ALDARUNNER

Report

Submitted by Dave Hoevenaars

Fontys Hogeschool Techniek en Logistiek

Reuver, 22-06-2021

# Summary

This report will describe, the artefacts discussed during the duration of the project sequence, as well as lists some resources in which further knowledge on certain aspects can be gained

# 1. Task Description & Goal

My assignment is consisting of researching and realising a prototype of an automated assignment handling system, which will remove the need for labour intensive testing and also reduce the complexity and possibly the vulnerability level of the current system in place. And document my findings about this in such a way that it can be easily picked up by whoever continues within this project sequence.

# Solution & Results

In this section the progress that has been made will be discussed such as analysis artefacts and working prototypes.

## Analysis

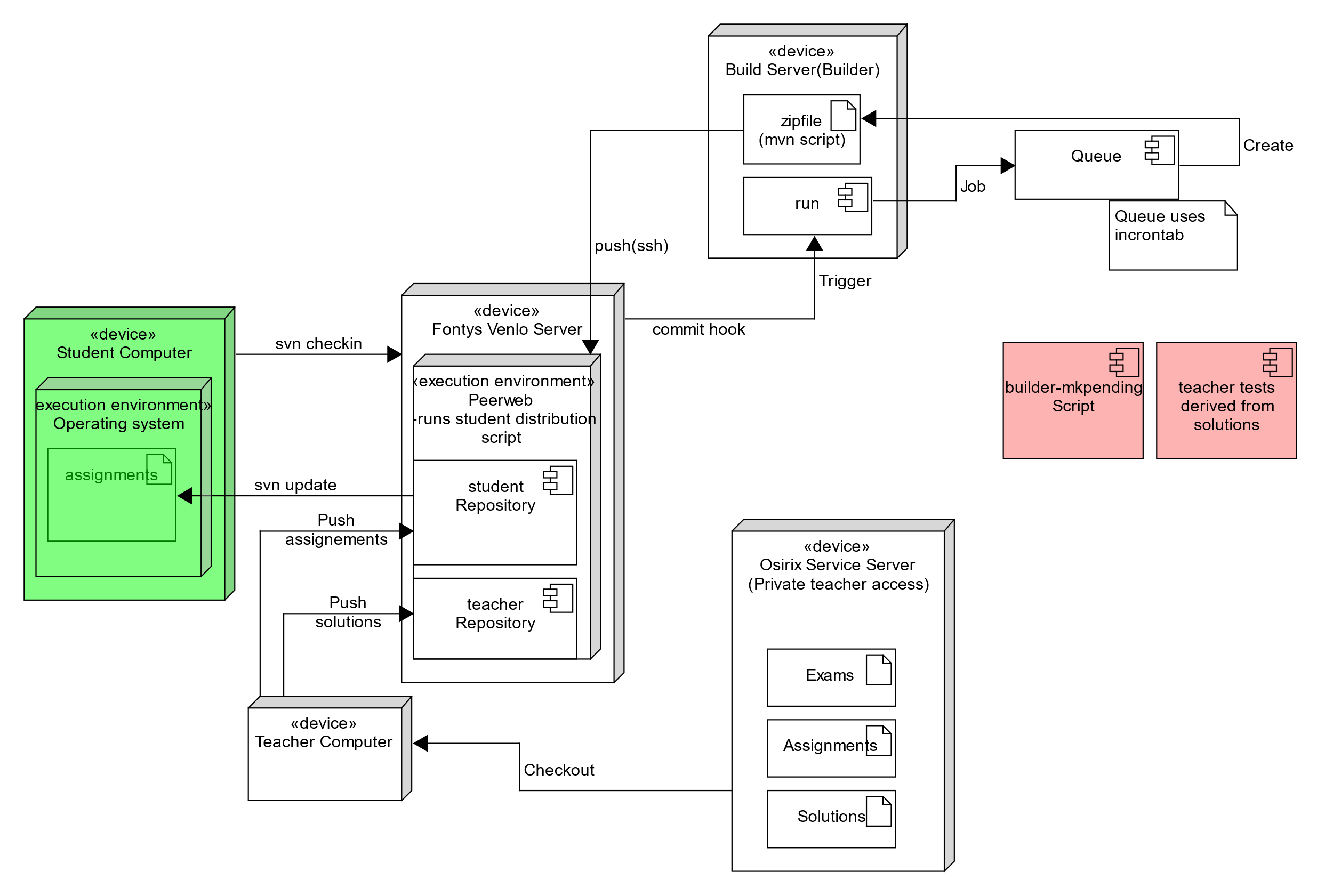


Figure 1 : Deployment Diagram(as is)

