DADock - Tutorial

V2_1806, Fri Aug 17 2018 03:57:29 GMT+0000 (UTC)

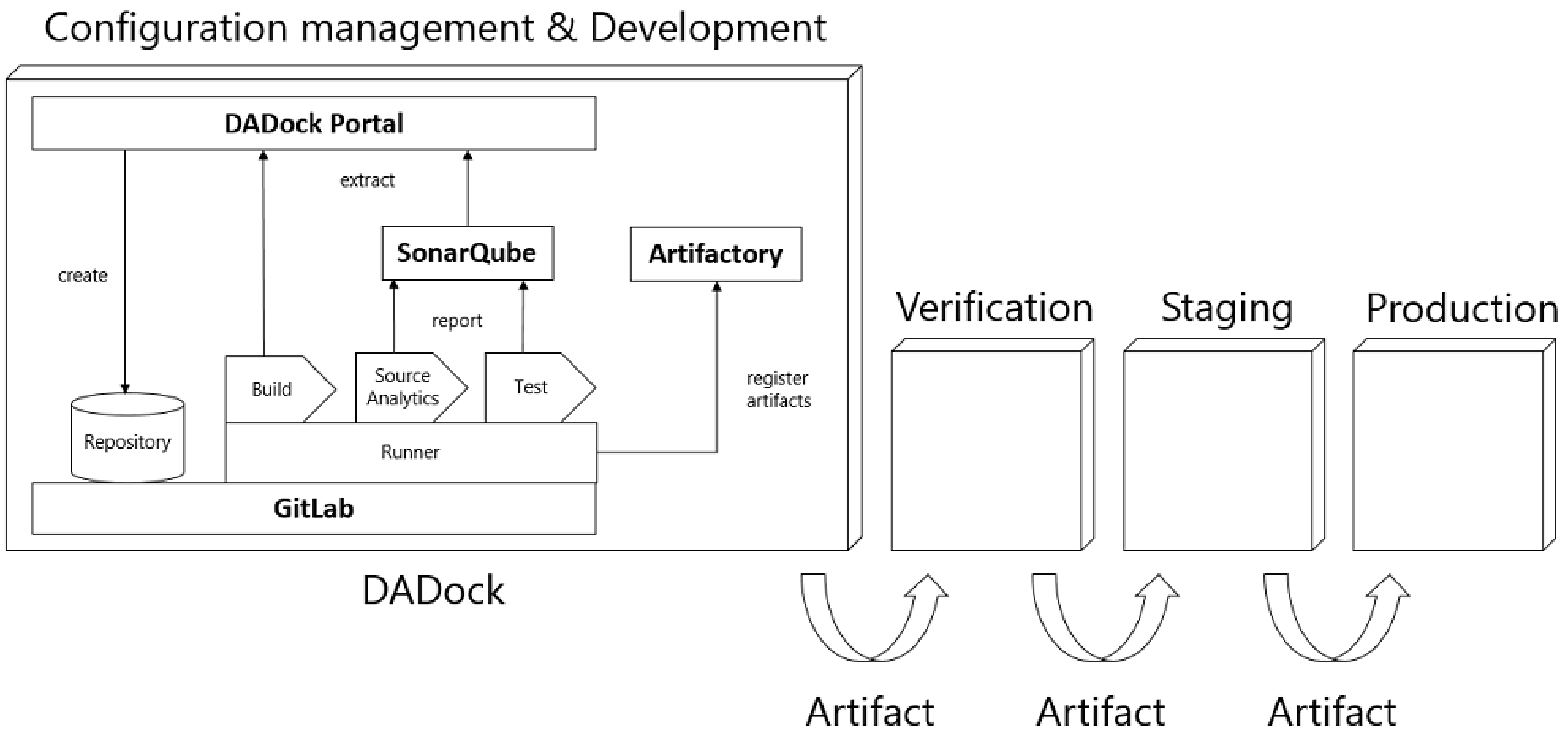
Table of contents

Introduction

Welcome to DADock Tutorial!

The diagram below shows an overall image of the project from configuration management up to the release phase with the use of DADock. First, the configuration management for development in DADock is implemented. After that, when the artifact during the development is done, the artifact is then deployed in a verification environment for checking purposes. In case there are no issues in the artifact after checking in the verification environment, the artifact will be deployed in the staging environment for final checking. Finally, if in case there are no issues after checking in the staging environment, the artifact will be released in the production environment.

This tutorial will allow the user to experience the configuration management phase up to development phase of the project.



Features when developing using DADock

This section explains the features of DADock when it is used in project development.

Source code sharing with the use of Git repository

In DADock, source codes are managed with the use of Git repository in GitLab. Each developer can push changes in the project one or more times a day, while maintaining the status that the shared source code is still running and working. With this, redoing a large scale of work can be controlled and checking the actual status progress of the project becomes easier.

Build/Static Analysis of source code/Automation of unit test

In DADock, there is already a template prepared for automatic execution of build, static analysis of source code, and unit tests during every push of the developer. The developer will then check the results of the automation from the following check points and make the necessary changes when needed.

- no build errors
- no source code static analysis errors
- all items in the unit test are successfully working
- the unit test coverage reached the quality index value (Ex: 80% - the standards are set depending on the project)

With these, the source code can constantly maintain a high level of quality.

1. Setup of working environment for this tutorial

This chapter explains the environment settings and the kinds of tools needed in the client machine, which is to be used in this tutorial.

1.1. Prerequisite of DADock settings

As a prerequisite of setting up DADock, it is a must that installation of DADock is already finished. Check the following list and should there be any issues, contact the administrator.

- Administrator's user name and password of DADock
- DADock Portal can already be accessed from the browser and login is possible with the administrator acount
- The Dashboard of DADock Portal is properly displayed

1.2. Varieties of tool to be used in this tutorial

This section explains the kinds of tools that will be used in this tutorial. The details on how to install each tool can be referred in Appendix.

About the installation of each tool

In case a different version of the tool indicated in this tutorial or an alternative tool is already installed in the client machine, the user can skip the installation of that particular tool.

Git

Git is a distributed version management system. This tutorial explains the flow of development by multiple teams done at the same time by using Git.

TortoiseGit

TortoiseGit is the GUI (Graphical User Interface) client of Git tool. If in case other GUI client is to be used, installation of TortoiseGit is not needed. However, this tutorial is based on the premise that TortoiseGit is already installed in the client.

JDK8

JDK8(Java SE Development Kit 8) is an environment used for development and execution of Java source codes.

Eclipse and related plugins

Eclipse is an integrated development environment mainly for Java language. This is used for modification of source codes and for execution of unit test.

Gradle

Gradle is a build tool mainly for Java language source codes. Since the repositories created by DADock uses Gradle Wrapper, installation of Gradle is no longer needed. However, in case the development is done in a proxy environment, it is a must to set the proxy of Gradle Wrapper.

Tomcat

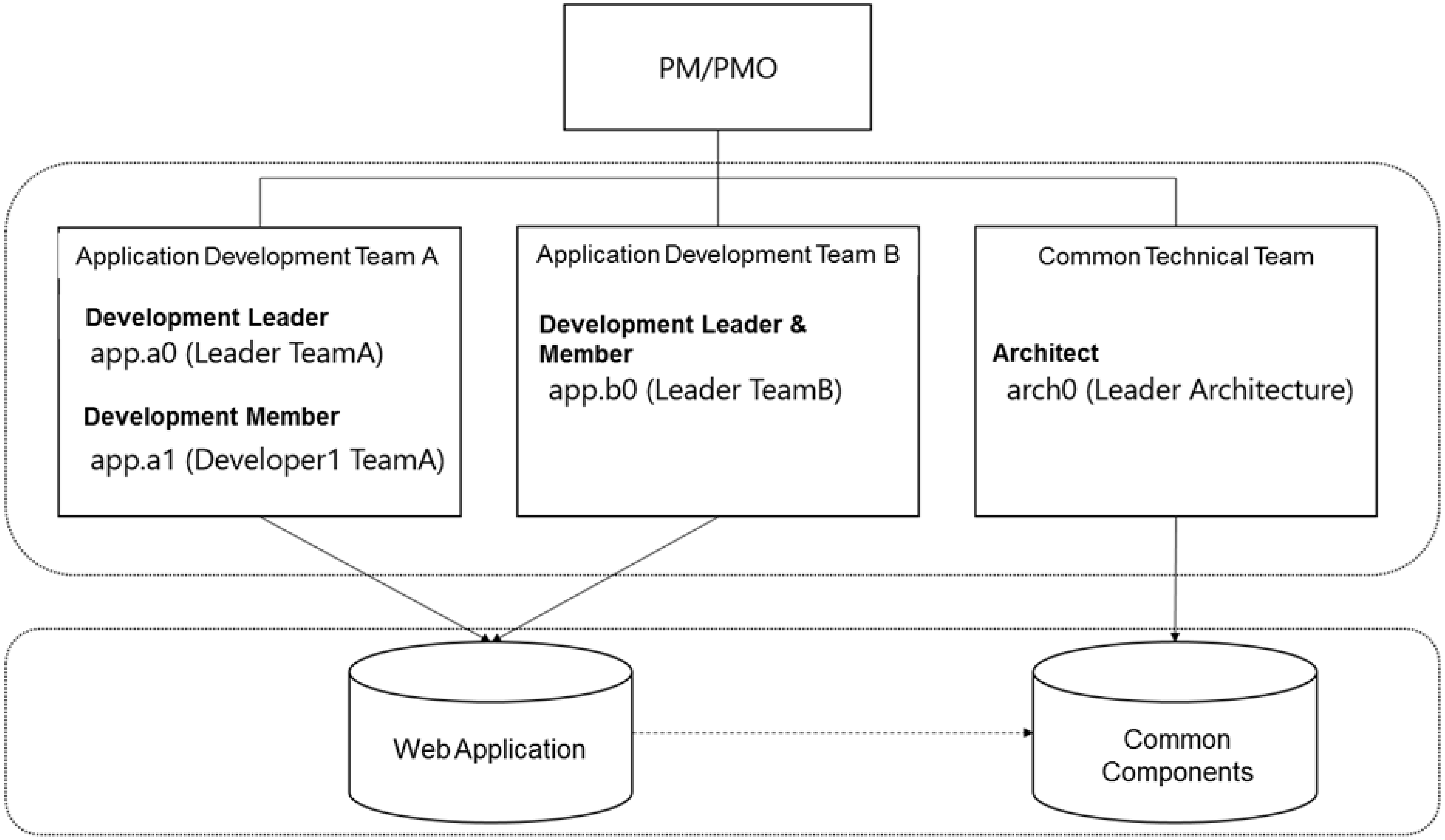
Tomcat is a servlet container tool. This is used for the deployment and operations of web application, which is created in this tutorial.

2. Tutorial outline

In this tutorial, development of Java web application with the use of DADock will be learned. In this chapter, the expected project will be explained. Within this tutorial, roles other than developer user will also be learned.

2.1. Project outline

The overall diagram of the project is shown below.



2.1.1. Structure

PM/PMO

Manager of the project. Manages the configuration of the users and creation of new repositories etc. in DADock.

Application Development Team A

A structure of 2 members team consisting of 1 development leader and 1 developer for developing web application.

Application Development Team B

A structure of 1 member team having the role of both the development leader developer for developing web application.

Common Technical Team

A structure of 1 member team having the role of an architect, who develops common components used by the web application.

2.1.2. Repository structure

Web application

A project for developing web application. The project is commonly used by both Team A and Team B. This is the web application developed by the Common Technical Team as a common component for the development.

list 1. Directory Framework of the Web Application

```
TutorialWeb
└─src
    ├─main
    │  ├─java
    │  │  └─project
    │  │      └─web
    │  │          ├─teama
    │  │          └─teamb
    │  └─webapp
    │      ├─META-INF
    │      └─WEB-INF
    │          └─jsp
    │              ├─teama
    │              └─teamb
    └─test
        └─java
            └─project
                └─web
                    └─teama
                    └─teamb
```

Application Development Team A saves assets in the following directory.

```
src/main/java/project/web/teama
src/main/webapp/WEB-INF/jsp/teama
src/test/java/project/web/teama
```

Application Development Team B saves assets in the following directory.

```
src/main/java/project/web/teamb
src/main/webapp/WEB-INF/jsp/teamb
src/test/java/project/web/teamb
```

Common Components

The project wherein the Common Technical Team develops the common components used by the Web Application.

list 2. Directory Framework of Common Components repository

```
TurorialLib
└─src
    ├─main
    │  └─java
    │      └─project
    │          └─lib
    └─test
        └─java
            └─project
                └─lib
```

