
Université d'Ottawa | University of Ottawa
Lab 8
Mutation Testing
SEG3103 Summer 2024 Lab 8
uOttawa.ca
uOttawa

What is Mutation Testing?

- The purpose of mutation testing is to make sure that the tests we write for our code are effective and can catch any mistakes or errors. It's like a test for our tests!

- Mutation testing is like checking for the effectiveness of our immune system against different strains of viruses. We deliberately create altered versions of a virus and see if our immune system can detect and fight them. It helps us strengthen our defenses and ensure our immunity can handle various viral mutations.

Introducing Java PITest

- Java Pitest: A popular mutation testing tool for Java
- Open-source and actively maintained
- Compatible with various testing frameworks (JUnit, TestNG, etc.)
- Provides comprehensive mutation testing capabilities

How Java PITest Works?

- Mutates the application code to create a set of mutants
- Executes the existing test suite against both original and mutant code
- Compares the test results to identify killed mutants
- Calculates the mutation score: (killed mutants / total mutants)

Mutators

- Multiple Mutators: PIT provides a wide range of mutators, each responsible for specific code transformations. These mutators introduce changes such as modifying operators, changing constants, altering method invocations, and more. This diversity ensures thorough testing coverage.

Benefits of Java PITest

- Accurate and reliable assessment of test suite effectiveness
- Identifies weak areas in test coverage
- Helps prioritize test
- improvement efforts
- Enables developers to write more robust and reliable tests
- Enhances overall software quality

Key Features of Java PITest

- Mutation Operators: Offers a variety of mutation operators for code alteration (e.g., conditional replacements, variable assignments)
- Incremental Mutation Testing: Allows selective mutation testing based on modified code regions
- Comprehensive Reporting: Detailed reports highlighting killed mutants, mutation score, and test coverage

Implementation

Prerequisites

1. IntelliJ idea community edition: Download
2. Sample Code Files: Available on BrightSpace

Real World Examples

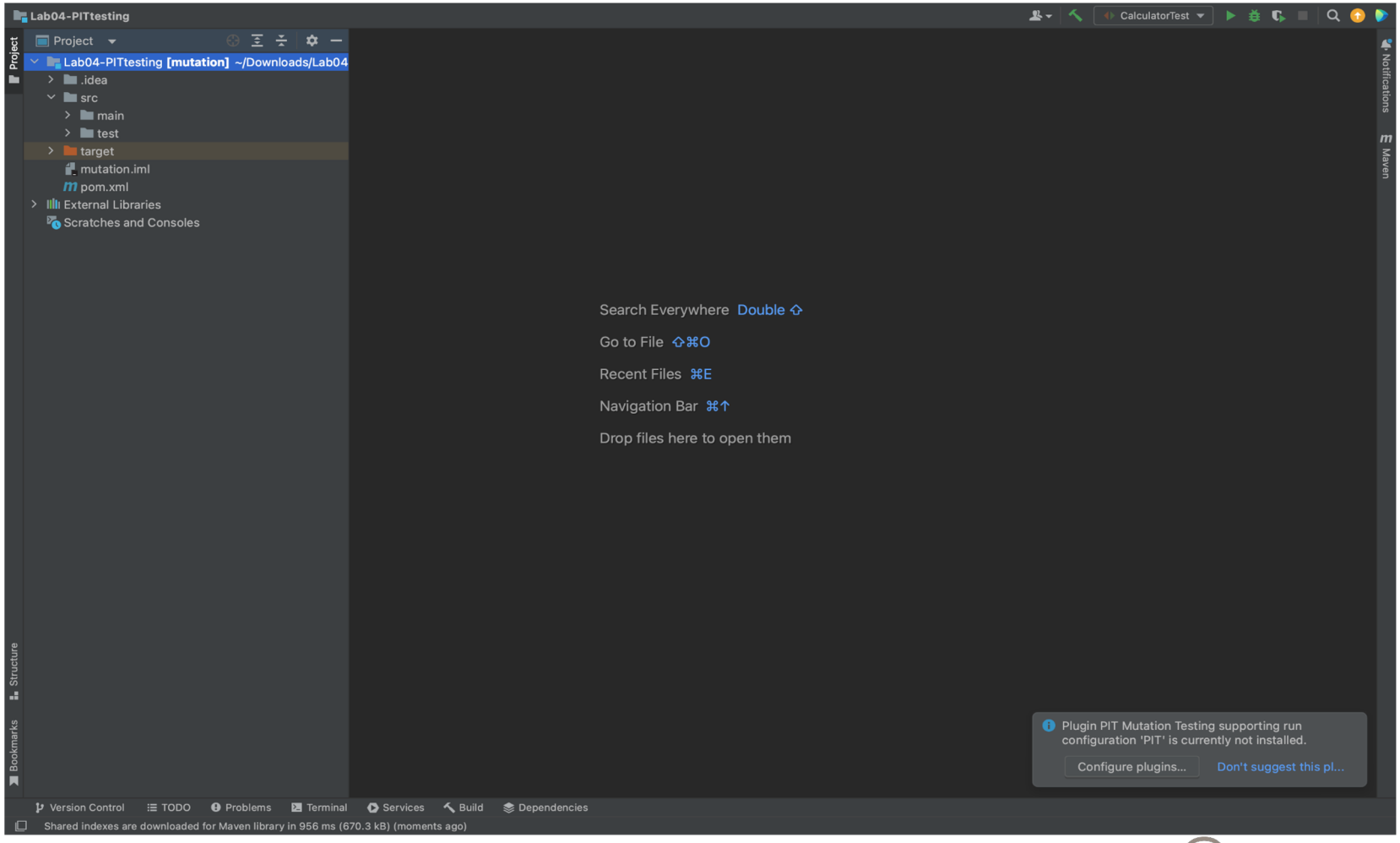
1. NASA's Mars Rover: When NASA developed the software for their Mars Rovers, they used mutation testing to ensure the highest level of reliability. By subjecting the Rover's software to mutated versions and carefully analyzing the test results, they were able to identify potential vulnerabilities and improve the code's robustness. This helped ensure that the Rover could withstand the challenging conditions of Mars and carry out its mission successfully.