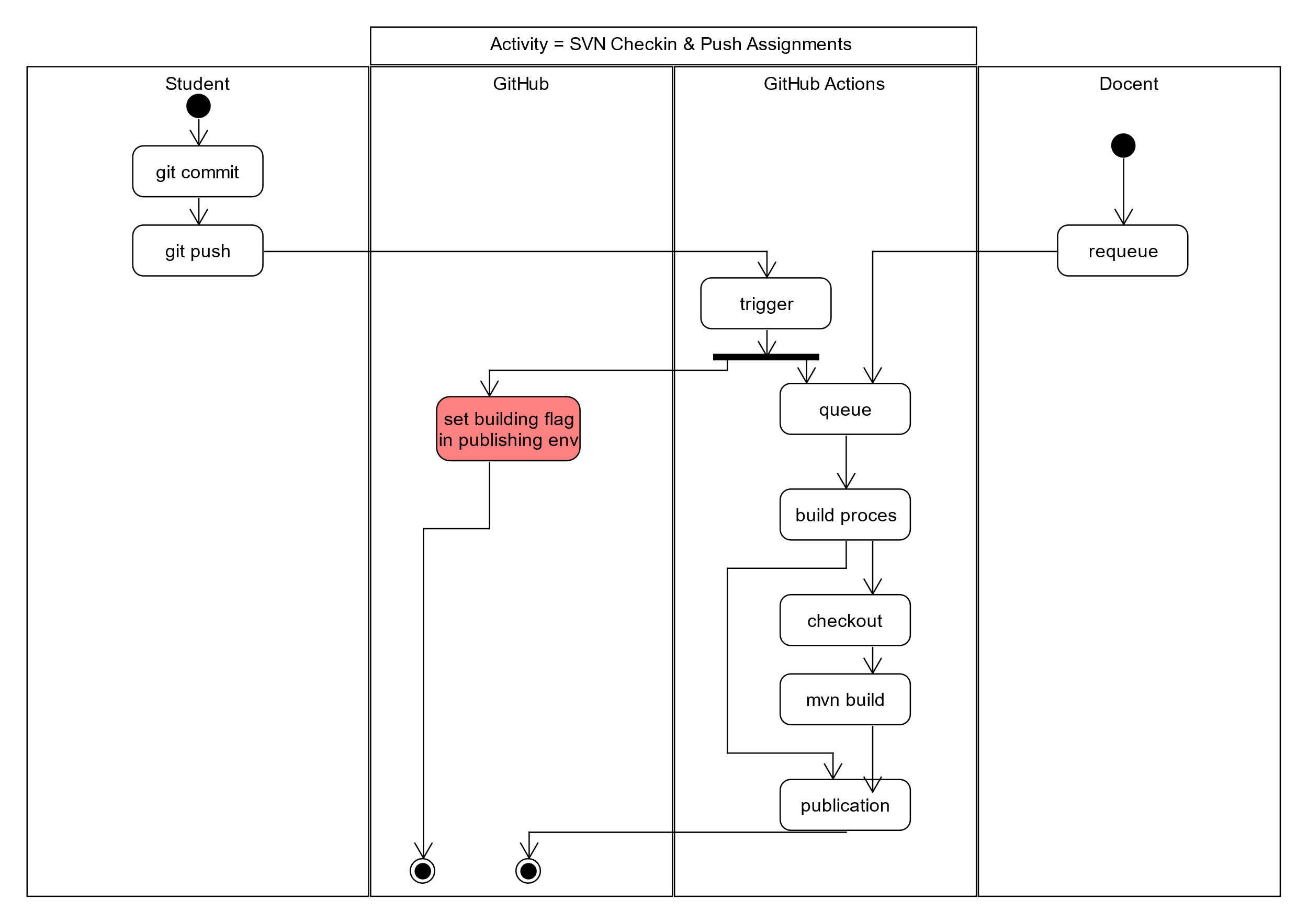


Figure 2 Activity Diagram: ToBe

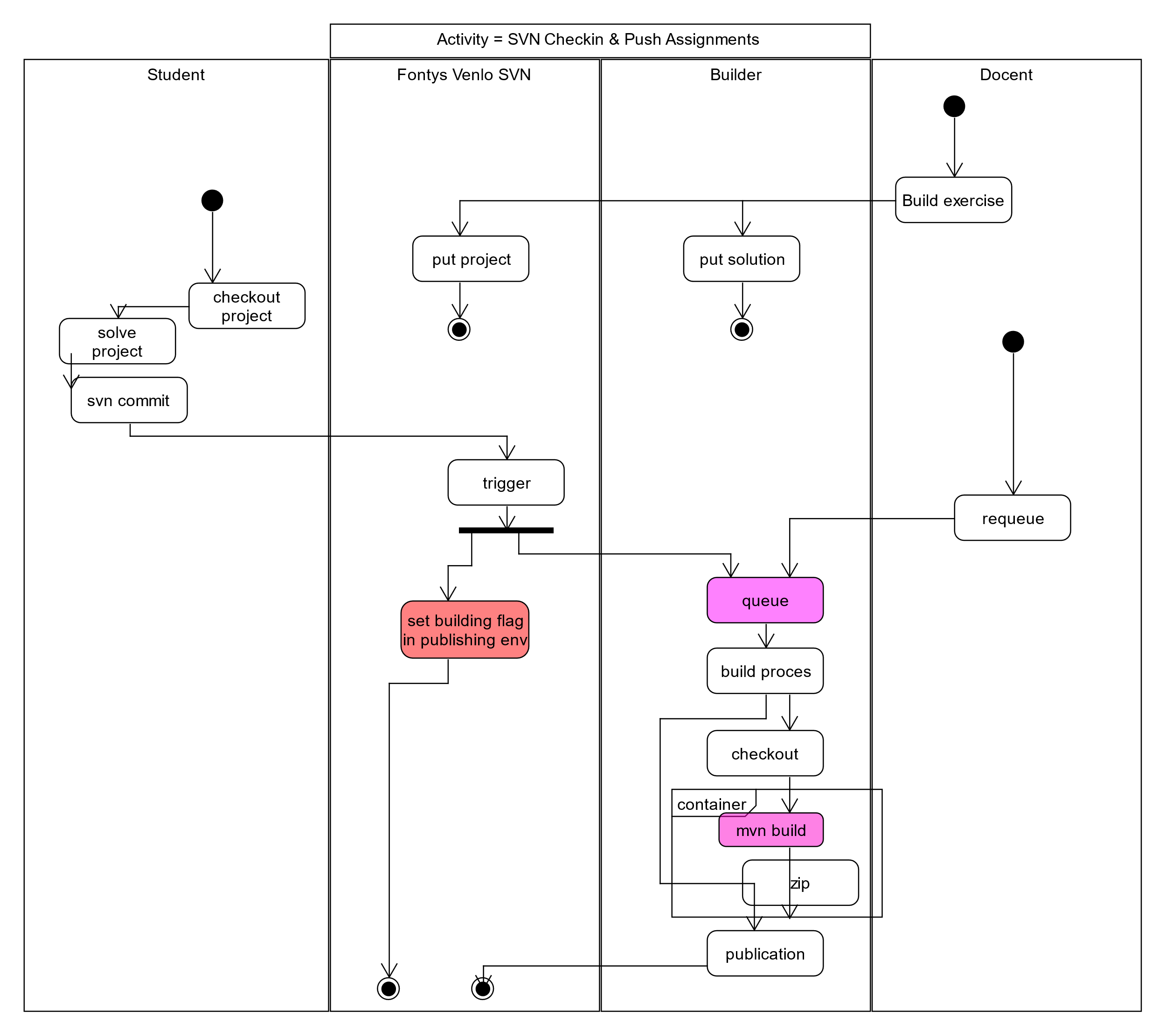