3. Configuration management

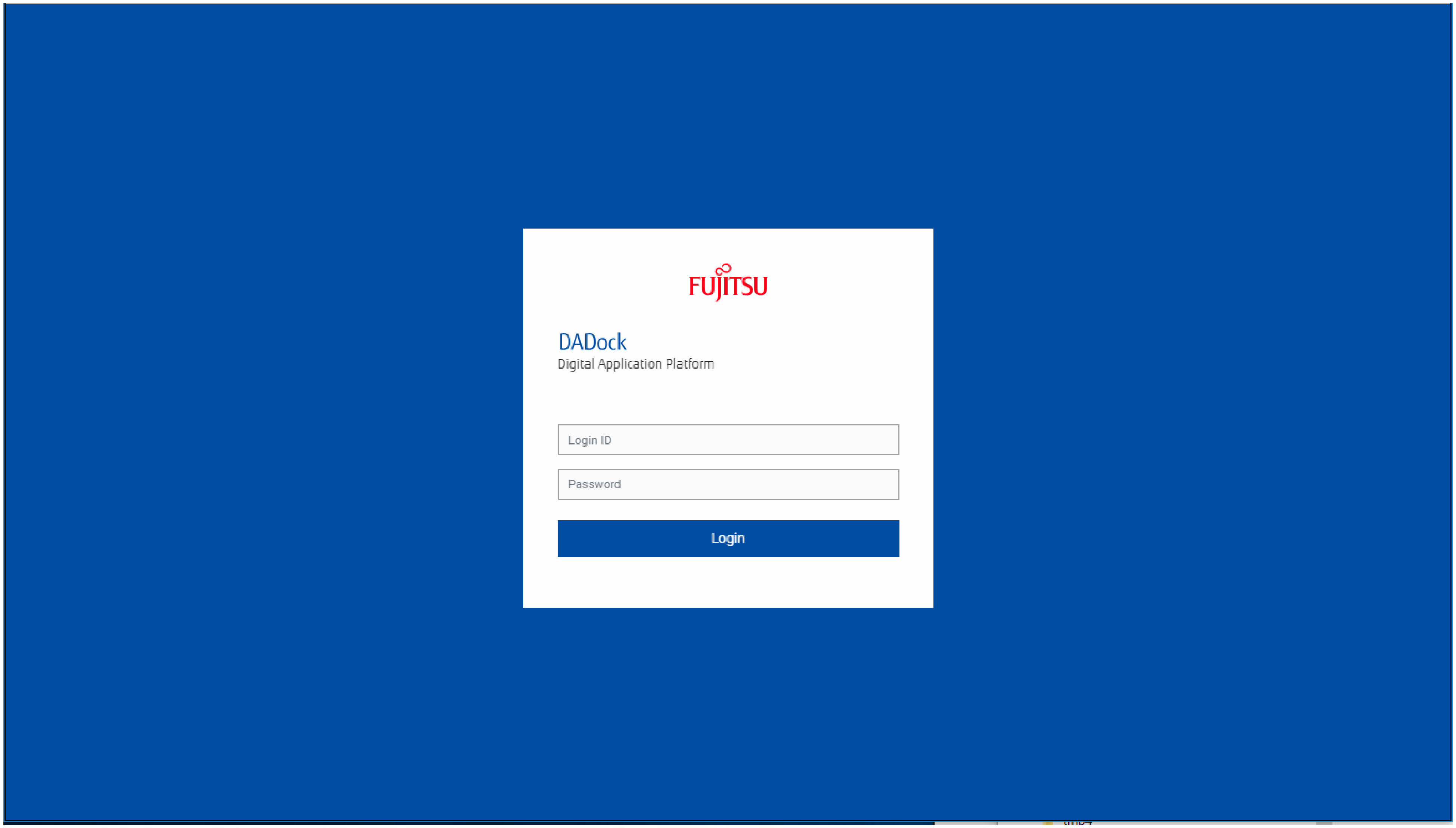
Configuration management is performed by the person who logs in as a privileged user to DADock in the role of PM / PMO. In here, the contents of the configuration management is a prerequisite of the development using DADock.

3.1. Create user

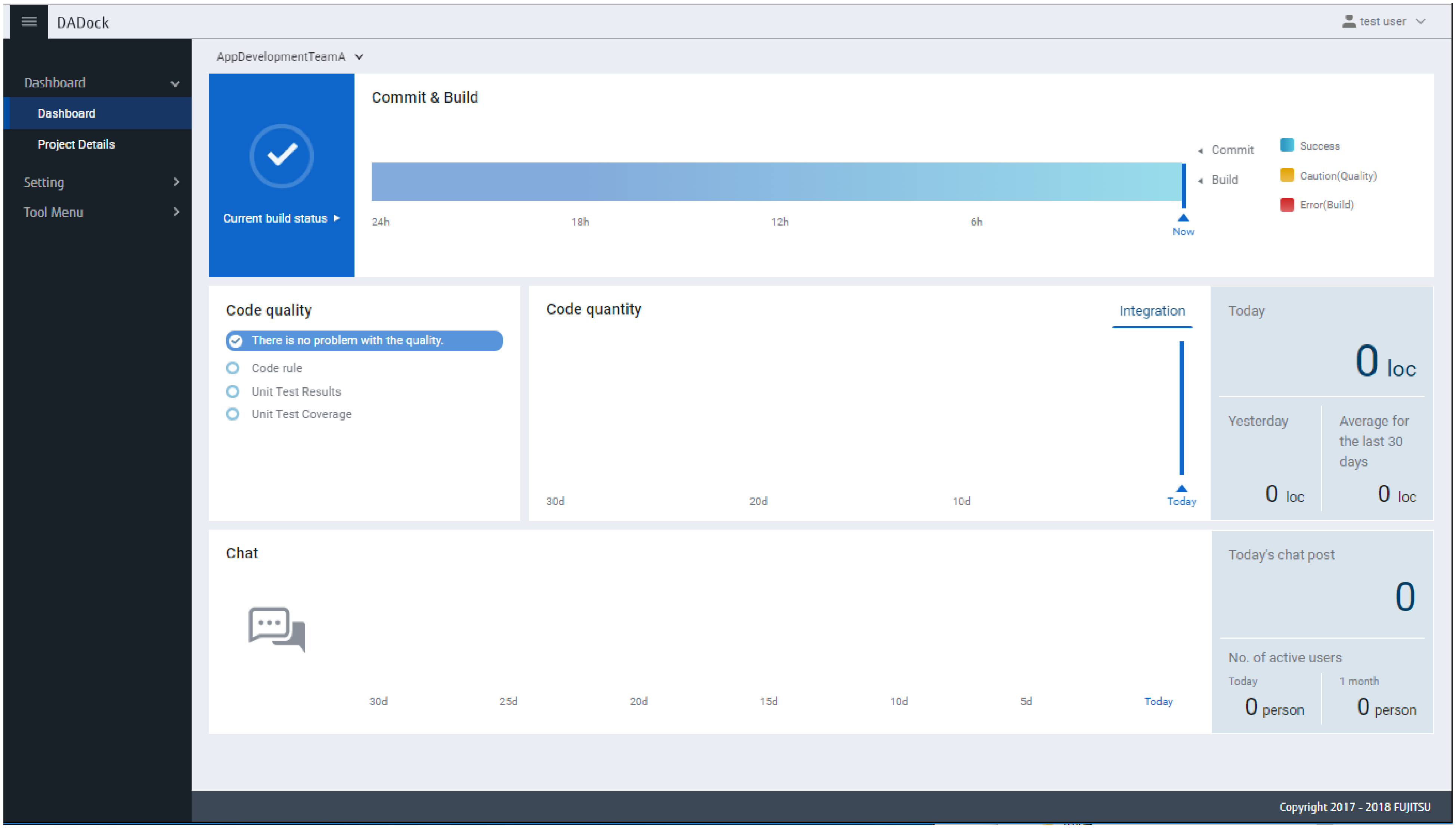
Create a project member user who will be using DADock.

3.1.1. Login to DADock

Input the URL of DADock Portal in the browser and the Login Page will be displayed.

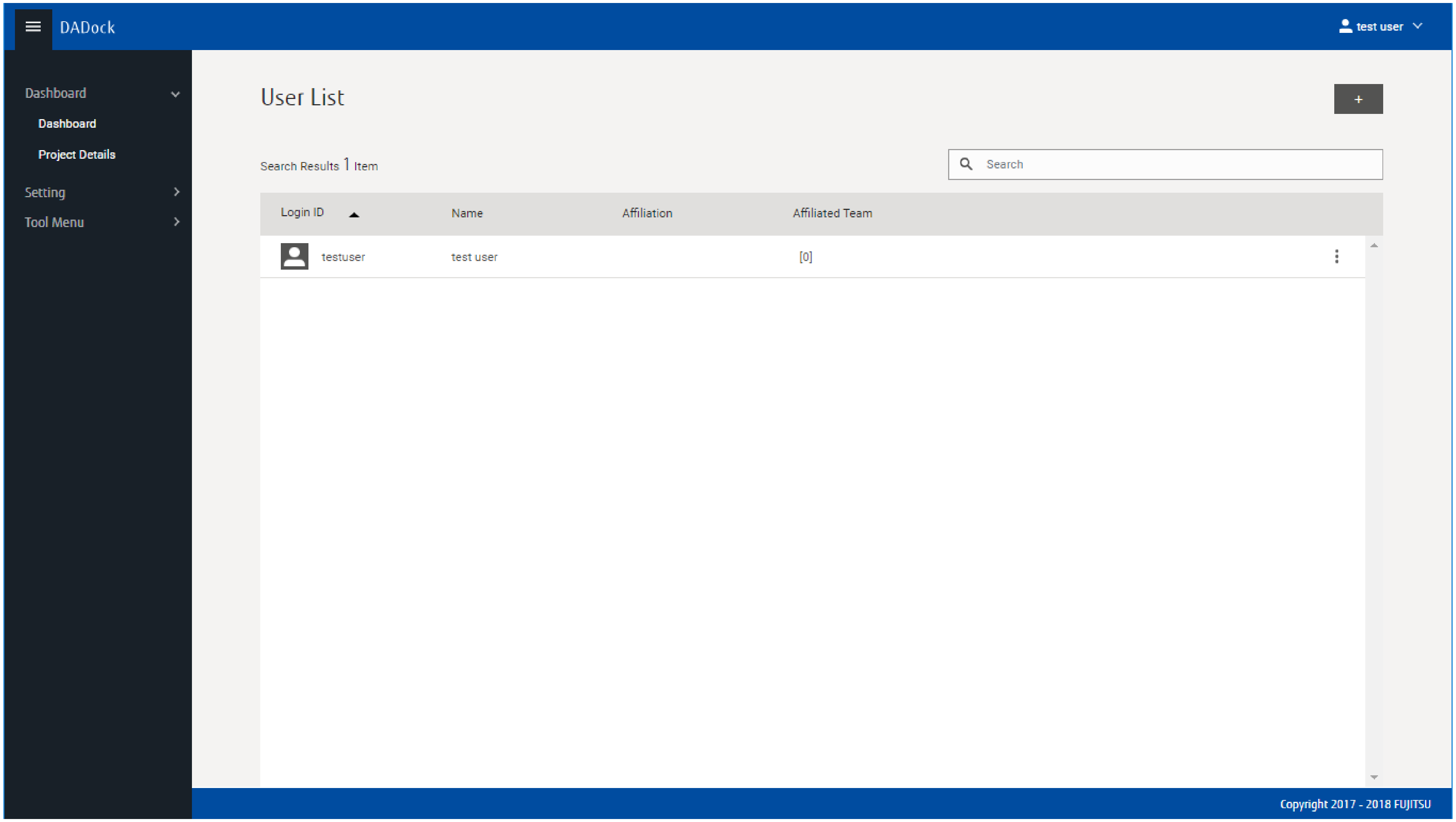


Input the User ID and Password of the administrator in the displayed fields and click the [Login] button then the Dashboard Page of DADock Portal will be displayed.



3.1.2. Create new user

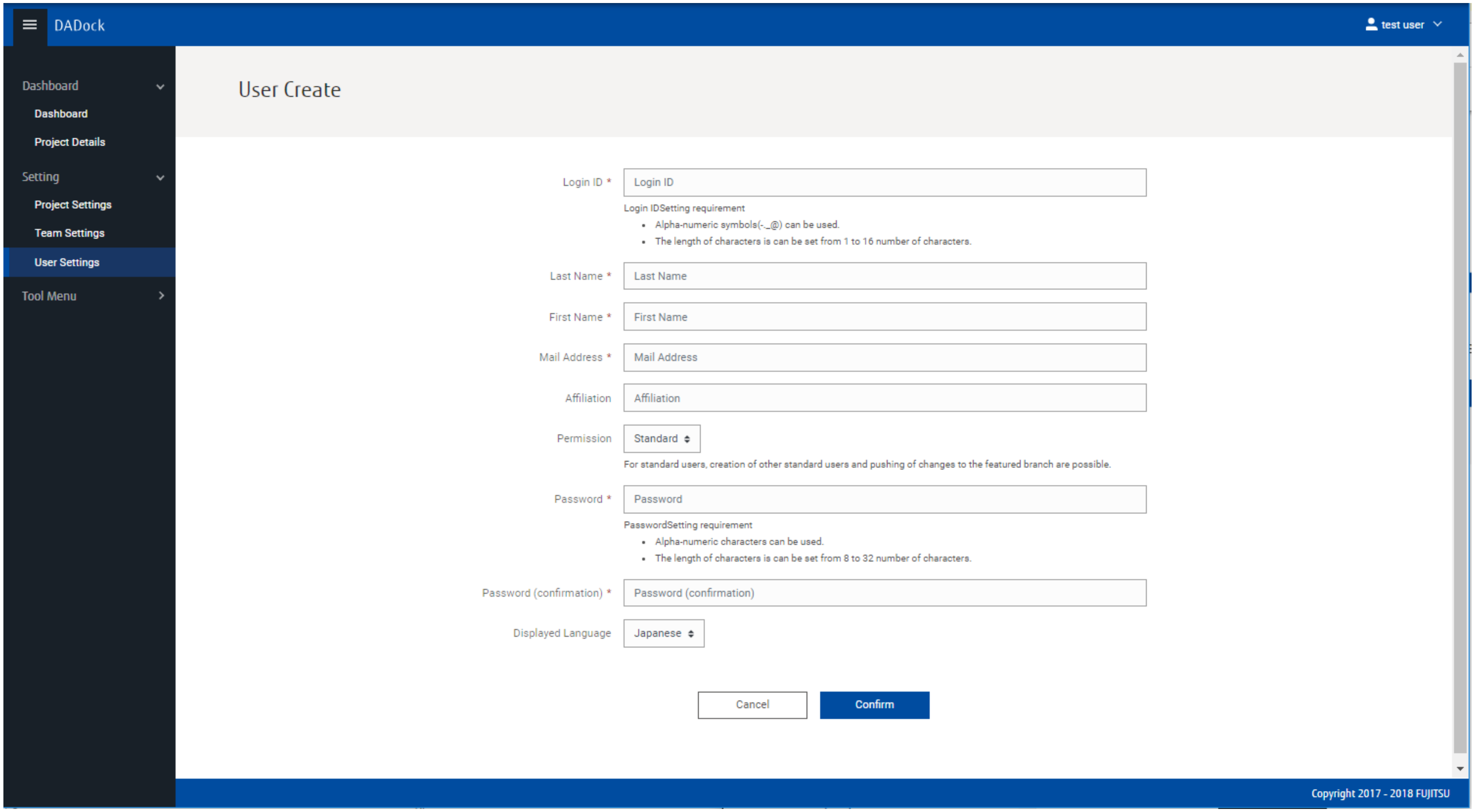
From the Dashboard Page of DADock Portal, expand the [Settings] tab then click [User Settings] to display the User List Page.



