	18
Medium	6
Low	6
Info	0

Clean Code Attribute

Consistency	19
Intentionality	9
Adaptability	2
Responsibility	0

Type

Bug	1
Vulnerability	1
Code Smell	28

Status

Security Category

Select issues Navigate to issue 30 issues 2h 20min effort

src/.../java/com/acrolinx/client/sdk/AcrolinxEndpoint.java

Cast one of the operands of this multiplication operation to a "long".
Intentionality
Reliability
cert cwe ...
Open Thilo Mühlberg L205 • 5min effort • 1 year ago • Bug • Minor

Use isEmpty() to check whether a string is empty or not.
Intentionality
Maintainability
No tags
Open Not assigned L455 • 2min effort • 1 year ago • Code Smell • Minor

src/.../com/acrolinx/client/sdk/check/CheckOptions.java

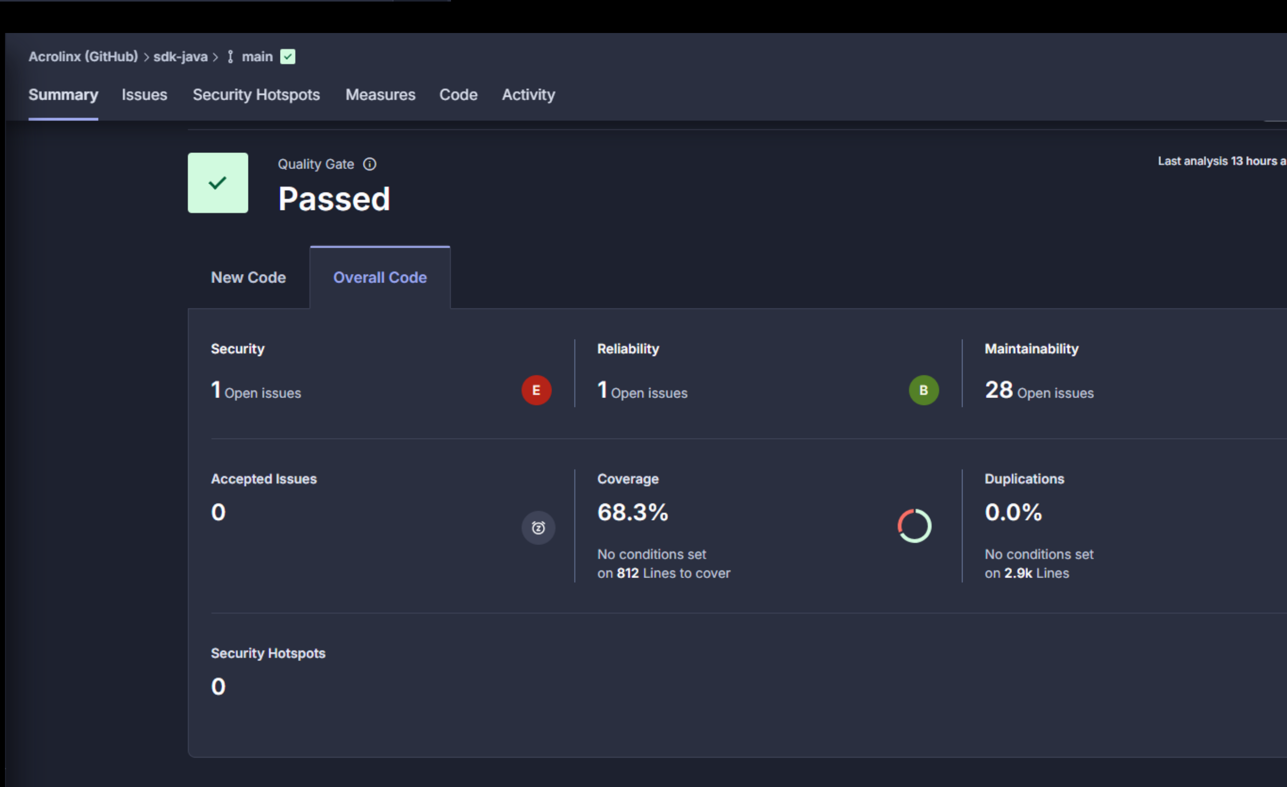
This class is part of one cycle containing 2 classes within package com.acrolinx.client.sdk.check.
Adaptability
Maintainability
architecture design
Open Not assigned L6 • 0min effort • 1 year ago • Code Smell • Major