Figure 3 Activity Diagram: AsIs

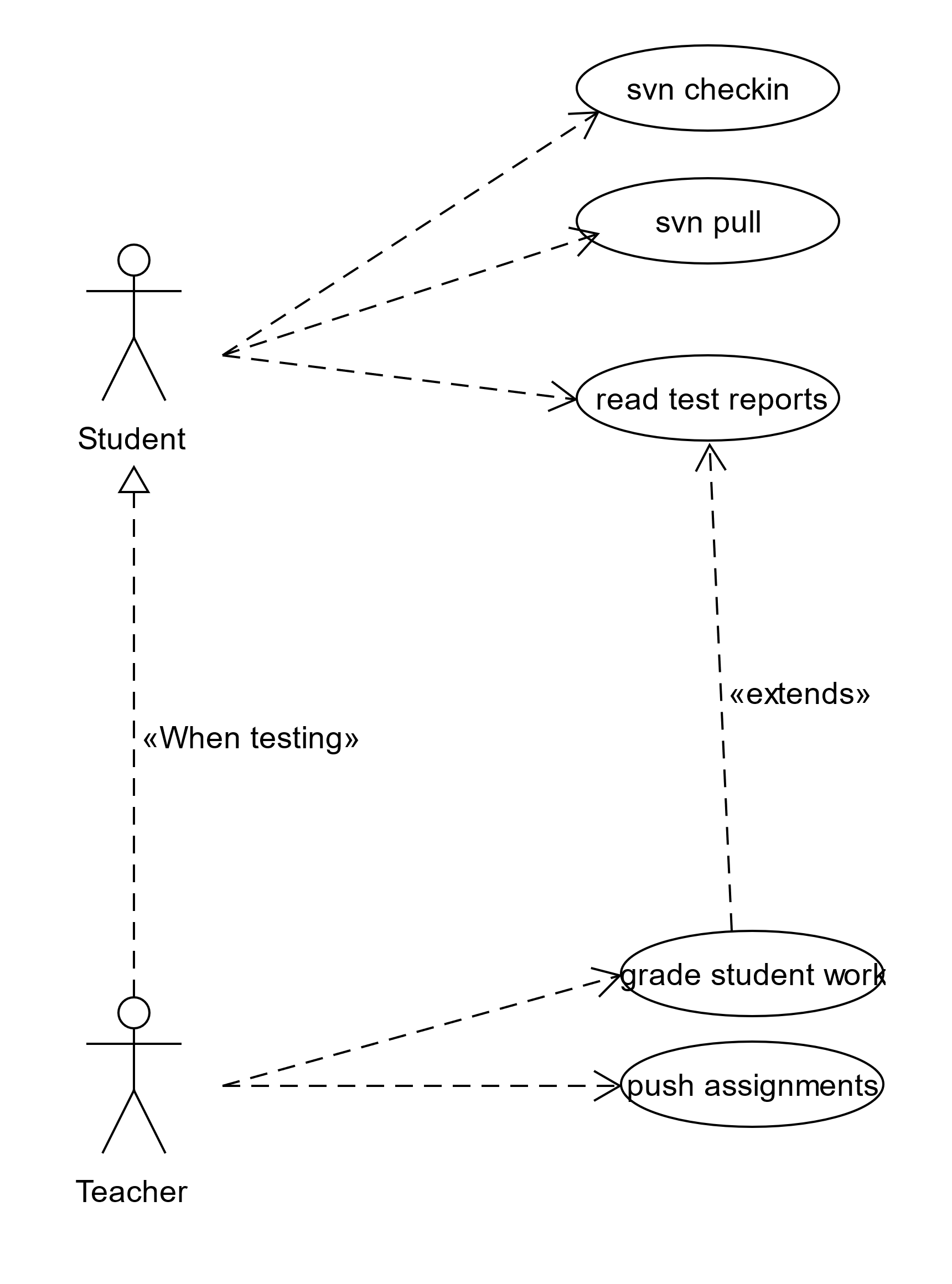


Figure 4 Usecase diagram

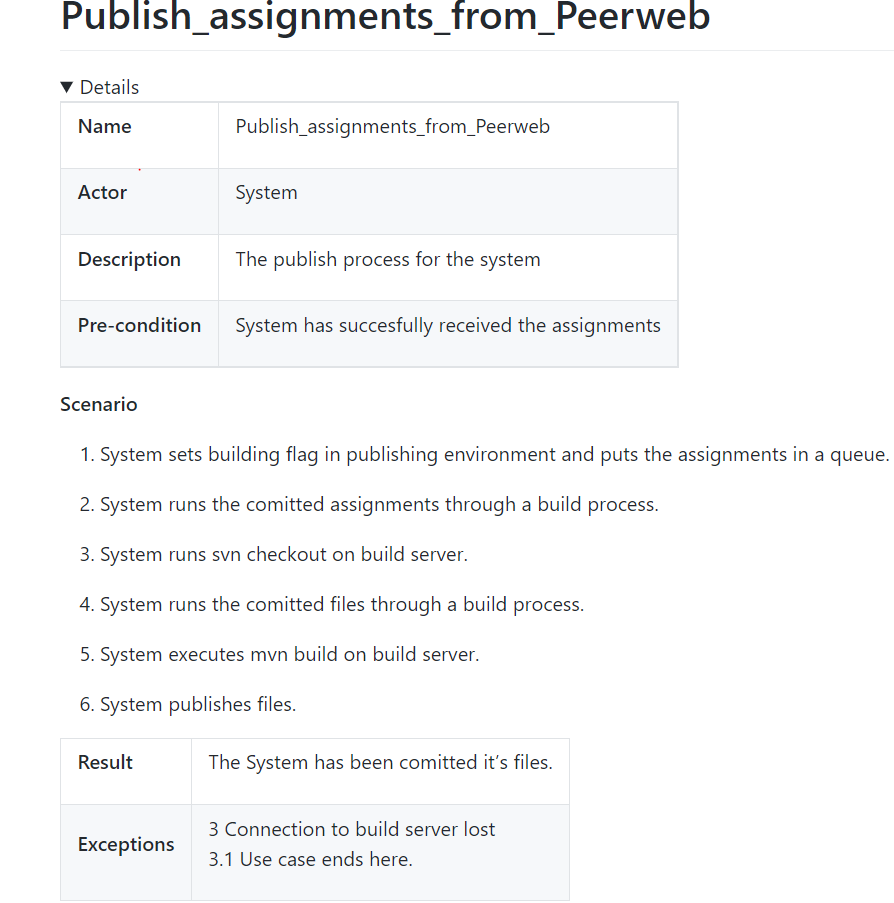