Search Results 1 Item

Login ID	Name	Affiliation	Affiliated Team
testuser	test user		[0]

At the upper right part of the User List Page, click [+] button and the User Create Page will be displayed.



User Create

Login ID * Login ID

Login IDSetting requirement
- Alpha-numeric symbols(-._@) can be used.
- The length of characters is can be set from 1 to 16 number of characters.

Last Name * Last Name

First Name * First Name

Mail Address * Mail Address

Affiliation Affiliation

Permission Standard

For standard users, creation of other standard users and pushing of changes to the featured branch are possible.

Password * Password

PasswordSetting requirement
- Alpha-numeric characters can be used.
- The length of characters is can be set from 8 to 32 number of characters.

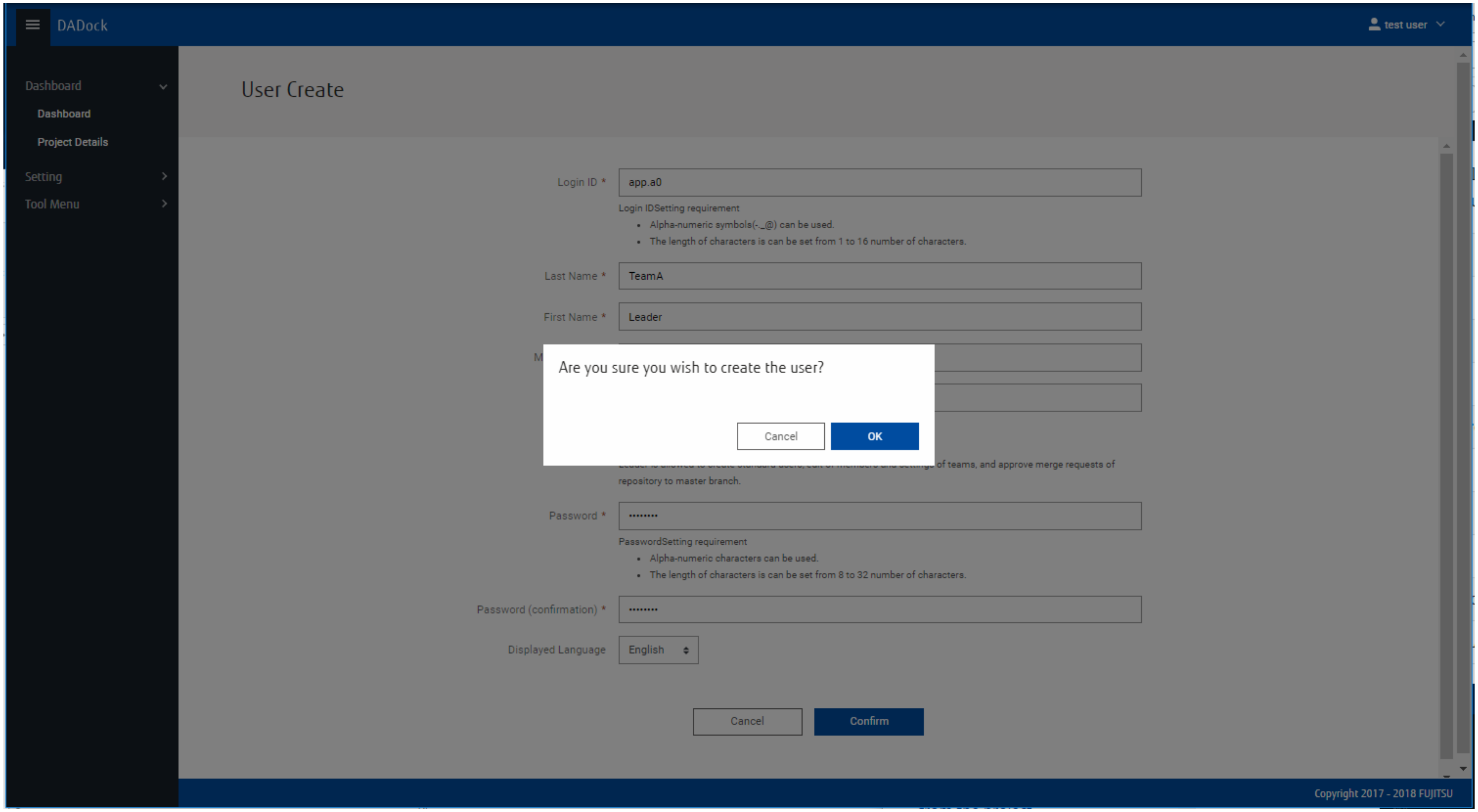
Password (confirmation) * Password (confirmation)

Displayed Language Japanese

Input the following account information of the development leader for Application Development Team A and click the [Confirm] button to save the entry.

Field	Value
Login ID	app.a0

Field	Value
Last Name	TeamA
First Name	Leader
Email Address	app.a0@example.com
Affiliation	<leave as blank>
Permission	Leader
Password	password
Password (confirmation)	password
Displayed Language	English



User Create

Login ID * app.a0

Login IDSetting requirement

- Alpha-numeric symbols(-._@) can be used.
- The length of characters is can be set from 1 to 16 number of characters.

Last Name * TeamA

First Name * Leader

Are you sure you wish to create the user?

of teams, and approve merge requests of repository to master branch.

Password * ••••••••

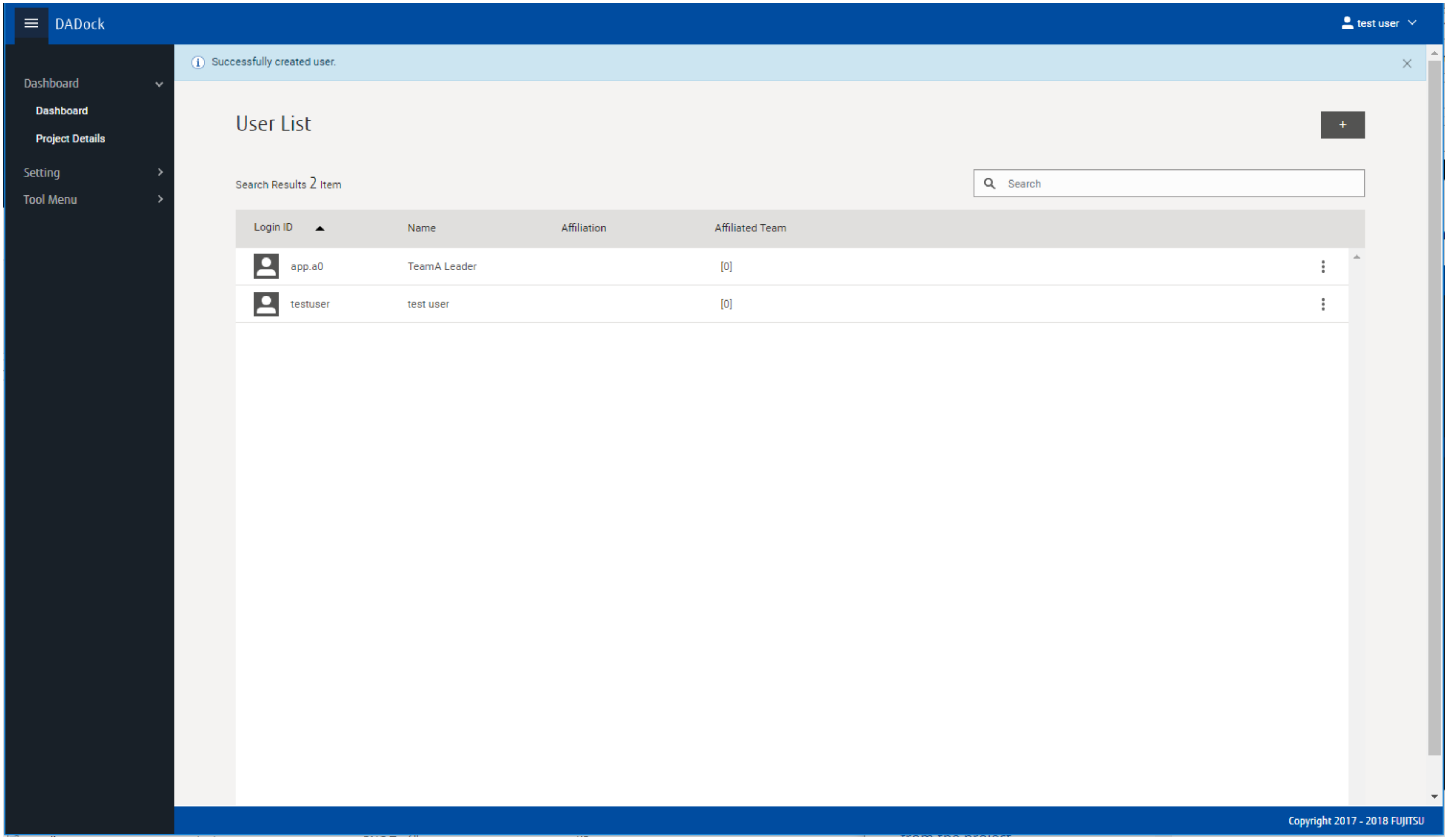
PasswordSetting requirement

- Alpha-numeric characters can be used.
- The length of characters is can be set from 8 to 32 number of characters.

Password (confirmation) * ••••••••

Displayed Language English

Check the message displayed in the pop-up page, then click the [OK] button.



Login ID	Name	Affiliation	Affiliated Team
app.a0	TeamA Leader		[0]
testuser	test user		[0]

Check that the user [app.a0] is created and displayed in User List Page, execute the same procedures to create the following 3 users.

・UserInformation of app.a1

Field	Value

Field	**Value**
Login ID	app.a1
Last Name	TeamA
First Name	Developer
Email Address	app.a1@example.com
Affiliation	<leave as blank>
Permission	Standard
Password	password
Password (confirmation)	password
Displayed Language	English

・UserInformation of app.b0

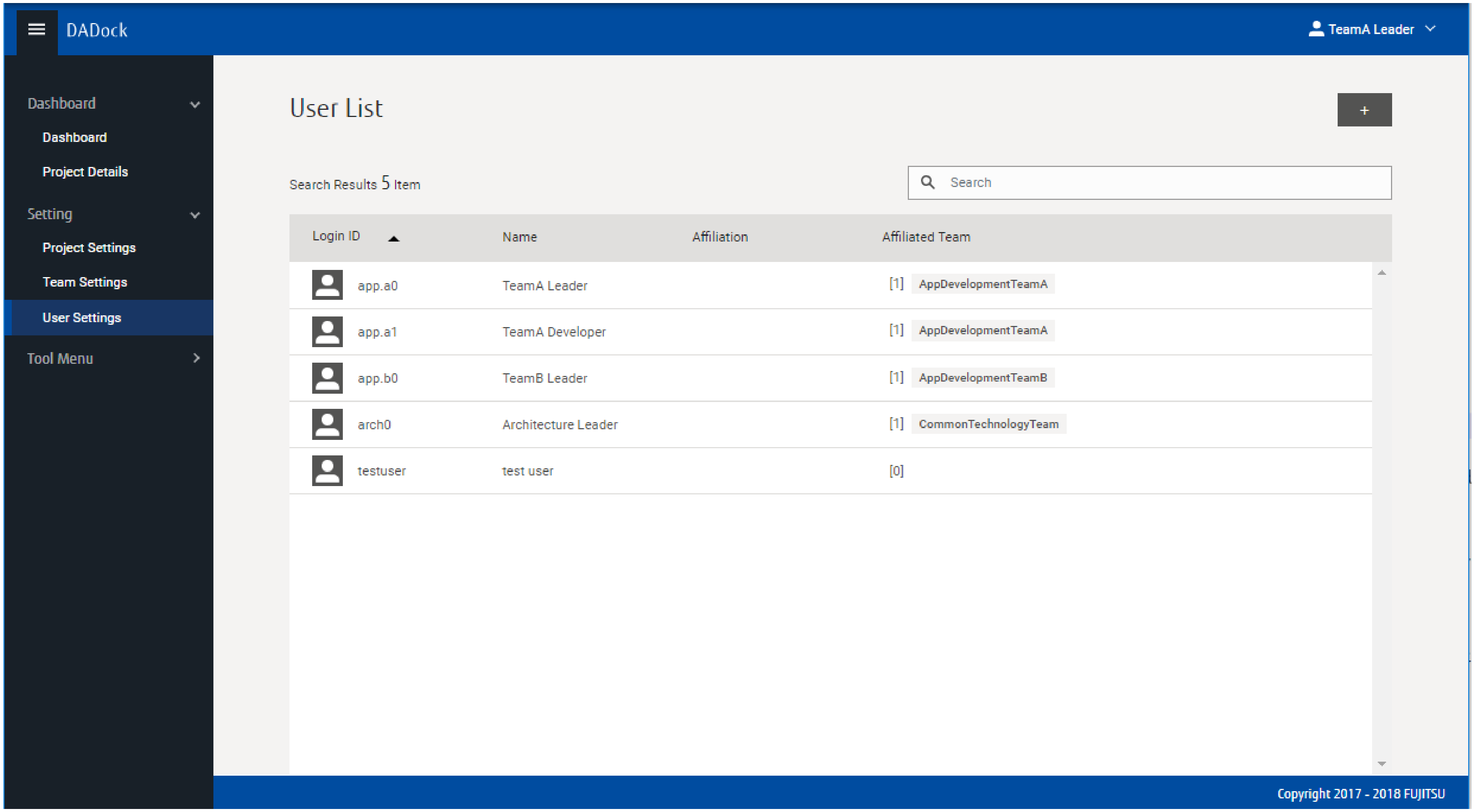
Field	**Value**
Login ID	app.b0
Last Name	TeamB
First Name	Leader

