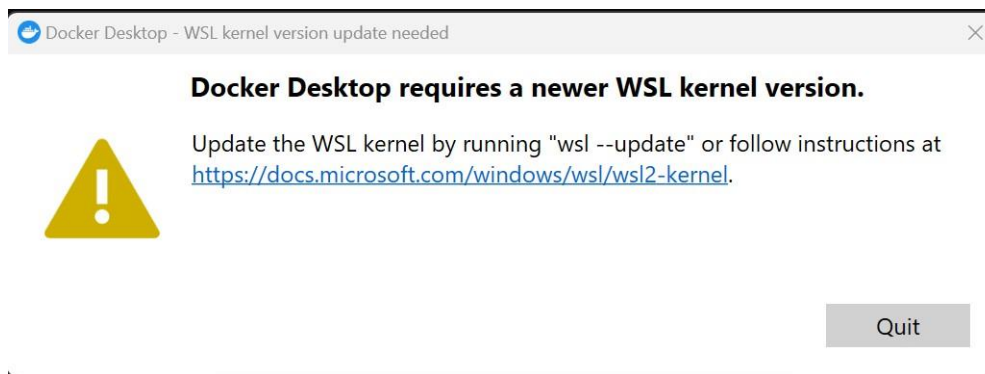


Test management tools Testlink and Jenkins

Remark: Testlink is not working for Mac.

1. Installing Jenkins and Testlink

- 1) Install Docker on your computer. <https://docs.docker.com/desktop/install/windows-install/>
- 2) Download the archive **vvss.zip**.
- 3) Unzip the archive in a directory.
- 4) Open a Command Prompt as administrator to the unzip directory.
- 5) Run the command: `docker-compose up -d`
- 6) If needed, use the newer WSL kernel version.



- 7) Accessing Testlink
<http://localhost:8090/>
user:student
password:student
- 8) Accessing Jenkins
<http://localhost:8080/>

user:student
password:student

Steps for using Testlink

1. User

<http://localhost:8090/>

user:student

password:student

2. Create a project

Each team will create a project for their SSVV project. Click the “Create” button and create
Example: Project_NameStudent1NameStudent2

The image shows two screenshots of the TestLink web application. The top screenshot displays the main menu with 'Test Project Management' highlighted. The bottom screenshot shows the 'Create a new project' form with the following details:

- Create from existing Test Project?**: No
- Name ***: Project_NameStudent1NameStudent2
- Prefix (used for Test case ID) ***: prjName11
- Project description**: (Empty text area)
- Enhanced features**:
 - ☒ Enable Requirements feature
 - ☒ Enable Testing Priority
 - ☒ Enable Test Automation (API keys)
 - ☒ Enable Inventory
- Issue Tracker Integration**: >> There are no Issue Tracker Systems defined <<
- Code Tracker Integration**: >> There are no Code Tracker Systems defined <<
- Availability**:
 - ☒ Active
 - ☒ Public

Buttons: Create, Cancel

* Mandatory fields

3. Assign custom fields

In the Project Menu, in the upper right corner select from the combo box your Test Project.

In the Project Menu, select Assign Custom Fields, select both JavaClassName and JavaTestMethodName and click button Assign.

The image shows two screenshots of the TestLink web application interface. The top screenshot shows the 'Test Project' dropdown menu in the upper right corner, with an arrow pointing to it. The left sidebar menu is open, and an arrow points to the 'Assign Custom Fields' option. The bottom screenshot shows the 'Assign custom fields' page for the selected test project. It features a table of available custom fields and an 'Assign' button.

Available custom fields

<input checked="" type="checkbox"/>	Name	Label	Type	Available for
<input checked="" type="checkbox"/>	JavaClassName	name of the autotest class	string	Test Case
<input checked="" type="checkbox"/>	JavaTestMethodName	name of the test method	string	Test Case

Assign

Each team will create his/her own Test Plan, the name of the test plan will be composed by groupNumber followed by the names of the team members concatenated with TestPlan.