src/.../com/acrolinx/client/sdk/check/CheckPollResponse.java

Rename field "progress"
Maintainability
Open acromarco L2

src/.../com/acrolinx/client/sdk/check/CheckRequest.java

UNITED AIRLINES



Acrolinx (GitHub) > sdk-java > main

Quality Gate
Passed

Last analysis 13 hours a

Security
1 Open issues
E

Reliability
1 Open issues
B

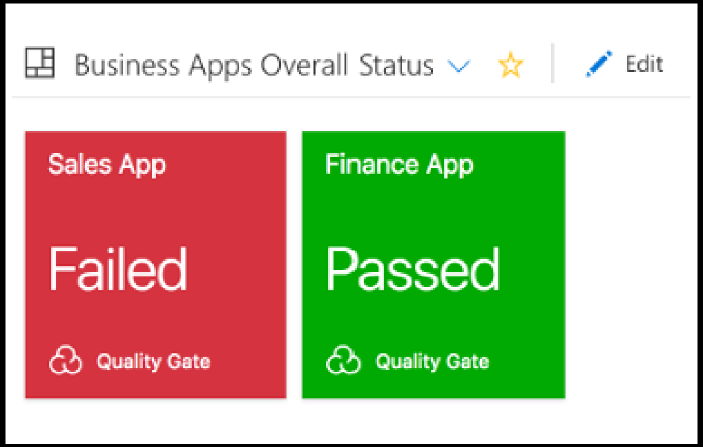
Maintainability
28 Open issues

Accepted Issues
0

Coverage
68.3%
No conditions set
on 812 Lines to cover

Duplications
0.0%
No conditions set
on 2.9k Lines

Security Hotspots
0

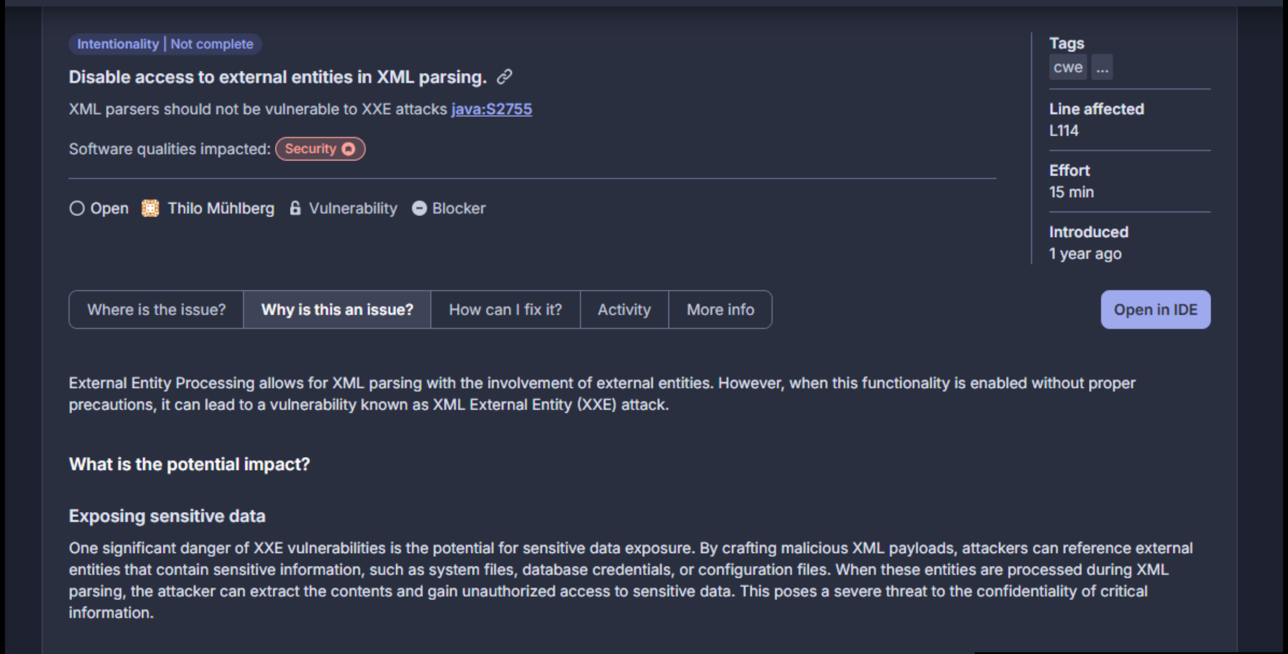


Business Apps Overall Status

Sales App
Failed
Quality Gate

Finance App
Passed
Quality Gate

SonarQube Summary



Disable access to external entities in XML parsing.