Field	**Value**
Email Address	app.b0@example.com
Affiliation	<leave as blank>
Permission	Leader
Password	password
Password (confirmation)	password
Displayed Language	English

・UserInformation of arch0

Field	**Value**
Login ID	arch0
Last Name	Architecture
First Name	Leader
Email Address	arch0@example.com
Affiliation	<leave as blank>
Permission	Leader

Field	Value
Password	password
Password (confirmation)	password
Displayed Language	English



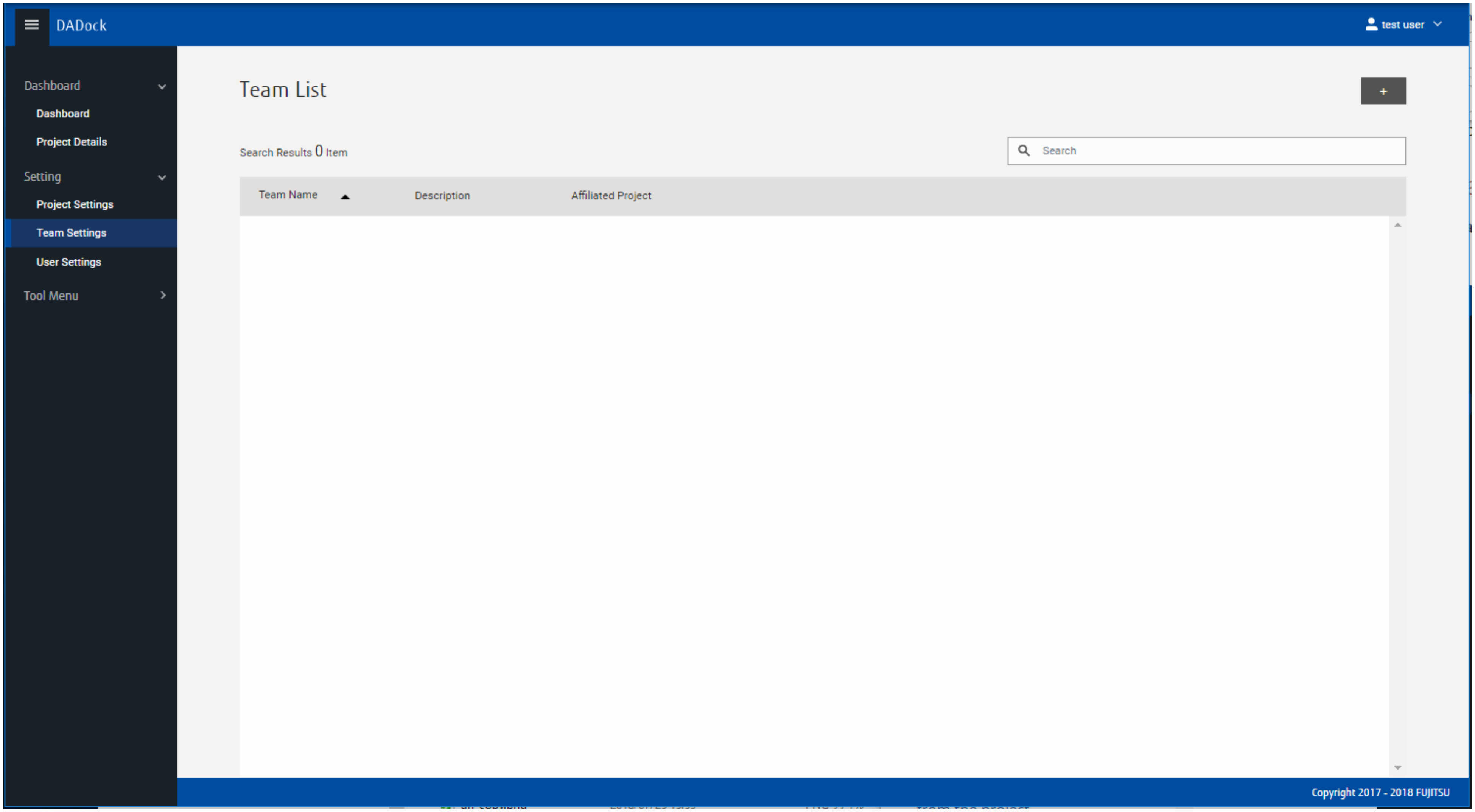
Search Results 5 Item

Login ID	Name	Affiliation	Affiliated Team
app.a0	TeamA Leader		[1] AppDevelopmentTeamA
app.a1	TeamA Developer		[1] AppDevelopmentTeamA
app.b0	TeamB Leader		[1] AppDevelopmentTeamB
arch0	Architecture Leader		[1] CommonTechnologyTeam
testuser	test user		[0]

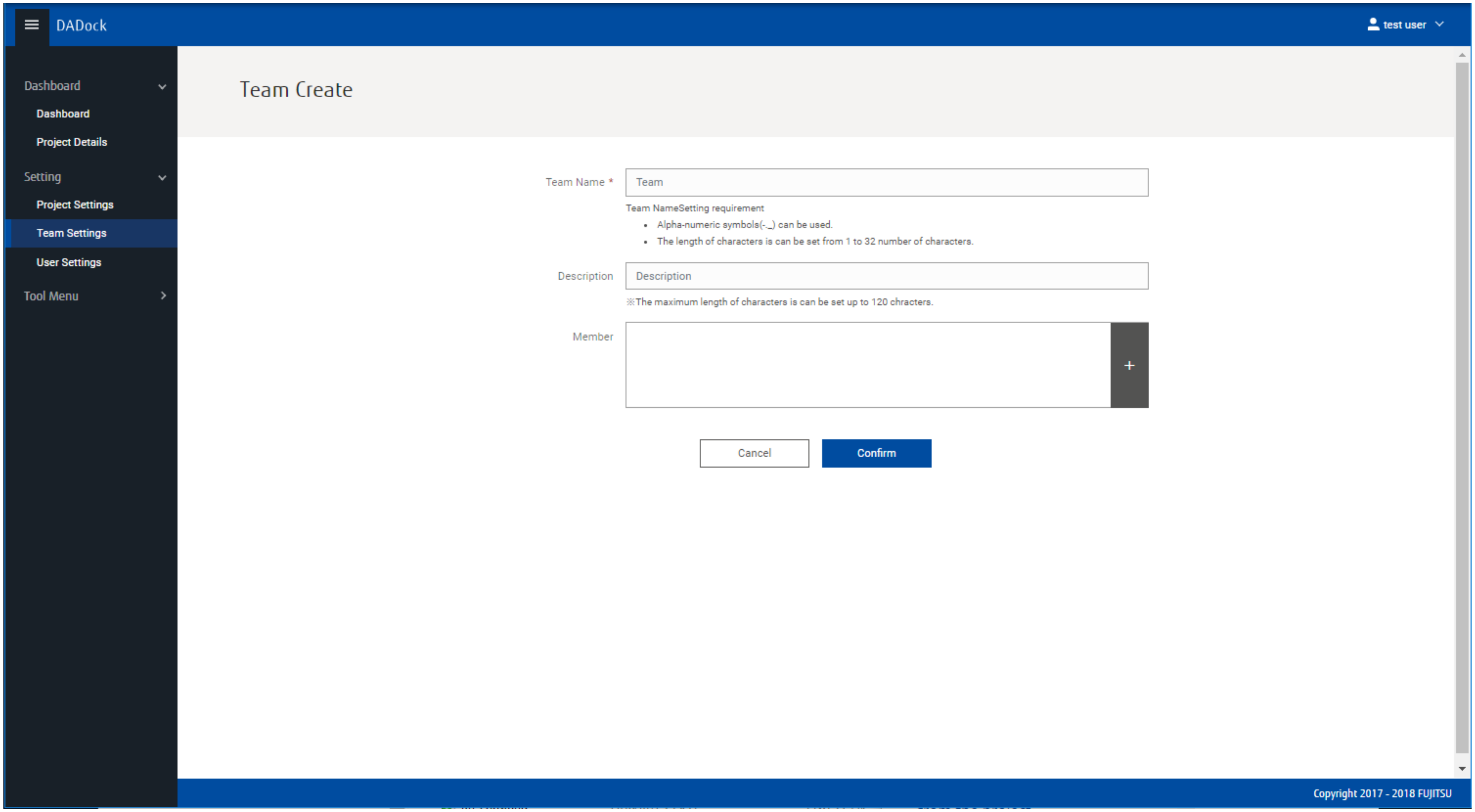
With these procedures, the newly created 4 users can now login and use DADock Portal.

3.2. Create new team

From the Dashboard Page of DADock Portal, expand the [Settings] tab then click [Team Settings] to display the Team List Page.



At the upper right part of the Team List Page, click [＋] button and the Team Create Page will be displayed.



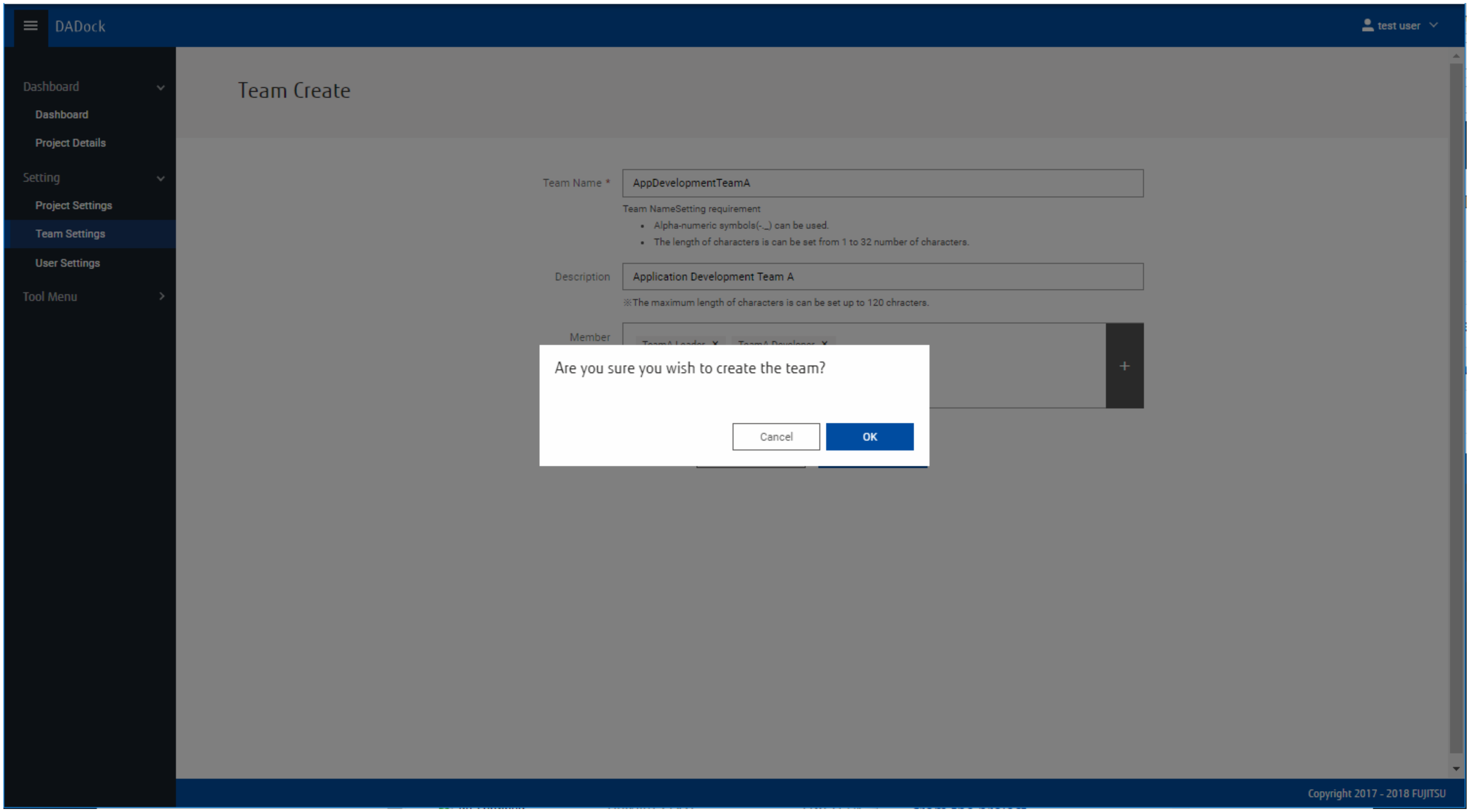
Team NameSetting requirement
- Alpha-numeric symbols(-._) can be used.
- The length of characters is can be set from 1 to 32 number of characters.

※The maximum length of characters is can be set up to 120 chracters.

Input the following team information of the Application Development Team A and click the [Confirm] button to save the entry.

Field	Value
Team Name	AppDevelopmentTeamA

Field	Value
Description	Application Development Team A
Member	TeamA Leader、 TeamA Developer



Team Name * AppDevelopmentTeamA

Team NameSetting requirement
- Alpha-numeric symbols(-._) can be used.
- The length of characters is can be set from 1 to 32 number of characters.