93X_Name01Name02_TestPlan

[illegible]

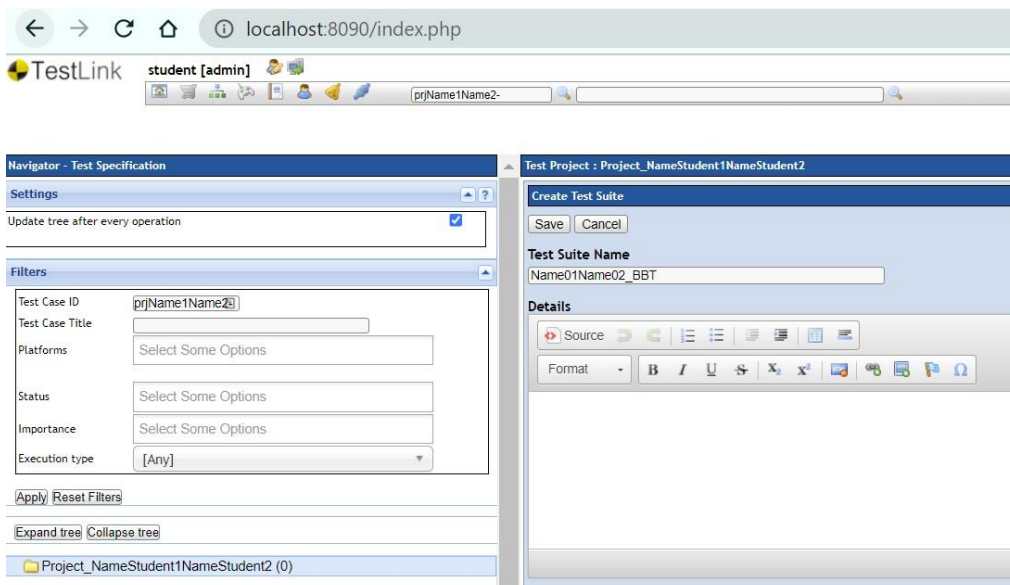
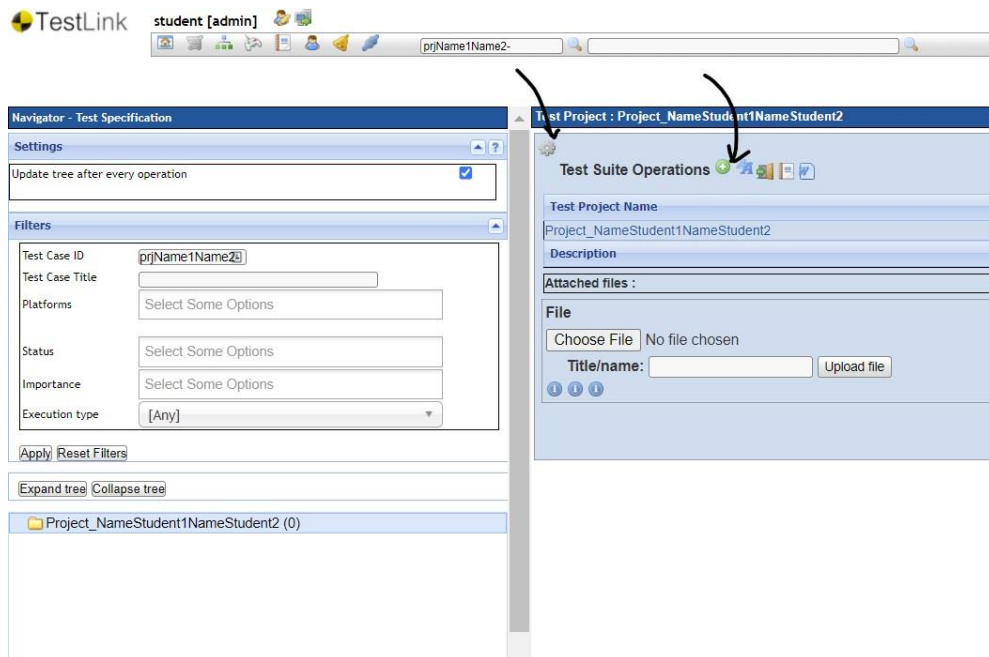
5. Create Test Suites/ Test Cases