Figure 5 Publish assignment from peerweb

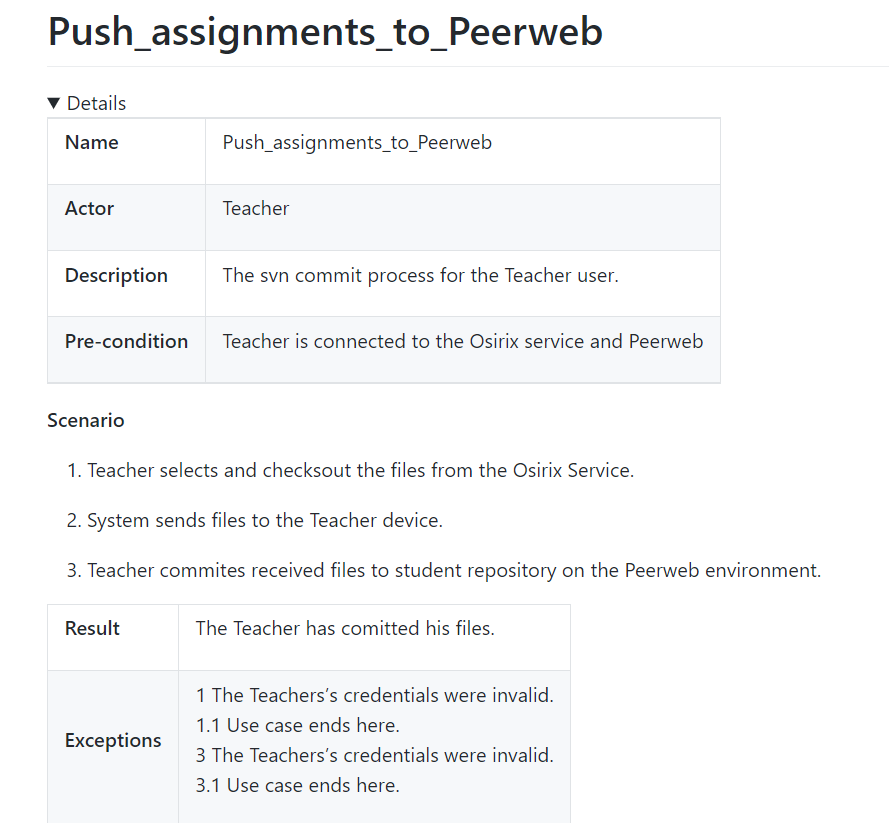


Figure 6push assignment to peerweb