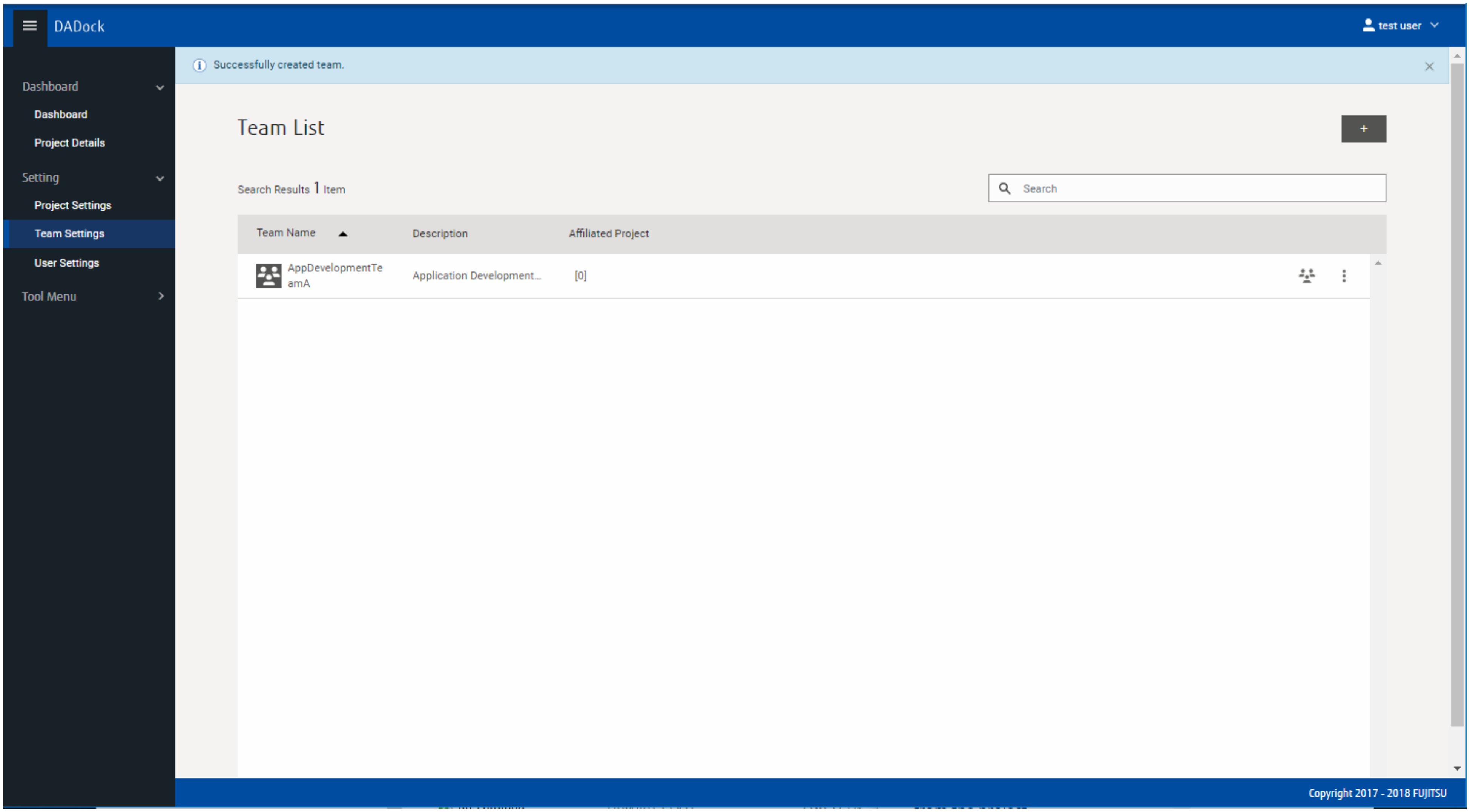
Description Application Development Team A

※The maximum length of characters is can be set up to 120 chracters.

Member

Are you sure you wish to create the team?

Check the message displayed in the pop-up page, then click the [OK] button.



Successfully created team.

Team List

Search Results 1 Item

Team Name	Description	Affiliated Project
AppDevelopmentTeamA	Application Development...	[0]

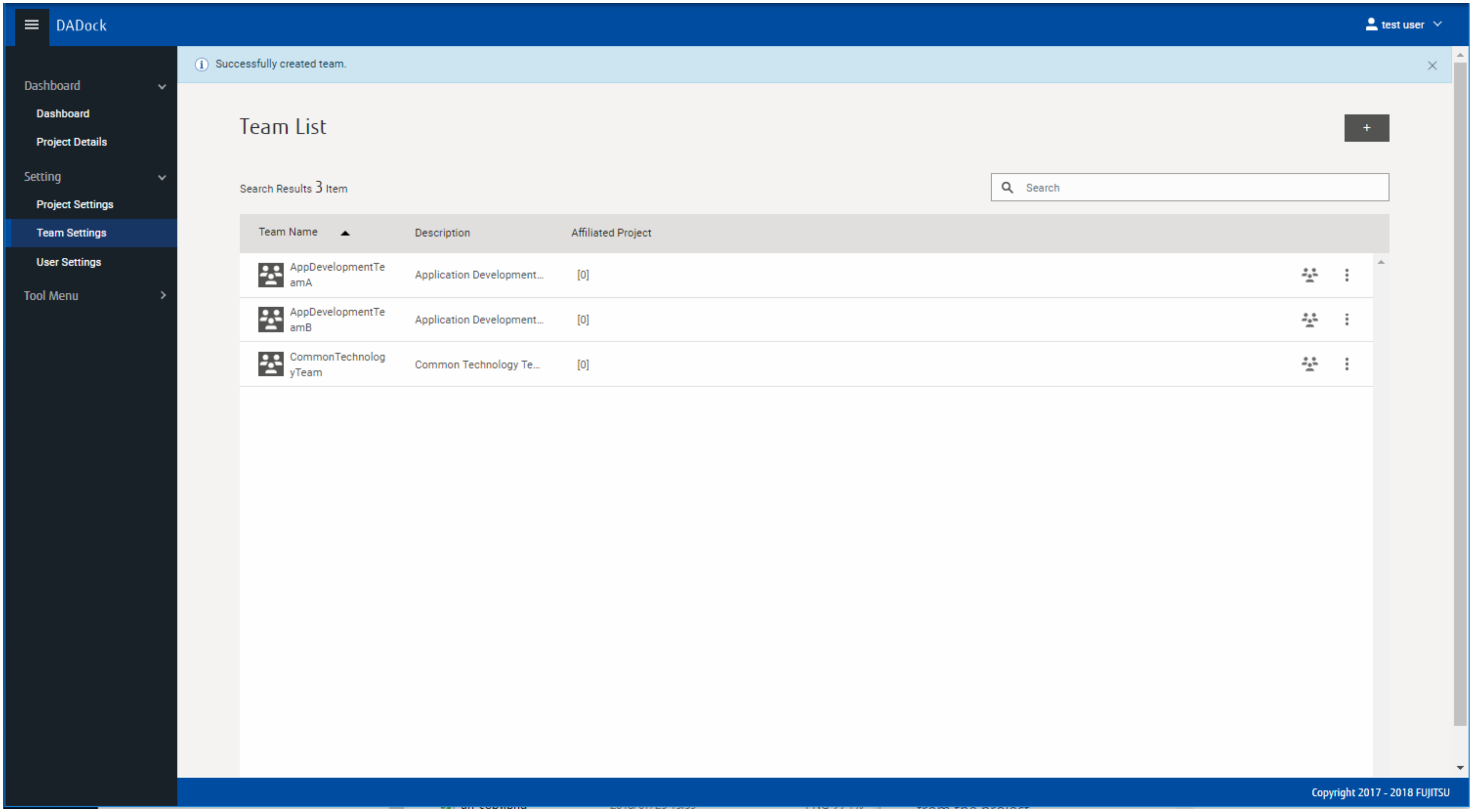
Check that the team [AppDevelopmentTeamA] is created and displayed in Team List Page, execute the same procedures to create the following 2 teams.

・TeamInformationofAppDevelopmentTeamB

Field	Value
Team Name	AppDevelopmentTeamB
Description	Application Development Team B
Member	TeamB Leader

・TeamInformationofCommonTechnologyTeam

Field	Value
Team Name	CommonTechnologyTeam
Description	Common Technology Team
Member	Architecture Leader



Successfully created team.

Team List

Search Results 3 Item

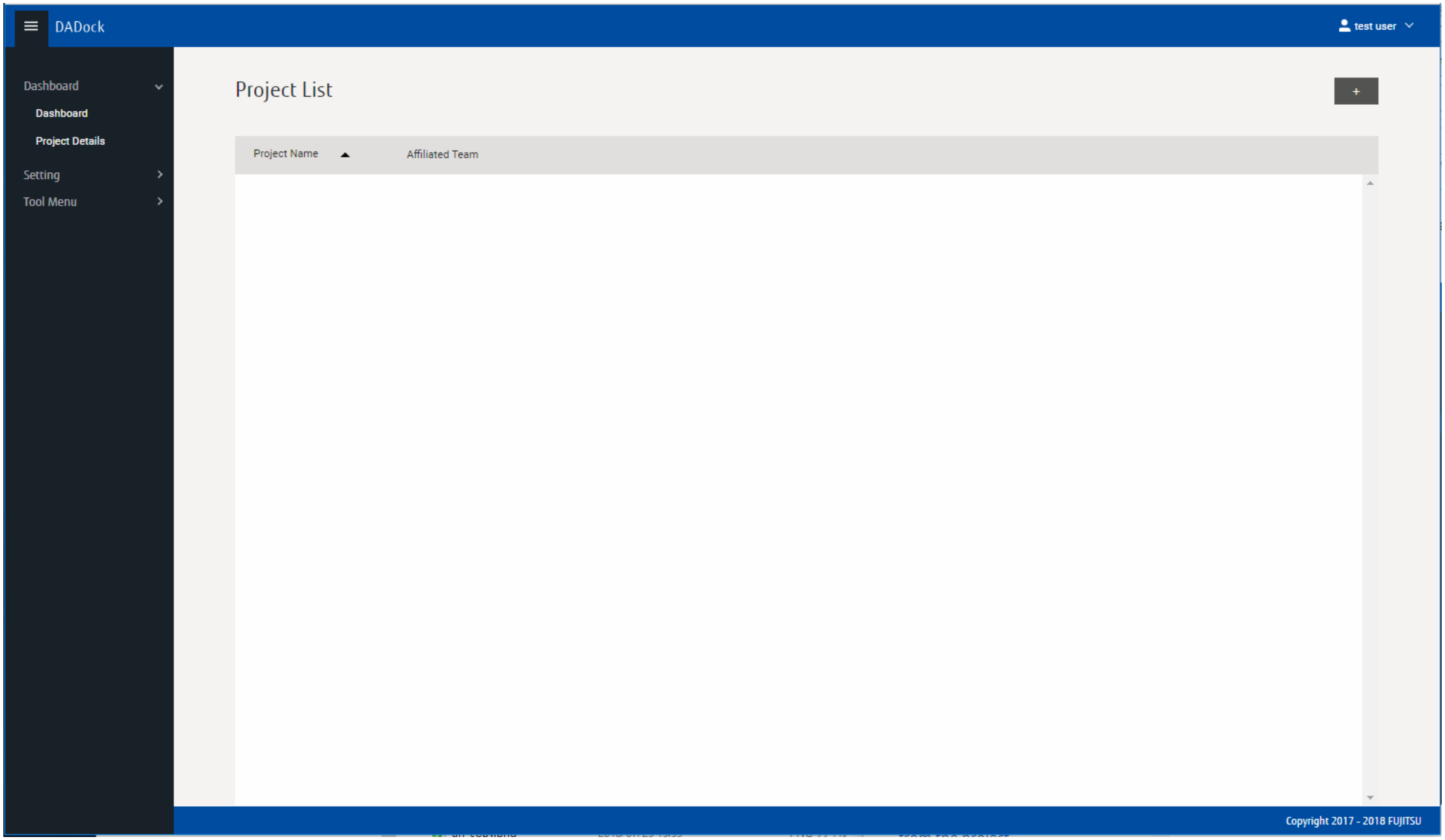
Team Name	Description	Affiliated Project
AppDevelopmentTeamA	Application Development...	[0]
AppDevelopmentTeamB	Application Development...	[0]
CommonTechnologyTeam	Common Technology Te...	[0]

3.3. Create project

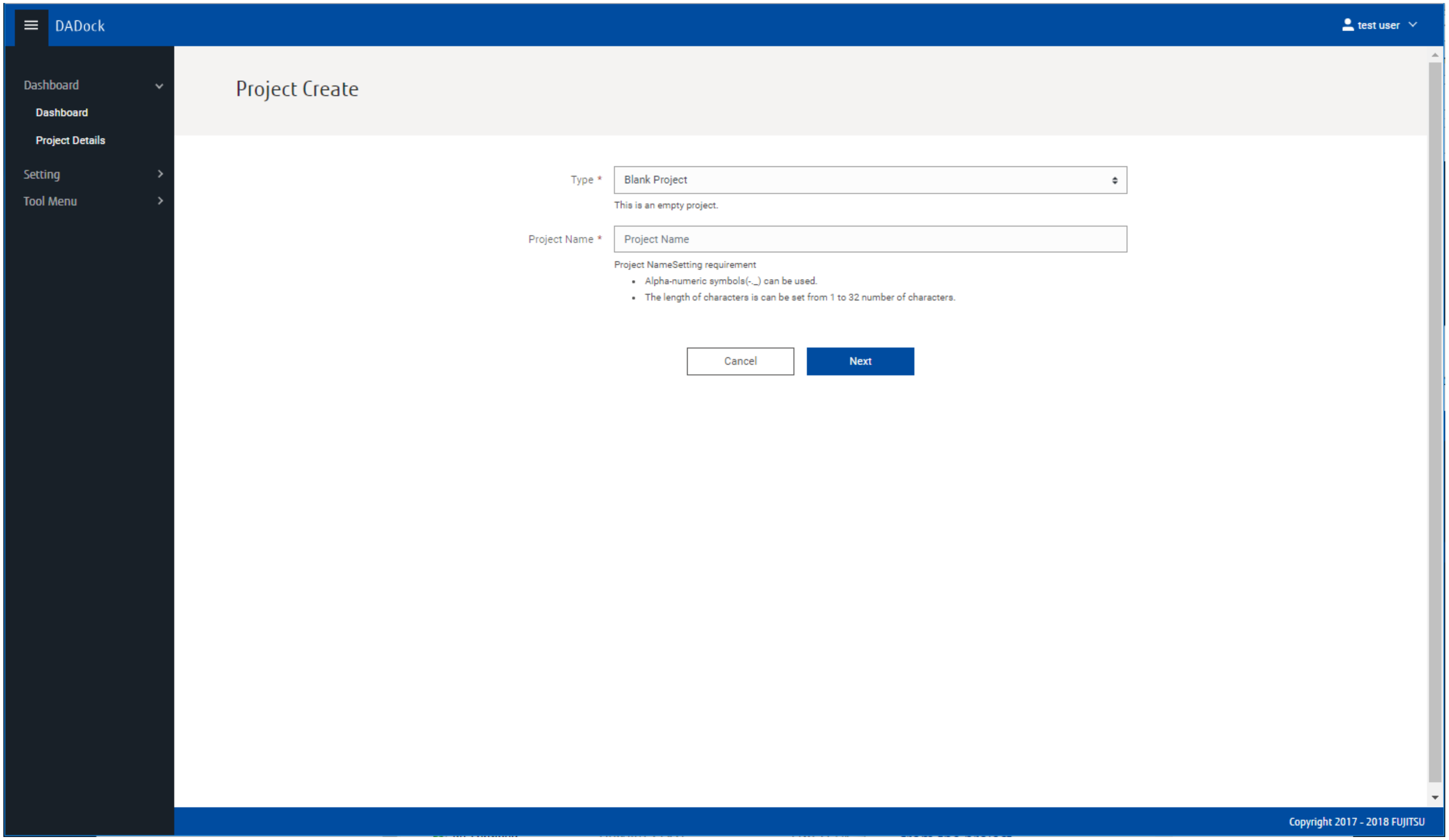
Create projects for Web Application development and common components development.