Each student will create in total 3 test suites:

- 1 test suite from laboratory 2 - black-box testing
 - Test Suite Name = **Name01Name02_BBT**
- 1 test case from laboratory 3 - white-box testing
 - Test Suite Name = **Name01Name02_WBT**
- 1 test case from laboratory 4 - integration testing.
 - Test Suite Name **Name01Name02_IntegrationT**

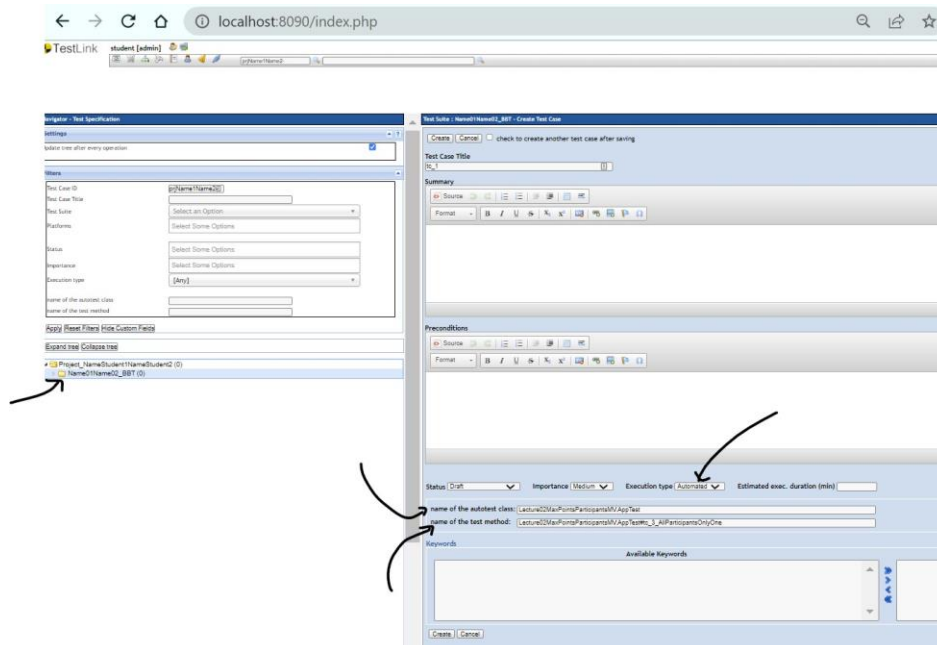
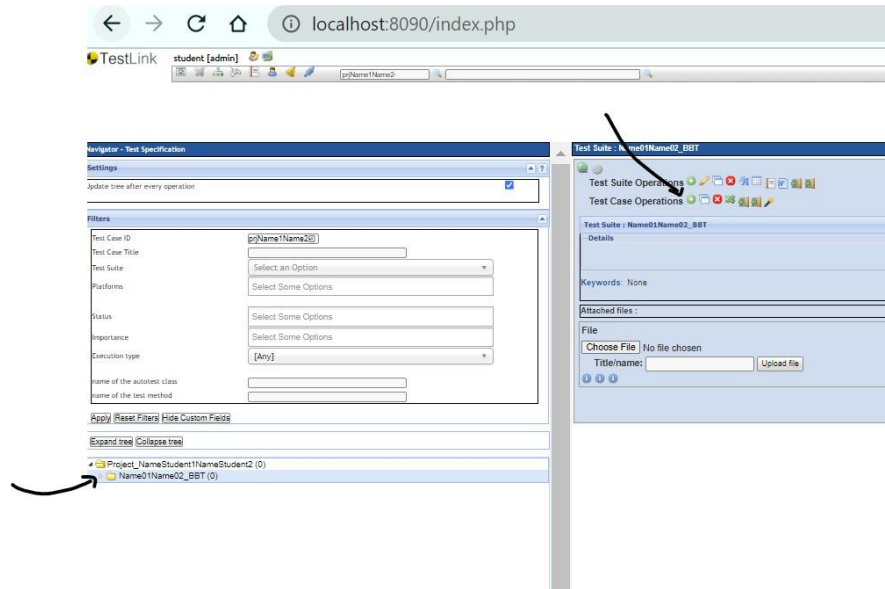
Create Test Suites - STEPS

- Project/Desktop menu --> Test Specification section --> click on Test Specification
- Select your test project in navigation tree (left side screen)
- In the Test Suite Operations panel click on the Action button and then on the Create button.