2. Financial Institutions: Many financial institutions employ mutation testing to enhance the reliability and security of their software systems. By subjecting critical financial software to mutations, they can identify potential flaws that could lead to errors or security breaches. This enables them to refine their testing strategies, uncover hidden defects, and build more secure and trustworthy systems for processing transactions, managing accounts, and protecting sensitive data.

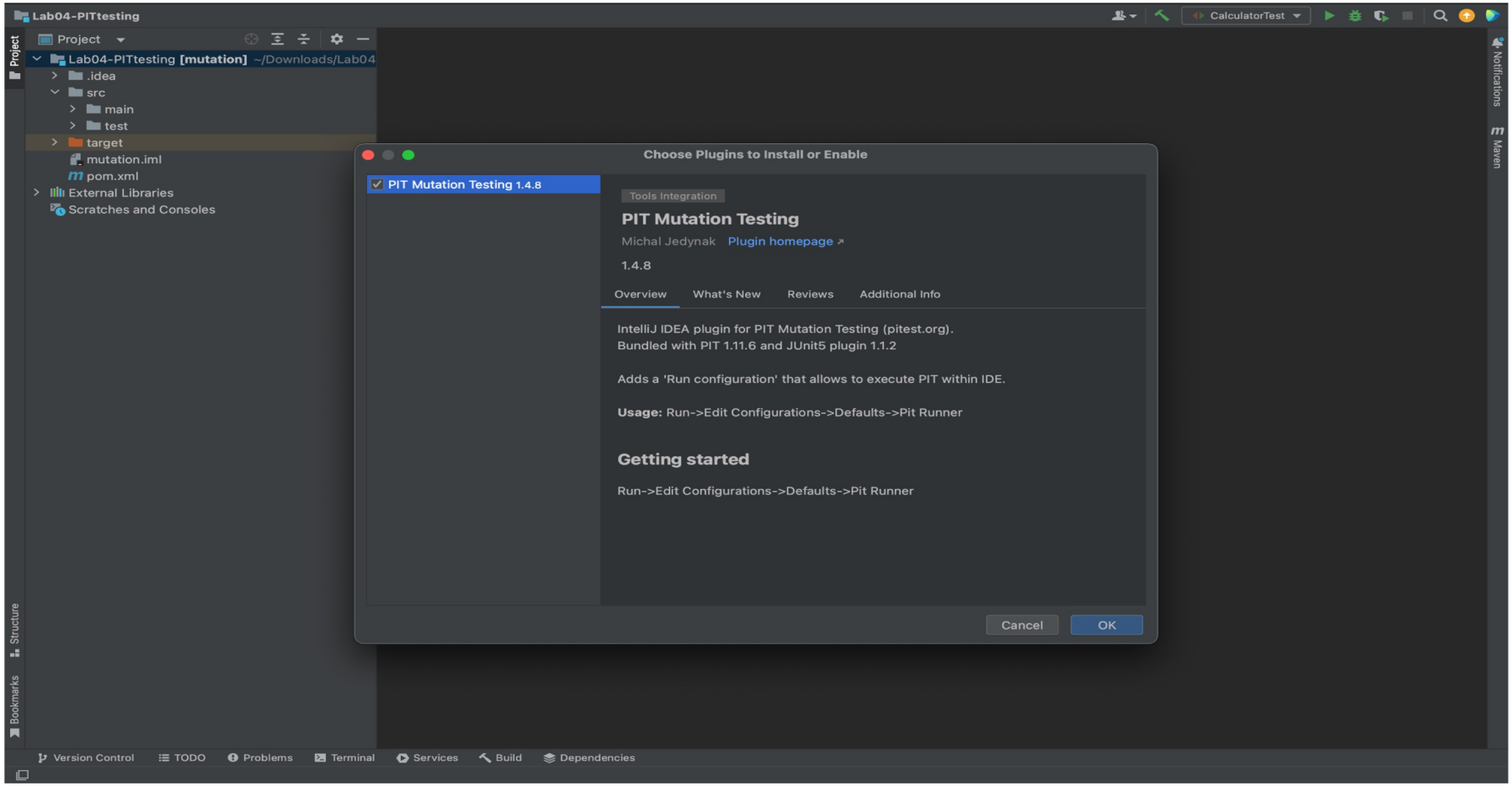
Implementation

1. Go to File - Open - open the sample code file folder.
2. Install the required PITest plugin from the popup.

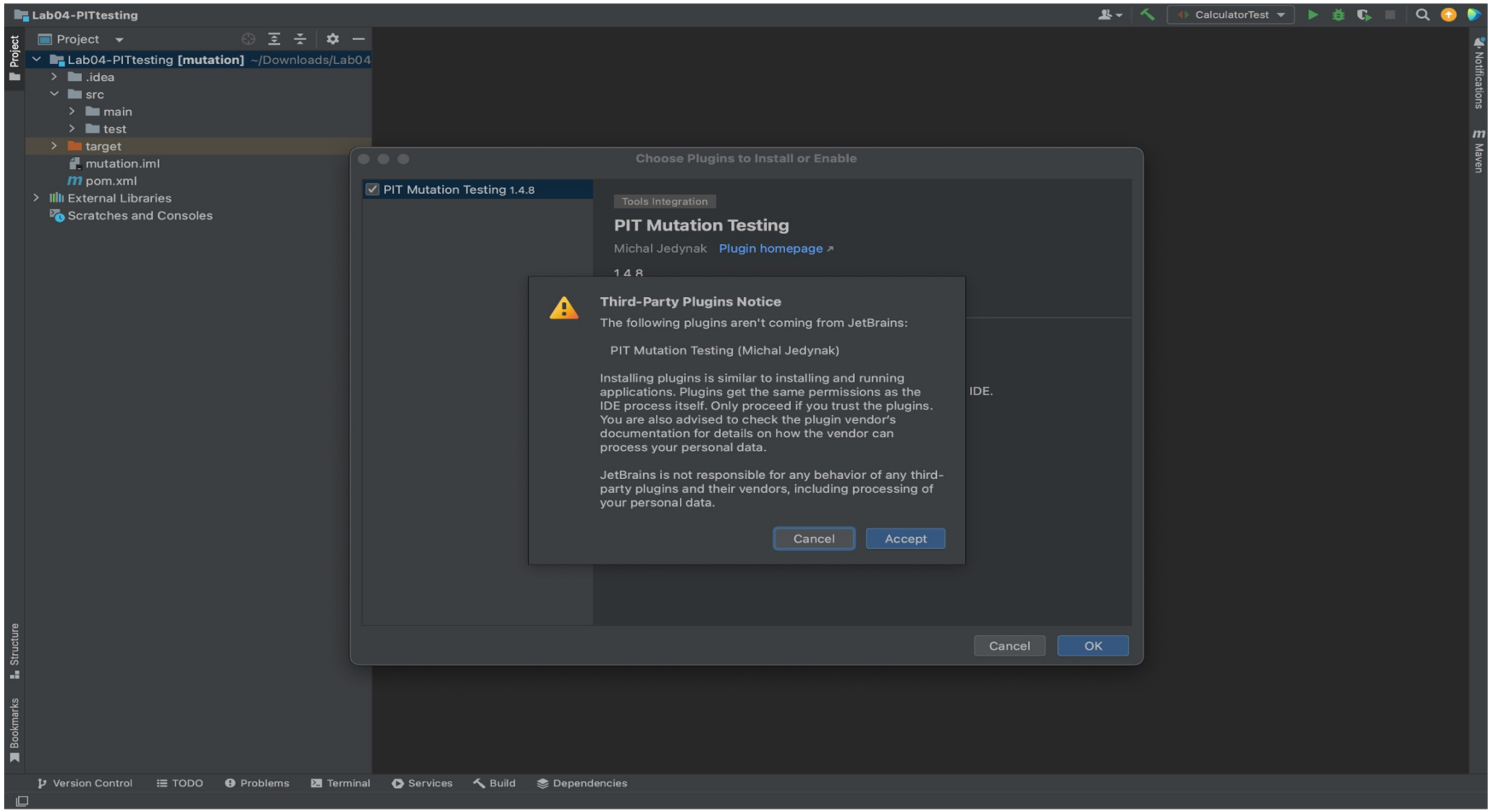
Install Plugin



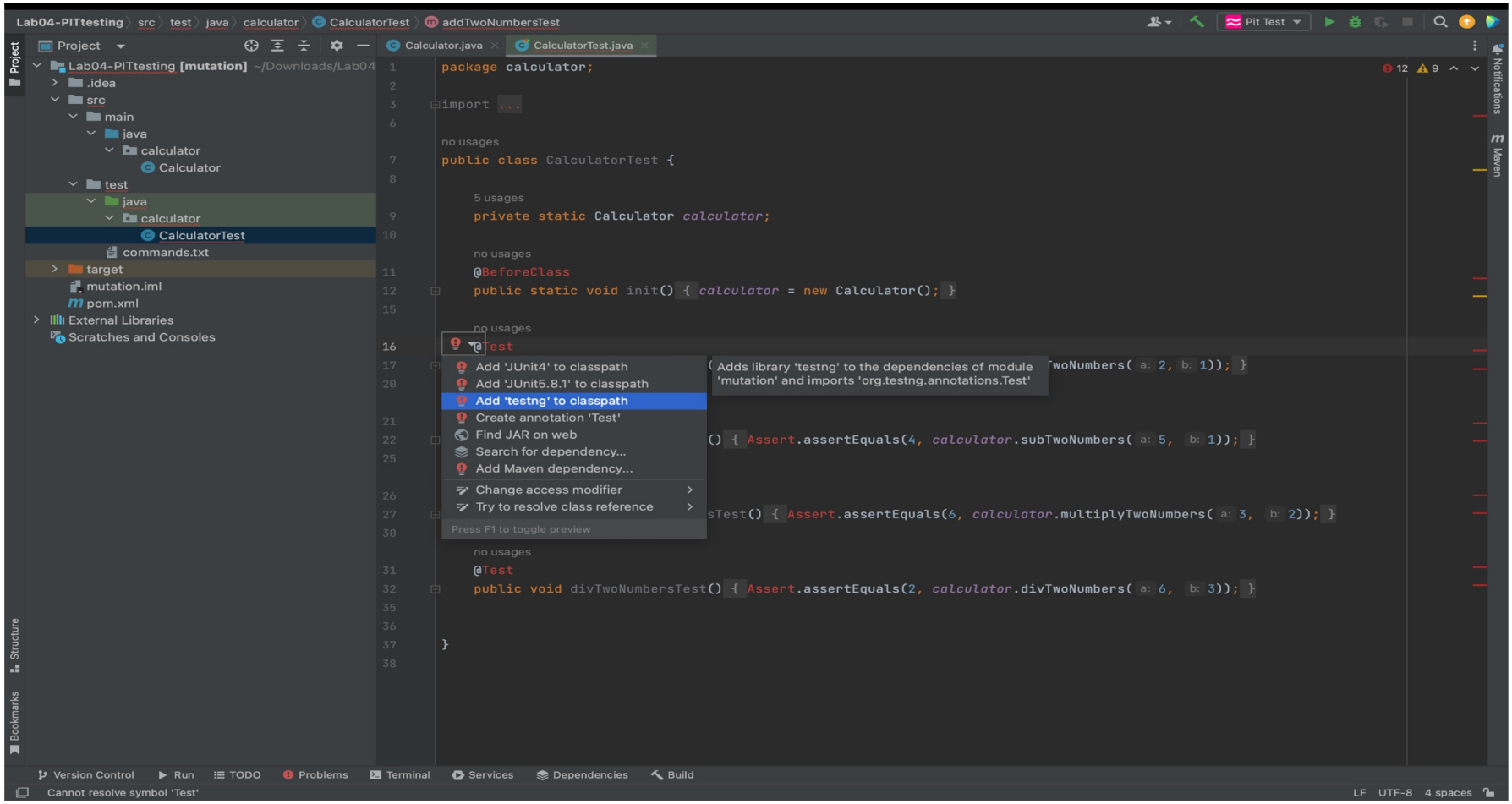