XML parsers should not be vulnerable to XXE attacks java:S2755

Software qualities impacted: Security

Tags	cwe ...
Line affected	L114
Effort	15 min
Introduced	1 year ago

External Entity Processing allows for XML parsing with the involvement of external entities. However, when this functionality is enabled without proper precautions, it can lead to a vulnerability known as XML External Entity (XXE) attack.

What is the potential impact?

Exposing sensitive data

One significant danger of XXE vulnerabilities is the potential for sensitive data exposure. By crafting malicious XML payloads, attackers can reference external entities that contain sensitive information, such as system files, database credentials, or configuration files. When these entities are processed during XML parsing, the attacker can extract the contents and gain unauthorized access to sensitive data. This poses a severe threat to the confidentiality of critical information.

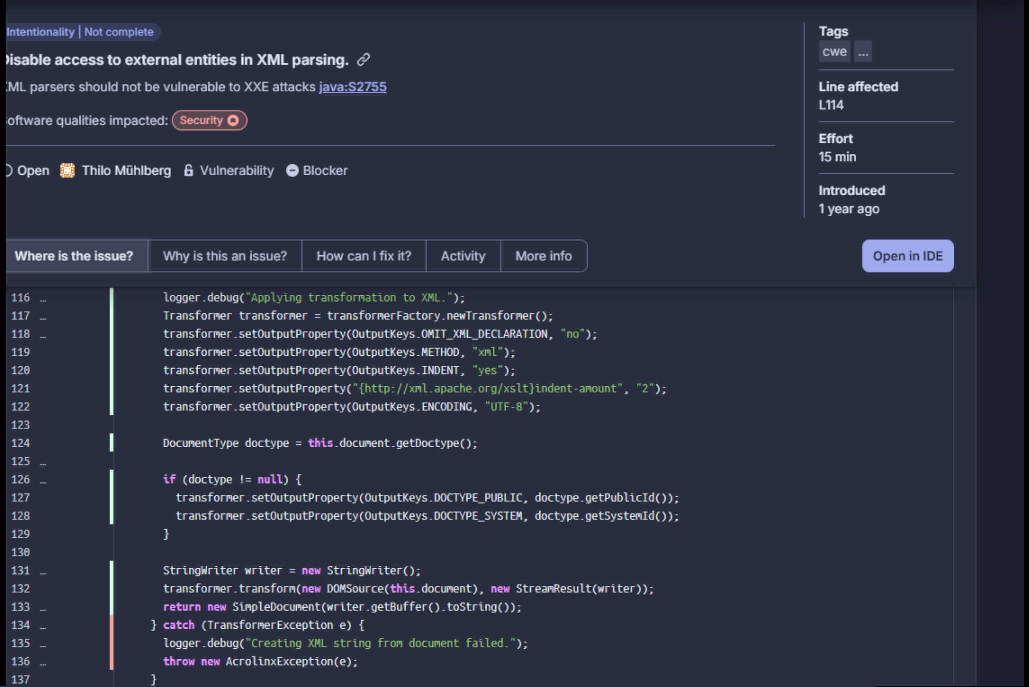
Exhausting system resources

Another consequence of XXE vulnerabilities is the potential for denial-of-service attacks. By exploiting the ability to include exte
construct XML payloads that cause resource exhaustion. This can overwhelm the system's memory, CPU, or other critical resou
unresponsiveness or crashes. A successful DoS attack can disrupt the availability of services and negatively impact the user ex

Forging requests

XXE vulnerabilities can also enable Server-Side Request Forgery (SSRF) attacks. By leveraging the ability to include external en