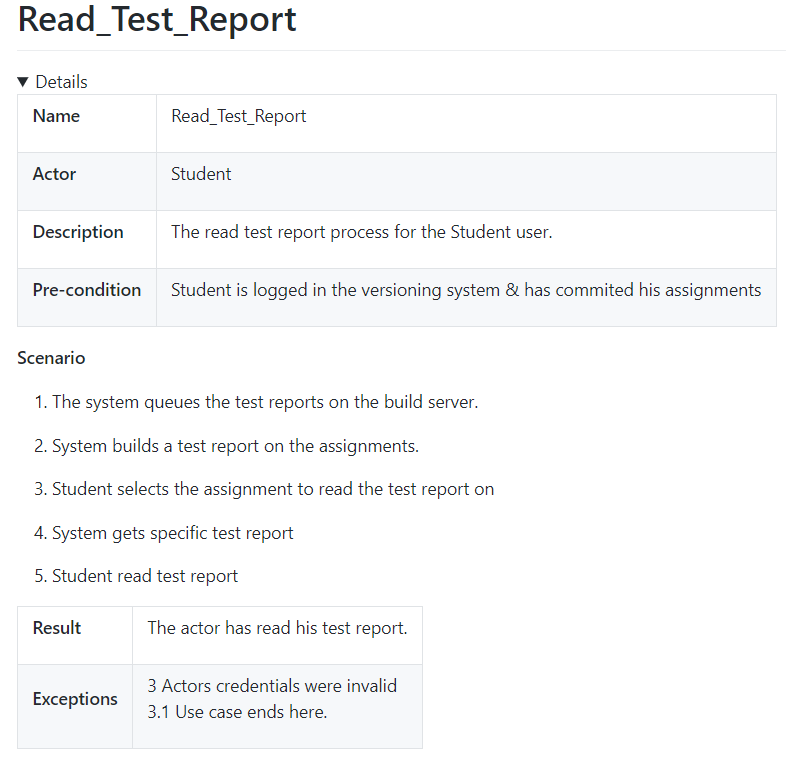


Figure 7 read test report

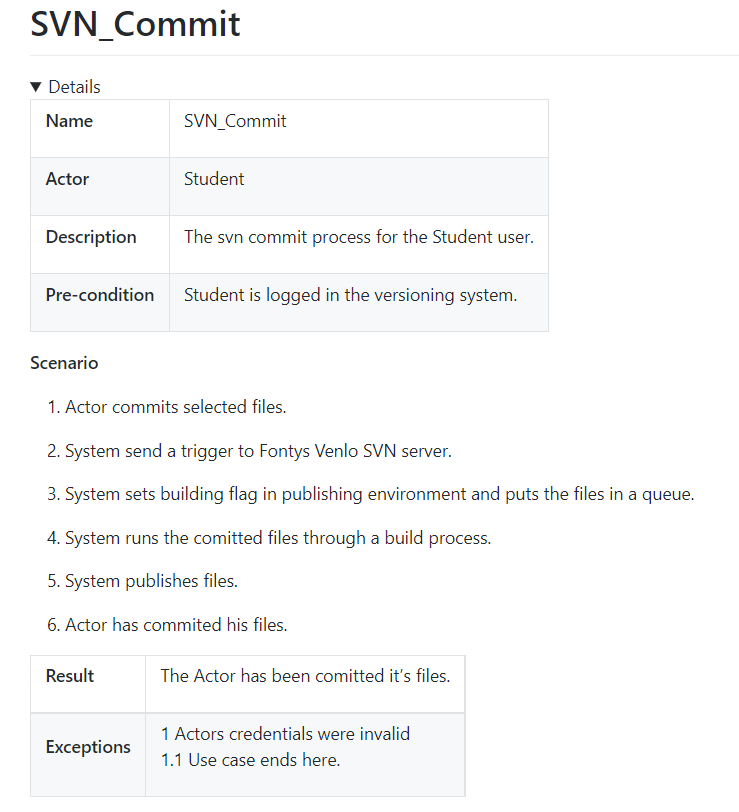


Figure 8 SVN Commit

# Conclusion & Recommendations

In this section there will be argued why certain automation services would be better for this particular case.