3.3.1. Create new project

From the Dashboard Page of DADock Portal, expand the [Settings] tab then click [Project Settings] to display the Project List Page.



At the upper right part of the Project List Page, click [＋] button and the Project Create Page will be displayed.



Project Create

Type * Blank Project

This is an empty project.

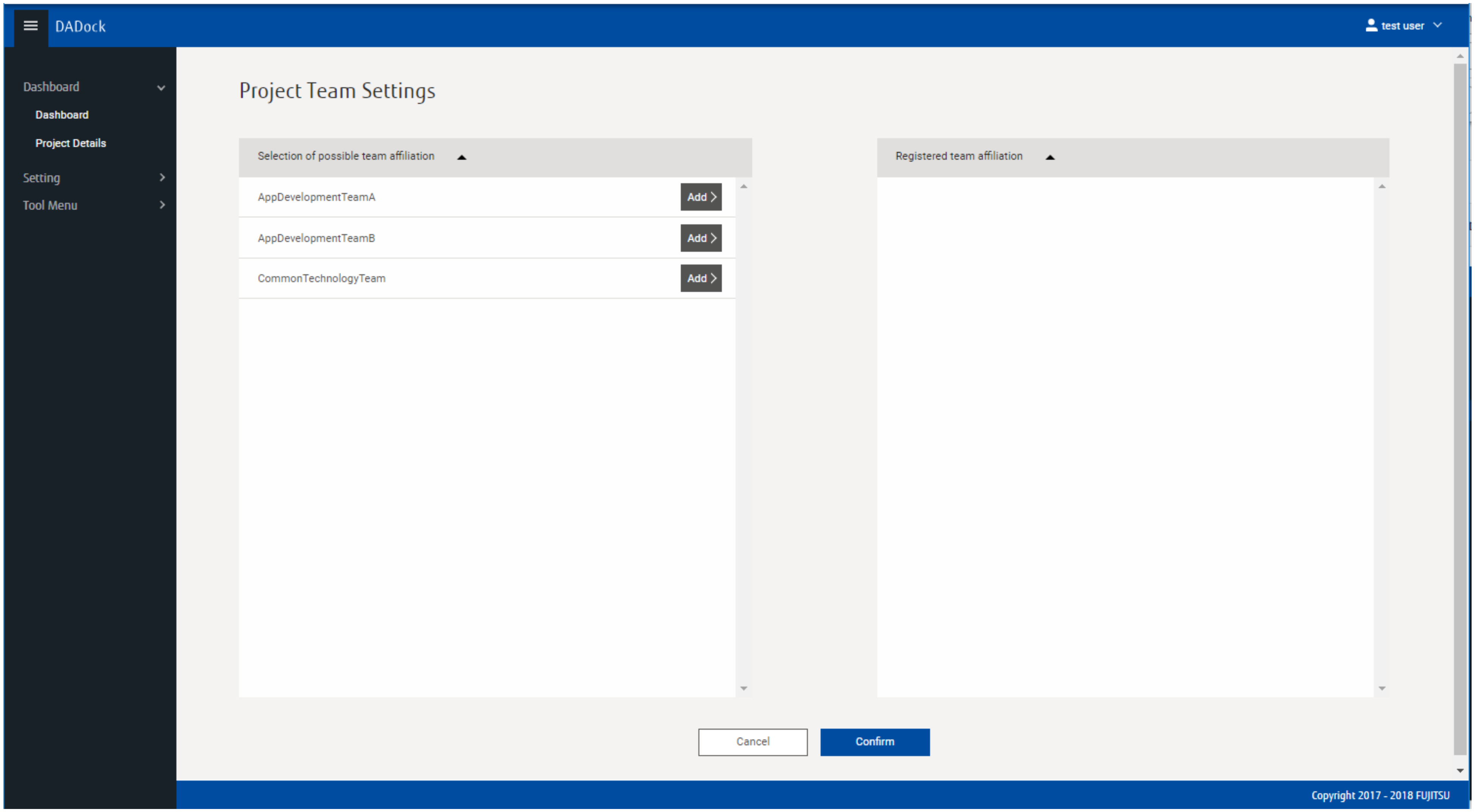
Project Name * Project Name

Project NameSetting requirement
- Alpha-numeric symbols(-._) can be used.
- The length of characters is can be set from 1 to 32 number of characters.

Input the following information to create a web project and the click [Next] button.

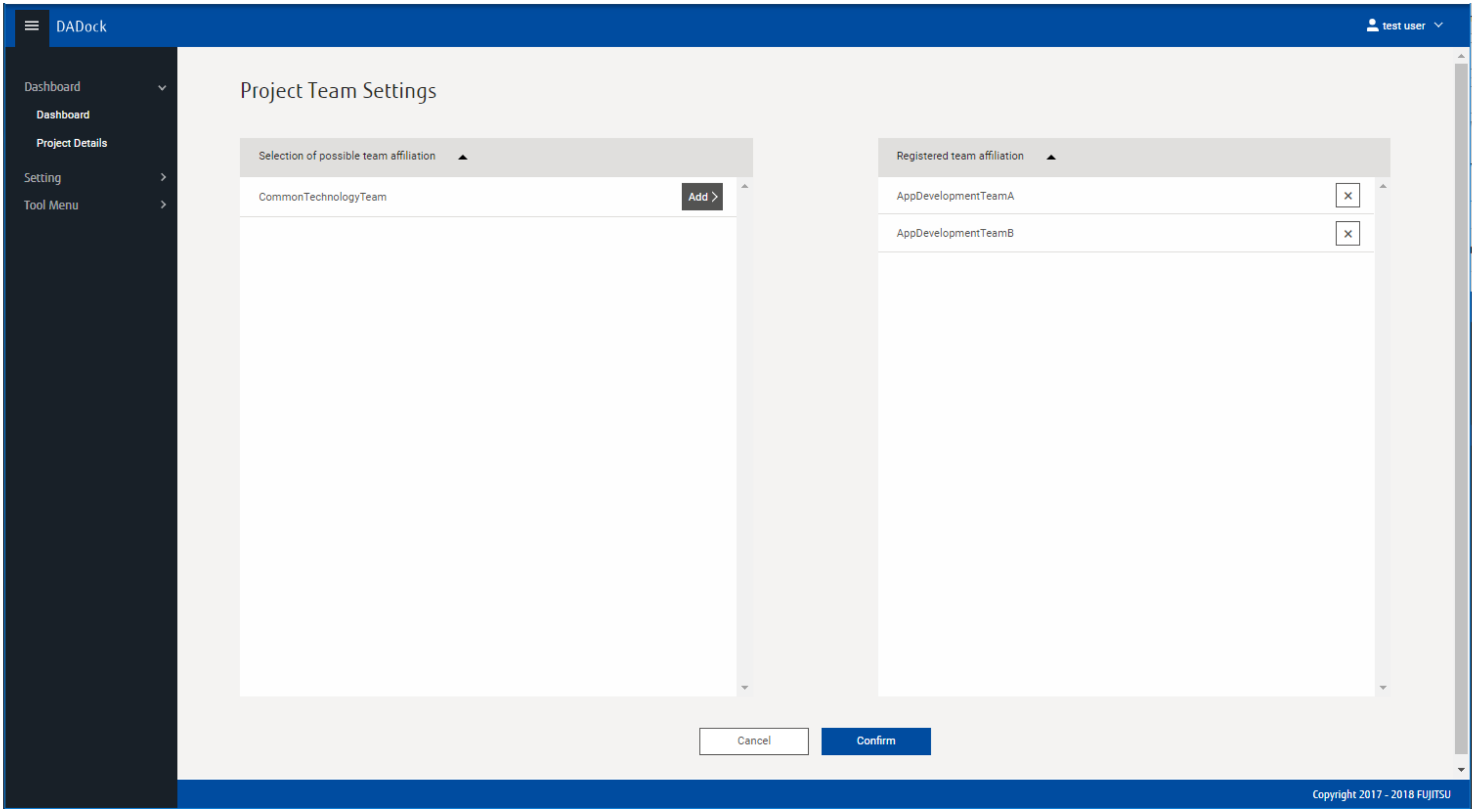
Field	Value
Type	Java Web Application

Field	Value
Project Name	TutorialWeb



The Project Team Settings will be displayed then from [Selection of possible team affiliation] list, click the [Add>] button of the following teams.