Analysis of the Issues faced



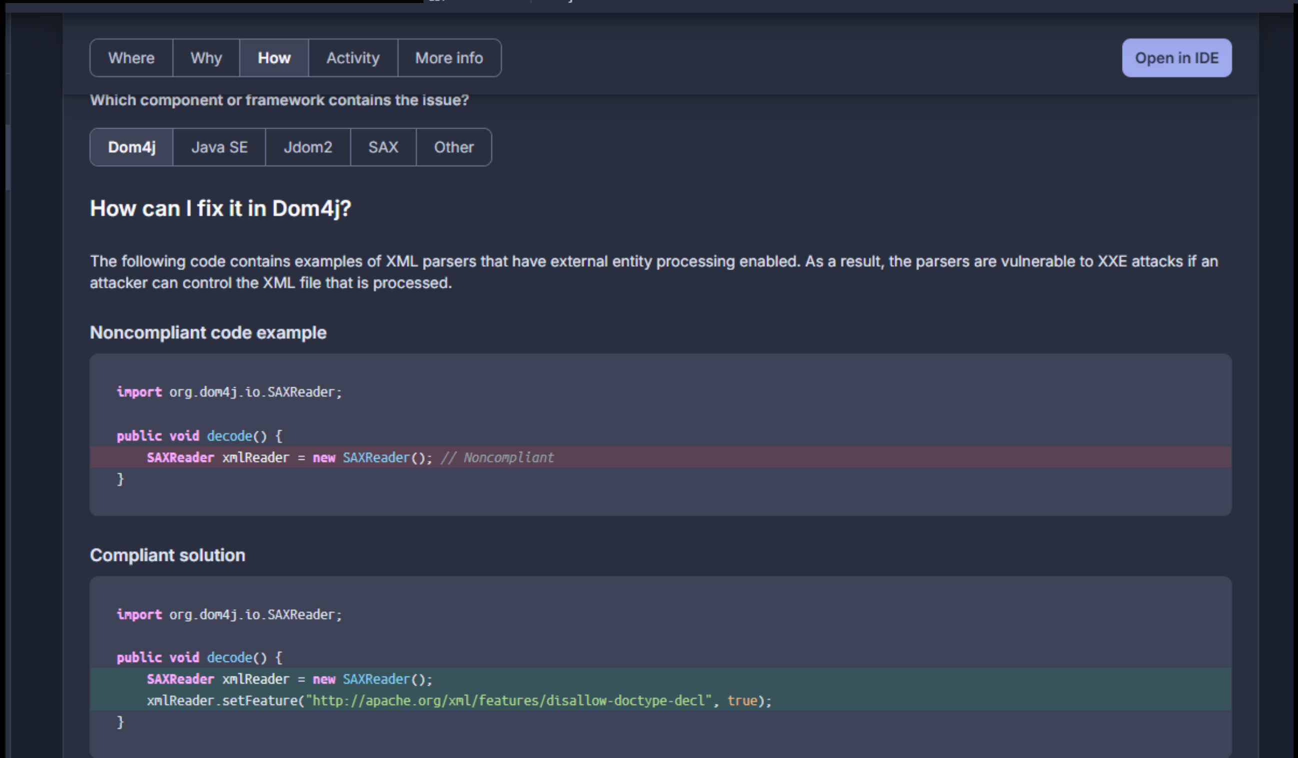
isable access to external entities in XML parsing.

ML parsers should not be vulnerable to XXE attacks java:S2755

oftware qualities impacted: Security

Tags	cwe ...
Line affected	L114
Effort	15 min
Introduced	1 year ago

```
        logger.debug("Applying transformation to XML.");
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.setOutputProperty(OutputKeys.METHOD, "xml");
        transformer.setOutputProperty(OutputKeys.INDENT, "yes");
        transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");
        transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8");

        DocumentType doctype = this.document.getDoctype();

        if (doctype != null) {
          transformer.setOutputProperty(OutputKeys.DOCTYPE_PUBLIC, doctype.getPublicId());
          transformer.setOutputProperty(OutputKeys.DOCTYPE_SYSTEM, doctype.getSystemId());
        }

        StringWriter writer = new StringWriter();
        transformer.transform(new DOMSource(this.document), new StreamResult(writer));
        return new SimpleDocument(writer.getBuffer().toString());
      } catch (TransformerException e) {
        logger.debug("Creating XML string from document failed.");
        throw new AcrolinxException(e);
      }
```



Which component or framework contains the issue?

How can I fix it in Dom4j?

The following code contains examples of XML parsers that have external entity processing enabled. As a result, the parsers are vulnerable to XXE attacks if an attacker can control the XML file that is processed.