Install Plugin



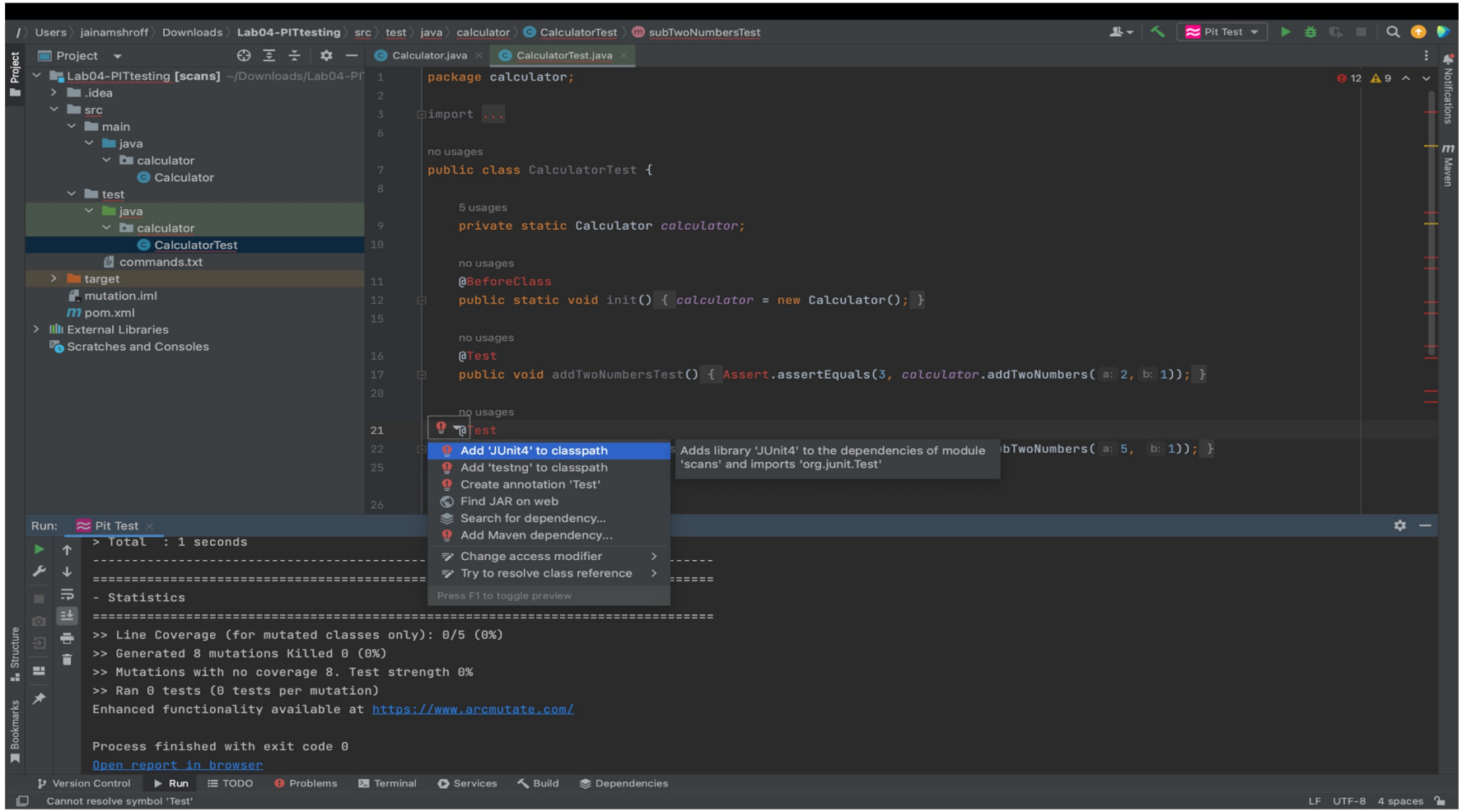
Install Plugin



Add dependencies to classpath



Add dependencies to classpath



Submission

1. Create a **lab08** folder in your **seg3103_playground** repository
2. Create a README.md
    1. Show at least one screenshot of starting the PIT Test plugin
    2. Show at least 2 screenshots of the HTML report
        1. Open the report from **./target/report/index.html** or click the link in the terminal
        2. In your own words explain what the various mutations are doing and what the report tells you
        3. Explain what it means for a mutant to be killed
    3. There is no code to write/upload for this lab
3. Share the repository with the professor and TAs