TeamName	**AppDevelopmentTeamA、AppDevelopmentTeamB**

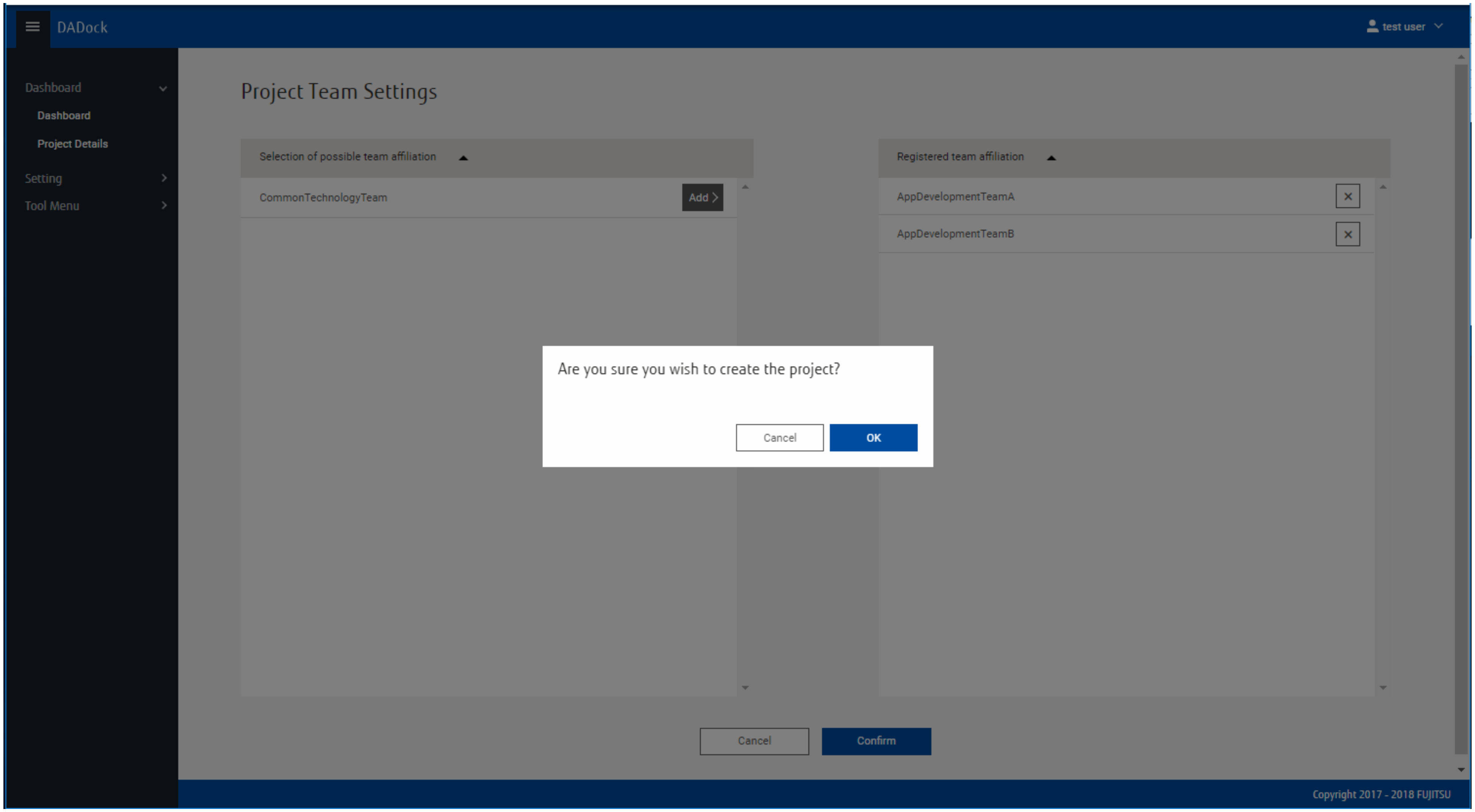


Project Team Settings

Selection of possible team affiliation
CommonTechnologyTeam

Registered team affiliation
AppDevelopmentTeamA
AppDevelopmentTeamB

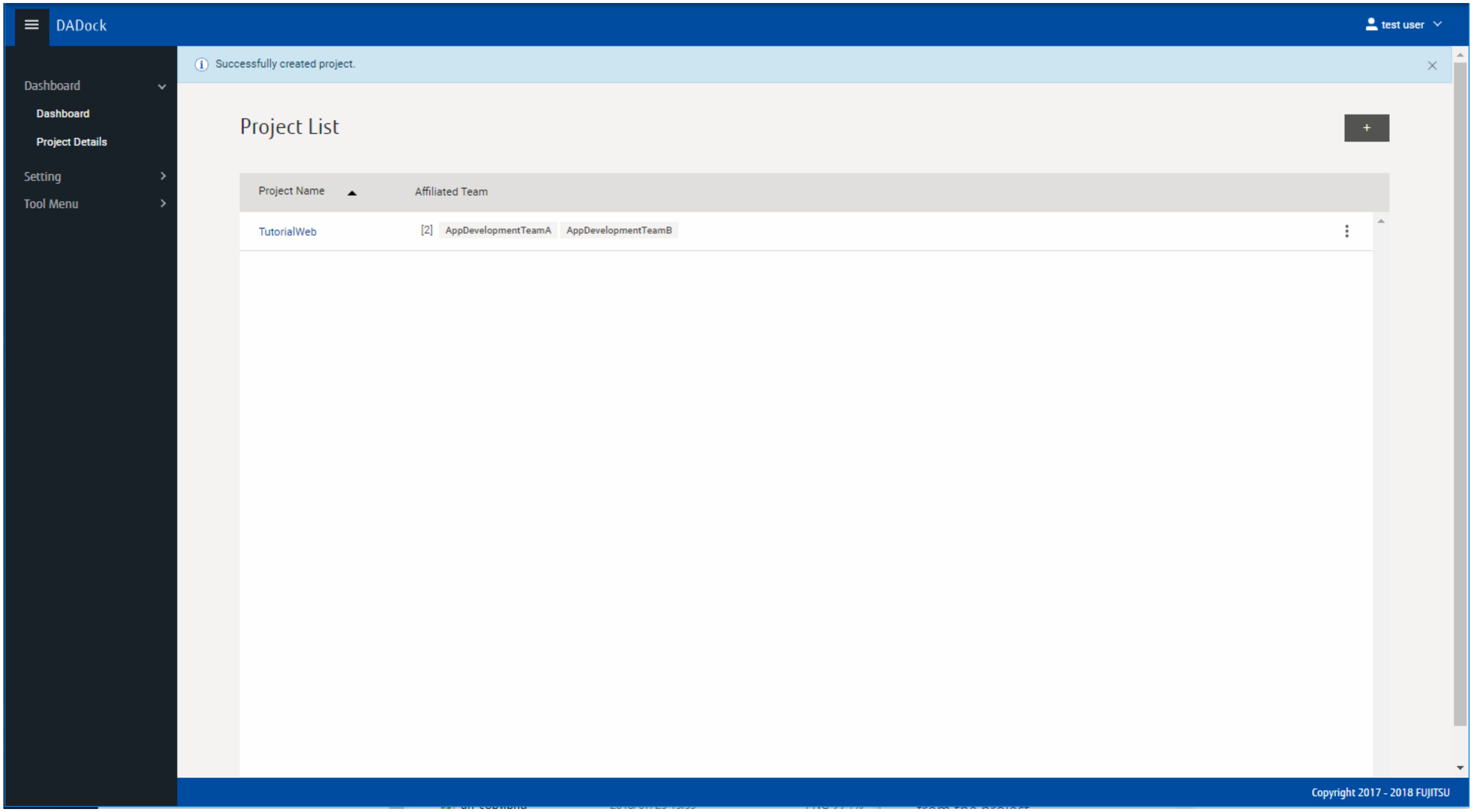
Check if the team is moved to the list of [Registered team affiliation], then click the [Confirm] button.



Are you sure you wish to create the project?

Cancel OK

Check the message displayed in the pop-up page, then click the [OK] button.



Successfully created project.

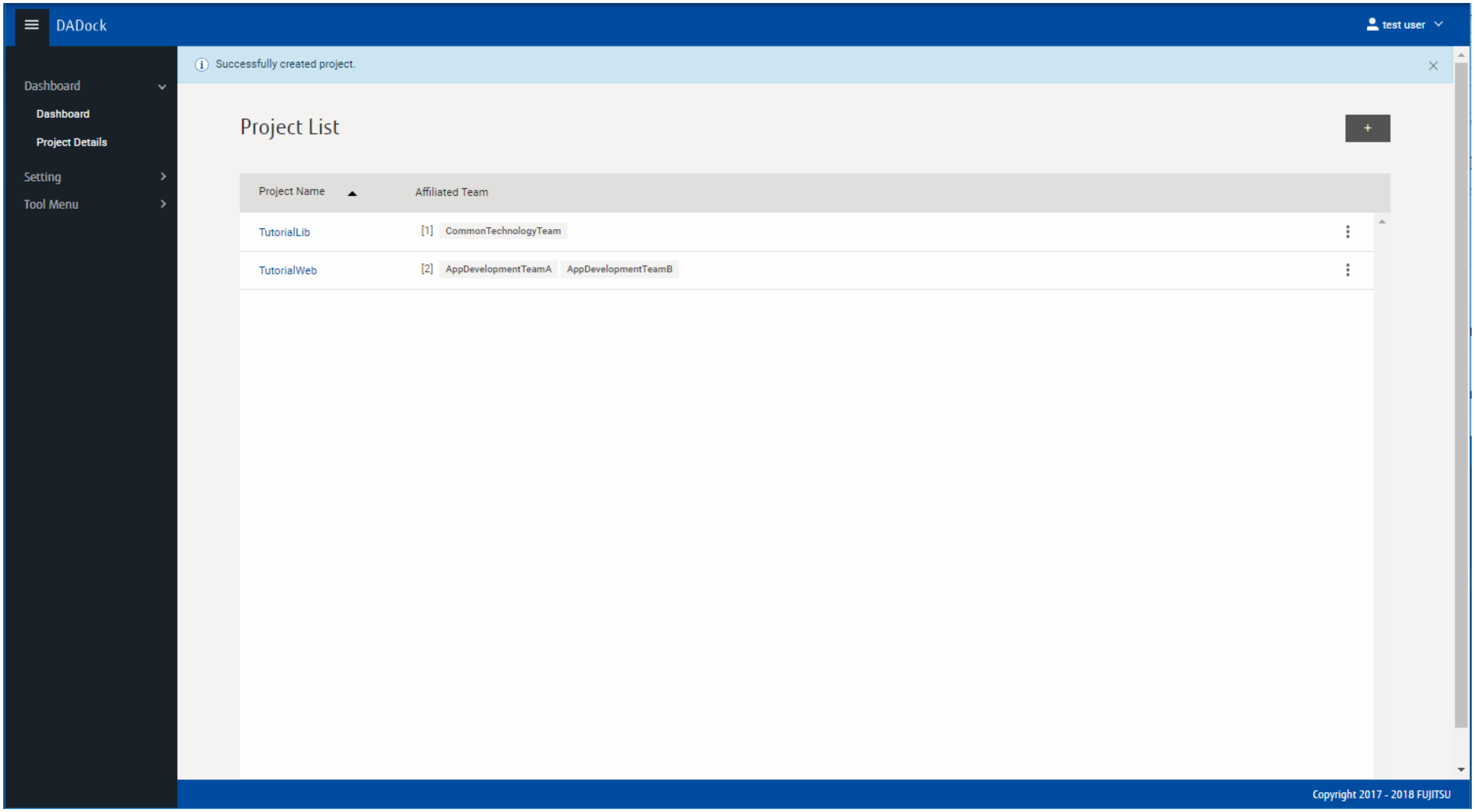
Project List

Project Name	Affiliated Team
TutorialWeb	[2] AppDevelopmentTeamA AppDevelopmentTeamB

Check that the project [TutorialWeb] is created and displayed in Project List Page, execute the same procedures to create project for common components.

Field	Value
Type	Java Library

Field	Value
Project Name	TutorialLib
Team Name	**CommonTechnologyTeam**



Successfully created project.

Project List

Project Name	Affiliated Team
TutorialLib	[1] CommonTechnologyTeam
TutorialWeb	[2] AppDevelopmentTeamA AppDevelopmentTeamB

With these procedures, 2 projects are now created.

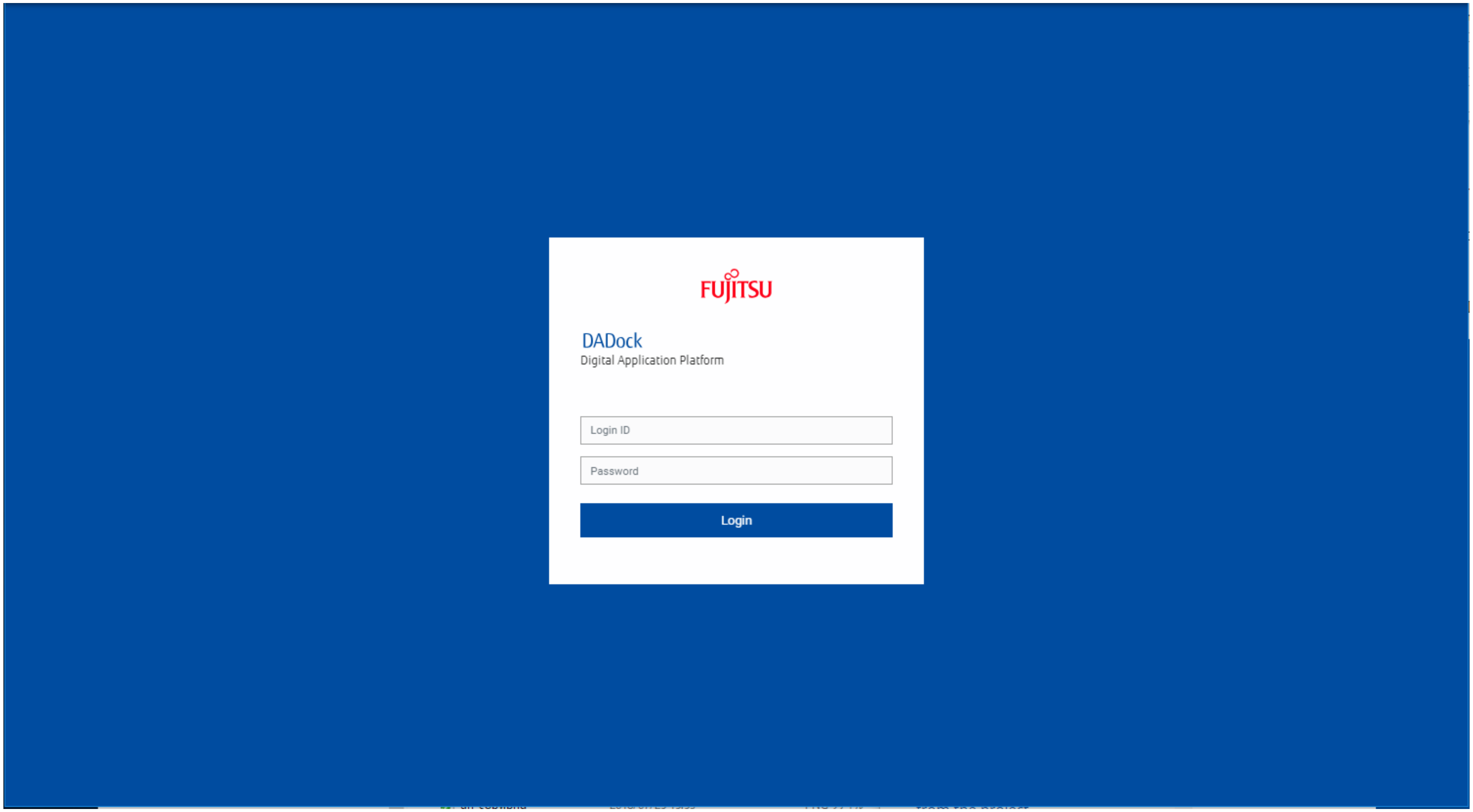
4. Common components development

This chapter explains the flow from development of common components to release of the project by common technical team.

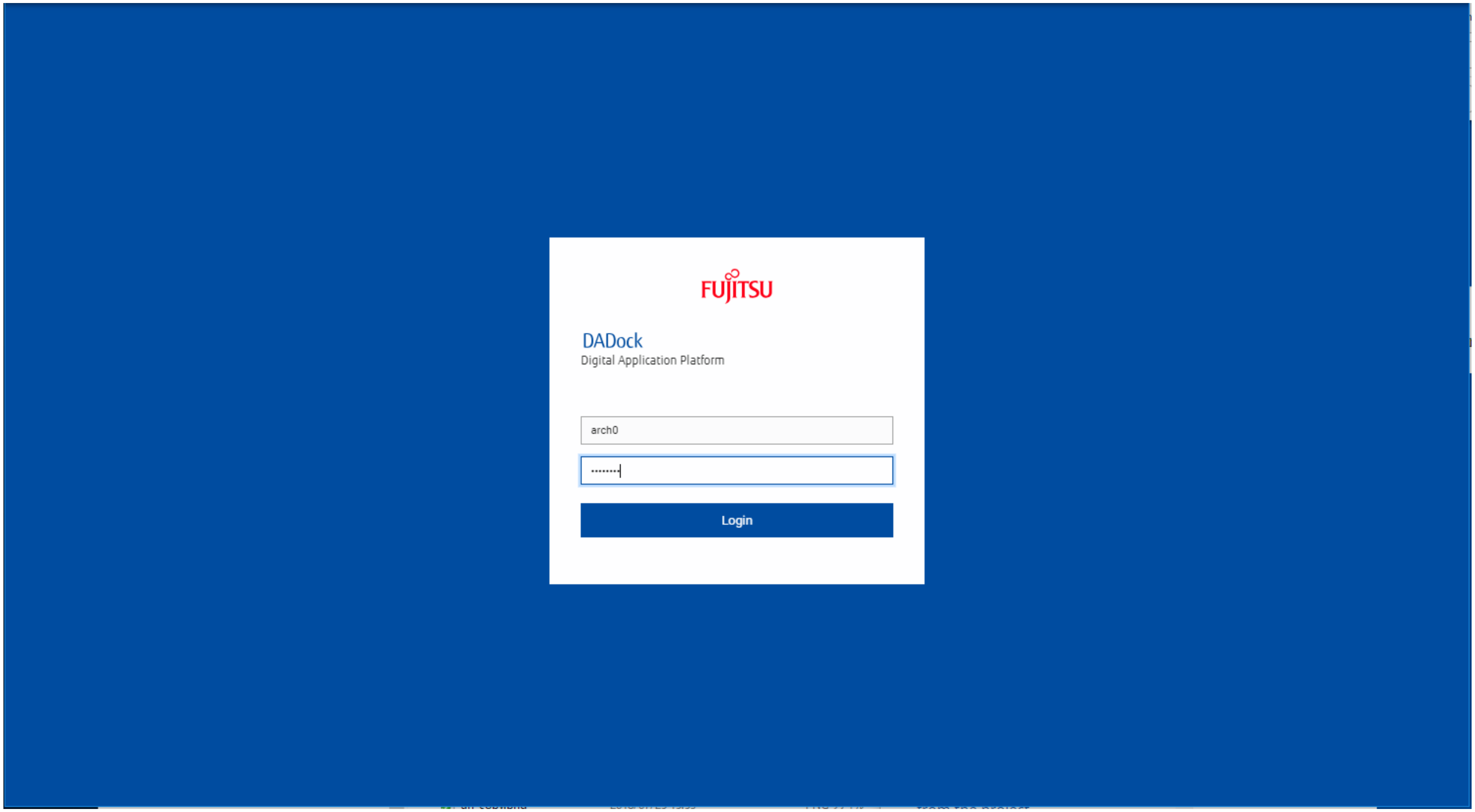
4.1. Clone the development assets from the project

In this section, clone the development assets from the Git repository and download the development assets in the client machine.