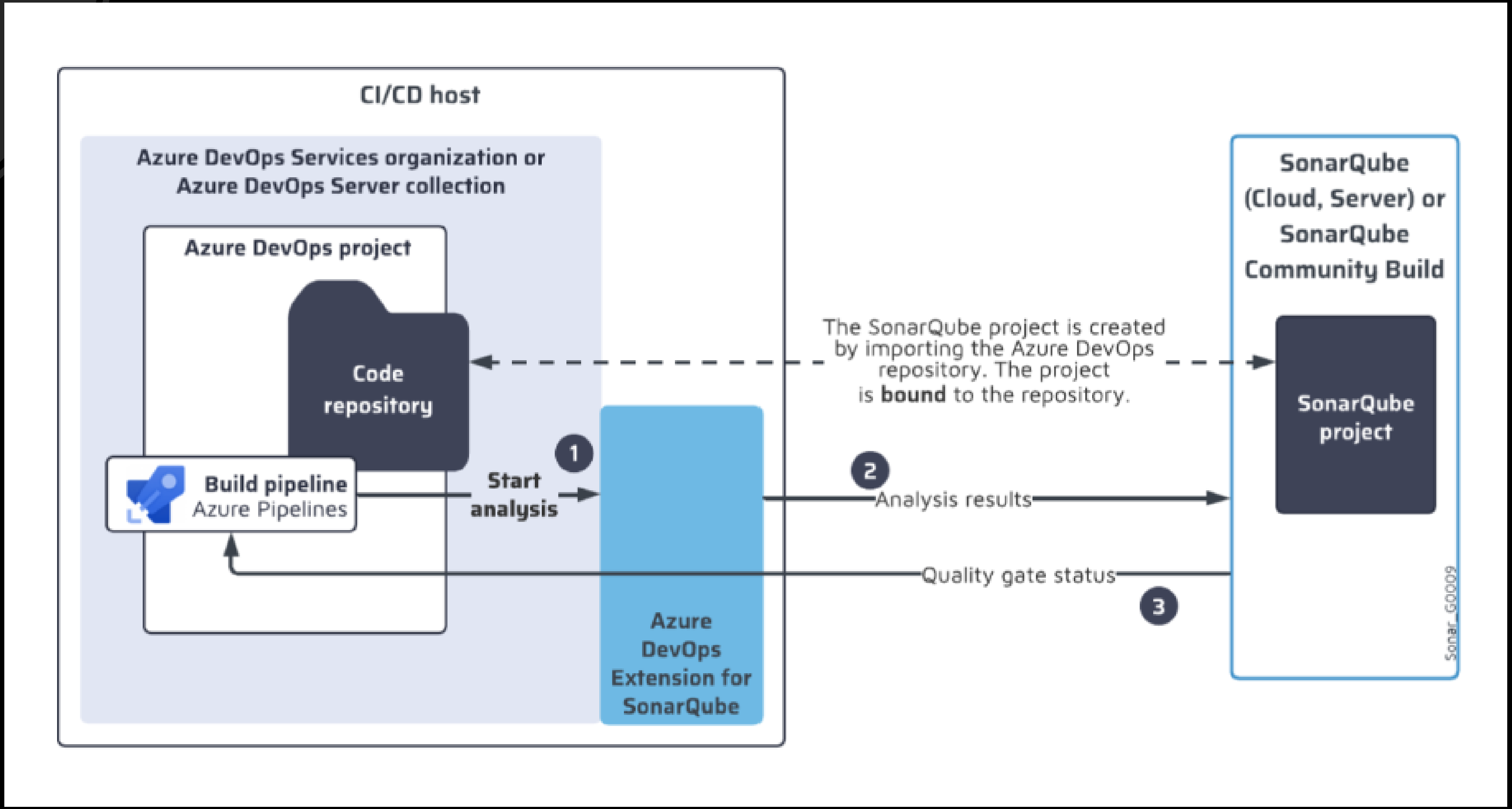
Noncompliant code example

```
import org.dom4j.io.SAXReader;

public void decode() {
    SAXReader xmlReader = new SAXReader(); // Noncompliant
}
```

Compliant solution

```
import org.dom4j.io.SAXReader;

public void decode() {
    SAXReader xmlReader = new SAXReader();
    xmlReader.setFeature("http://apache.org/xml/features/disallow-doctype-decl", true);
}
```