6. Create Test Cases - STEPS

- Select your test suite in navigation tree (left side screen)
- In the Test Suite Operations/ Test Case Operations panel click on the Create button (from Test Case Operation). Figure 6a and Figure 6b. **Remember to modify Execution type to automated.**



7. Add a Test Case to a Test Plan

The screenshot shows the TestLink web application interface. The browser address bar displays 'localhost:8090/index.php'. The user is logged in as 'student [admin]'. The main content area is the 'Test Case' form, which includes sections for 'Test Case Operations', 'Test Case Version Operations', 'Summary', 'Preconditions', 'Create step', 'Status', 'Importance', 'Execution type', 'Estimated exec. (min)', 'name of the autotest class', 'name of the test method', 'Keywords', 'Platforms', 'Requirements', 'Relations', and 'Attached files'. The 'Test Case Version Operations' section has a button labeled 'Add to Test Plan'. The left sidebar, titled 'Test Specification', shows a tree view of test cases. A test case named 'prjName1Name2-1-to_1' is selected, and an arrow points to it. Another arrow points to the 'Add to Test Plan' button in the main form.

The screenshot shows the TestLink web application interface. The browser address bar displays 'localhost:8090/index.php'. The user is logged in as 'student [admin]'. The main content area is the 'Test Case' form, which includes sections for 'Test Case Operations', 'Test Case Version Operations', 'Summary', 'Preconditions', 'Create step', 'Status', 'Importance', 'Execution type', 'Estimated exec. (min)', 'name of the autotest class', 'name of the test method', 'Keywords', 'Platforms', 'Requirements', 'Relations', and 'Attached files'. The 'Test Case Version Operations' section has a button labeled 'Add to Test Plan'. The left sidebar, titled 'Test Specification', shows a tree view of test cases. A test case named 'prjName1Name2-1-to_1' is selected, and an arrow points to it. Another arrow points to the 'Add' button in the 'Test Plan usage' table.

Steps for using Jenkins

1. User

Accessing Jenkins

<http://localhost:8080/>

user:student

password:student

2. Create a new Job

Each team will create his/her own **Job (for the TestPlan created in Testlink)**, the name of the job will be composed of: groupNumber followed by the names of the team members concatenated with the word Job. (93X_Name01Name02_Job)

- for example, for one of the team in group 931 Pop and Popescu the name of the job will be:
 - 931_PopPopescu_Job.

Create a new Job from other existing job

1. In the Dashboard, click on the +NewItem
2. Use “Copy from”: Job_Lecture02_BBT

localhost:8080/view/all/newJob

Jenkins

Dashboard > All >

Enter an item name

93X_Name1Name2_Job

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (declarative and scripted).

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is a virtual container, items are actually stored in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from
Job_Lecture02_BBT

OK

3. Modify in your Job the following elements

a. Click on the created Job and click on the Configure (left side menu)

b. Github link with your Github link

c. Test Project and Test Plan with

- Lecture01_Demo_BBT with [Project_NameStudent1NameStudent2](#) (the name of the Project that you created in Testlink)

- Lecture02_TestPlan_BBT with [93X_Name01Name02_TestPlan](#) (the name of the Test Plan that you created in Testlink).

localhost:8080/job/93x_Name1Name2_Job/configure

Dashboard > 93x_Name1Name2_Job > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

☐ Discard old builds ?

☒ GitHub project

Project url ?

☐ This project is parameterized ?

☐ Throttle builds ?

☐ Execute concurrent builds if necessary ?

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

localhost:8080/job/93x_Name1Name2_Job/configure

Dashboard > 93x_Name1Name2_Job > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Invoke TestLink

TestLink Configuration

TestLink Version ?

Test Project Name ?