Firstly, I would like to mention that the services provided by GitHub makes a good automation tool for distributing, testing and grading student work. Especially GitHub Actions will provide you with the flexibility and ease of use continuous integration, which as an excellent example for student to follow.

Secondly GitHub classrooms can work along side GitHub actions really well by taking advantage of the repositories where those actions are defined in, as well as being an effective and automatable Student administration tool.

In this project sequence I managed to construct a prototype which allows for automated software testing and (maven) building. I also was able to make this work in collaboration with GitHub Classrooms. And more importantly, I provided a starting point on which my successors can build on.

# Literature List

* GitHub Guides
* YouTube
* Mozilla

# Resources

GitHub Actions Tutorial:

<https://docs.github.com/en/actions>

GitHub Actions Demo:

<https://youtu.be/cP0I9w2coGU>

**Summary GitHub Actions:**

You can automate all the actions in a normal github work flow(main, branch) +

But also add custom actions in between those normal actions (Custom software development lifecycle workflows directly in your GitHub repository. +

Github actions are event driven, meaning that you can run a series of commands after a specified event has occurred(like a pull request).

Github actions use YAML syntax to define the events, jobs and steps(this is stored in your repository @ .github/workflows

In this case push is the event that triggered the action, and uses steps to control the order in which actions are run which automate your software testing.

These steps are: GitHub actions checks out the pushed code, install the software dependencies, runs bats-v to output the software version

Furthermore you can build test and publish across multiple Operating systems and languages in one workflow.

Canvas API:

<https://developer.mozilla.org/nl/docs/Web/API/Canvas_API>

GitHub Actions Template Workflow:

<https://github.com/actions/starter-workflows/blob/055373ee0b531de9b779896c520d0555e7df48ae/ci/blank.yml>

GitHub Packages Explained:

<https://www.youtube.com/watch?v=N_-Cu9_2YAA>

**Summary GitHub Packages:**

Enabling sharing of packages that are to be trusted and that you can rely on

Fully integrated with github

Anyone with an account can publish registries

Works with npm, maven, docker, ruby registry protocols.

Unless you have a personal access token, you get not directly get packages available from github

ACCESS TOKENS can be given certain permissions.

GitHub Classroom Setup Tutorial:

<https://www.youtube.com/watch?v=KXWXg68KpTY>

Translating Test results to Canvas or other student administration systems:

<https://youtu.be/KXWXg68KpTY?t=702>

Creating Maven packages with GitHub:

<https://www.youtube.com/watch?v=MhzoxE7NdpI>

How to deploy a Maven Package:

<https://docs.github.com/en/actions/guides/building-and-testing-java-with-maven>

Sure Fire report Template:

<https://github.com/marketplace/actions/surefire-report>

JaCoCo Coverage:

<https://github.com/marketplace/actions/jacoco-report>

JaCoCo Badge:

<https://github.com/marketplace/actions/jacoco-badge-generator>

Conditional PIT Test with PR-Landmine:

<https://github.com/marketplace/actions/pr-landmine>

# Questions

1. Can GitHub do functionally the same as the current systems in place? With some adaptations, yes
2. If students have been imported from Canvas to GitHub Classrooms, can they still be added? Yes through a CSV file, if your GitHub Classroom is linked to a Canvas systems you’ll have to add the students there.
3. Can students access the GitHub actions? If they are defined in the assignment folder then yes, there is a solution in the work however in which the actions are called from a private repo in which cases the students won’t have access.
4. How to publish JaCoCo Coverage? Through Canvas or Email.
5. Where to the Maven build artefacts end up? In the target folder of your repository