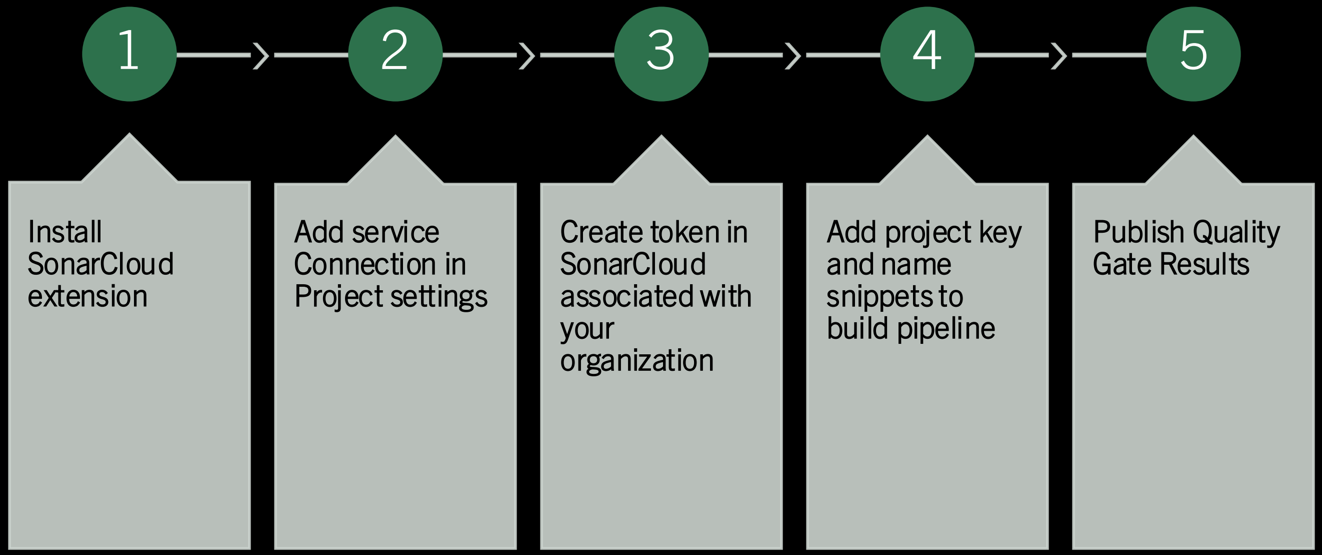
WHAT THE SONARQUBE ARCHITECTURE LOOKS LIKE



CI/CD host
Azure DevOps Services organization or Azure DevOps Server collection
Azure DevOps project
Code repository
Build pipeline
Azure Pipelines
1
Start analysis
Azure DevOps Extension for SonarQube
The SonarQube project is created by importing the Azure DevOps repository. The project is **bound** to the repository.
2
Analysis results
Quality gate status
3
SonarQube (Cloud, Server) or SonarQube Community Build
SonarQube project
Sonar_G0009

INTEGRATION WITH MS AZURE PIPELINE

ADO PIPELINE



1. Install SonarCloud extension
2. Add service Connection in Project settings
3. Create token in SonarCloud associated with your organization
4. Add project key and name snippets to build pipeline
5. Publish Quality Gate Results

MANUALLY TRIGGER

1. Create token in SonarCloud associated with your organization
2. Set SONAR_TOKEN environmental variable
3. Add project and organization snippets to build files
4. Invoke with by manually calling sonar

Example ./gradle sonar

IMPROVEMENT RECOMMENDATIONS

- Break down user stories into developer tasks with estimated hours.
- **Use start and target date** fields on stories.
- **Tags each User Story** for rollovers to keep organization in the backlog.
- **Establish new states** – multiple states account for completion or in-progress.
- **Daily syncs** with the team.
- Run weekly backlog grooming to keep everything up to date.

CLOSING SUMMARY

Key Takeaways

- The current system at United Airlines lacks structured visibility into developer productivity, code quality, and sprint planning.
- Our solution provides a real-time dashboard, measurable KPIs, and AI integrations tailored for Azure DevOps.

Benefits of our proposed solution

- Improved resource allocation.
- Automated quality checks.
- Data-driven sprint decisions.
- Enhanced transparency across teams.

- EXHIBIT-

EXHIBIT 1: RECOMMENDED AI TOOLS FOR UNITED AIRLINES

Tool	Key Features	Suitability for United Airlines
Azure OpenAI	Custom AI solutions, code generation.	Highly suitable; enables tailored AI-driven solutions.
CodeAnt AI	Automated code reviews, bug detection.	Highly Suitable; aligns with code quality assessment needs.
SonarQube	Static code analysis, quality gates.	Suitable; offers comprehensive code quality analysis.
GitHub Copilot	Autocomplete code suggestions.	Suitable; enhances developer productivity.
Amazon Q Developer	Code suggestions, code transformation.	Less suitable due to ASW-centric design.

THANK YOU!

We hope you liked our recommendations!

Any questions?