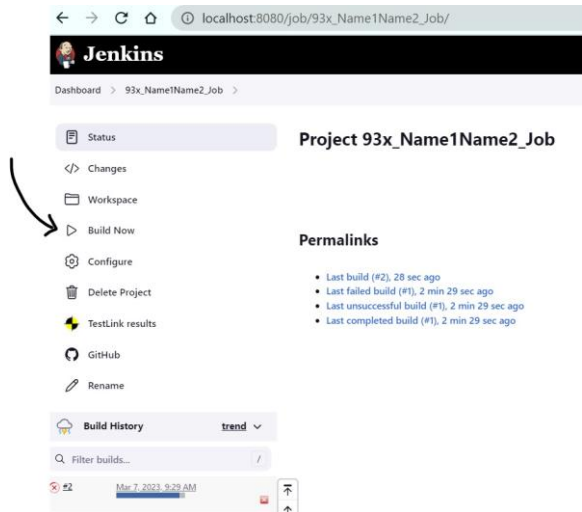
Test Plan Name ?

Build Name ?

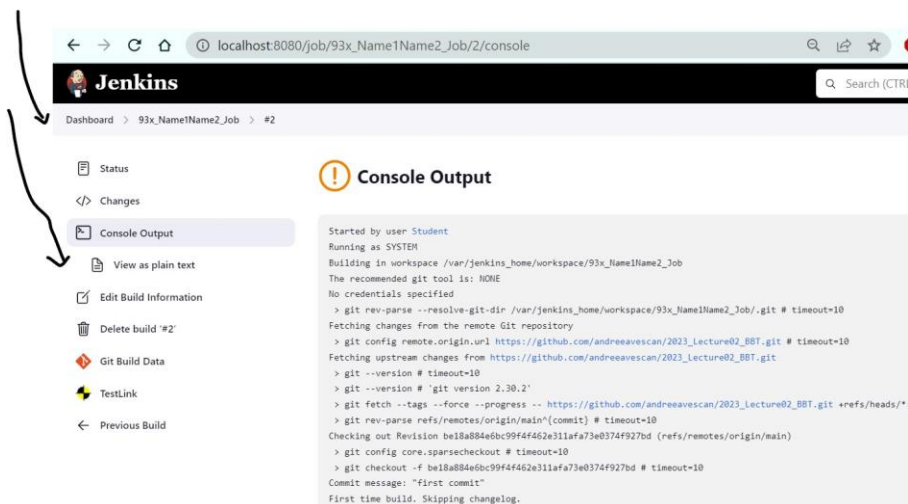
Custom Fields ?

4. Build

- Select the Job
- Select Build Now



- Console Output



5. Sending results from Jenkins to Testlink

After successfully running the job, in Testlink it is updated automatically the status of the test cases in Not Run, Passed, Failed, or Blocked.

6. View the state of a Test case in Testlink

In the Execute Tests menu, within the chosen project, eg Projects931, a desired Test Suite is selected, e.g., abie1234_BBT, and a test case.

The screenshot displays the Testlink web interface. On the left, the 'Execute Tests' sidebar is visible, showing settings for a test plan (93X_Name01Name02_TestPlan) and build (build_3). The 'Filters' section shows a search for 'prjName1Name2'. The main panel shows 'Test Results on Build build_3' for the test plan '93X_Name01Name02_TestPlan'. It lists the build 'build_3' and the test suite 'Name01Name02_BBT'. Below this, a table shows the 'Last execution (any build)' with a single entry: Date: 03/07/2023 09:39:51, Tested by: student, Build: build_3, Status: Passed. A second table shows the 'Last execution (current build) - Build: build_3' with a single entry: Date: 03/07/2023 09:39:51, Build: build_3, Tested by: student, Status: Passed. A yellow box indicates 'Test Case prjName1Name2-1 - Version: 1 - lc_1' with 'No testor assigned'. The bottom section shows 'Execution type: Automated' and 'Estimated exec. duration (min):'. Arrows point to the 'Test Case prjName1Name2-1' and the 'Status: Passed' in the execution table.

Date	Build	Tested by	Status	Exec (min)
03/07/2023 09:39:51	build_3	student	Passed	