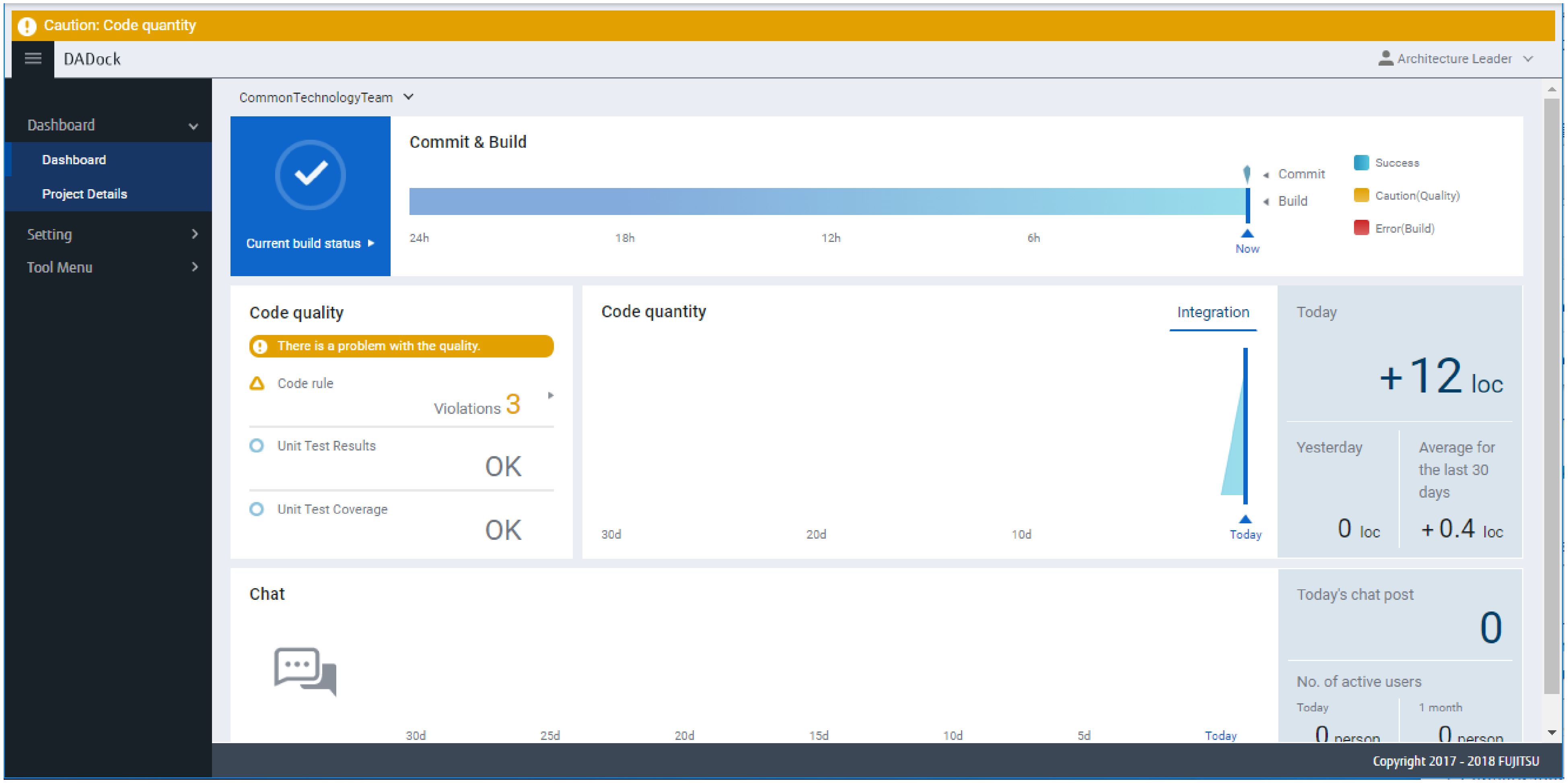
1.Access DADock Portal and the Login Page will be displayed.



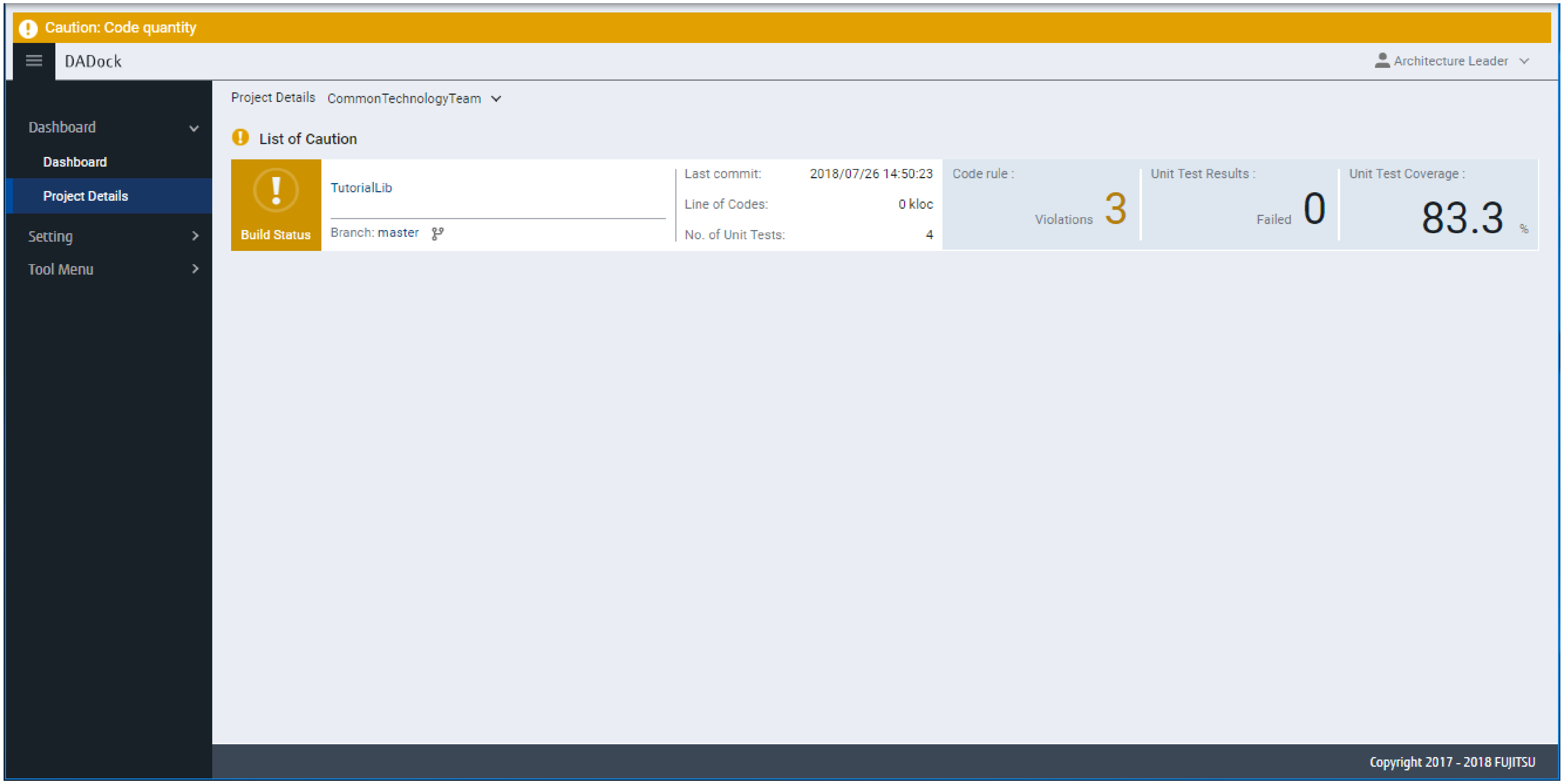
2.Login as arch0 (common technical team leader).



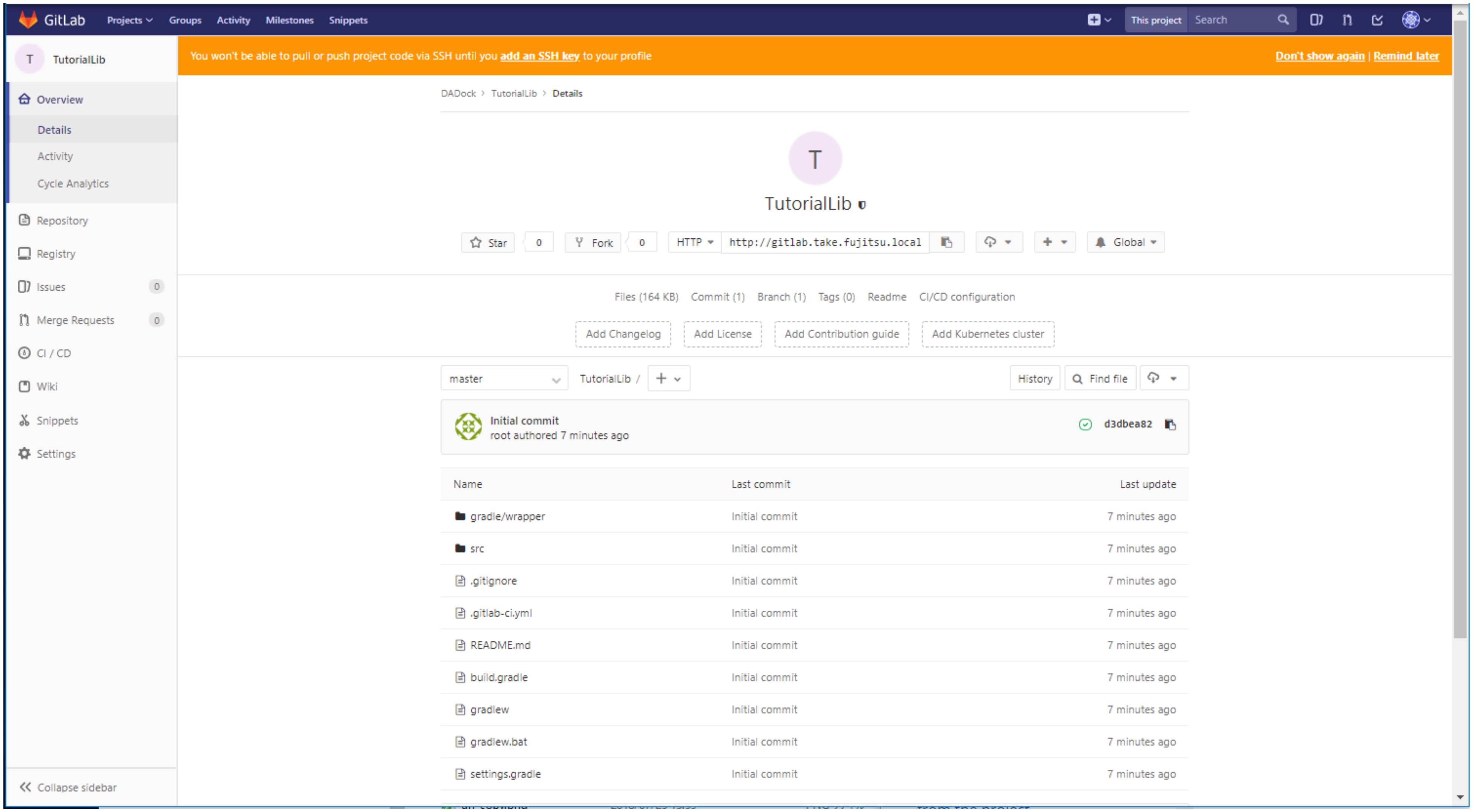
3.Check that the quality status information is displayed in the Dashboard Page and then click [Project Details] from the Menu Bar.



4.Check that the detailed quality information is displayed in the Project Details Page then, click the project name link of TutorialLib project.



5.In the new tab of the browser, the TutorialLib project from GitLab page is displayed, copy the displayed URL located at the right side of the [HTTP] field.



You won't be able to pull or push project code via SSH until you add an SSH key to your profile

DADock › TutorialLib › Details

TutorialLib

http://gitlab.take.fujitsu.local

Files (164 KB) Commit (1) Branch (1) Tags (0) Readme CI/CD configuration

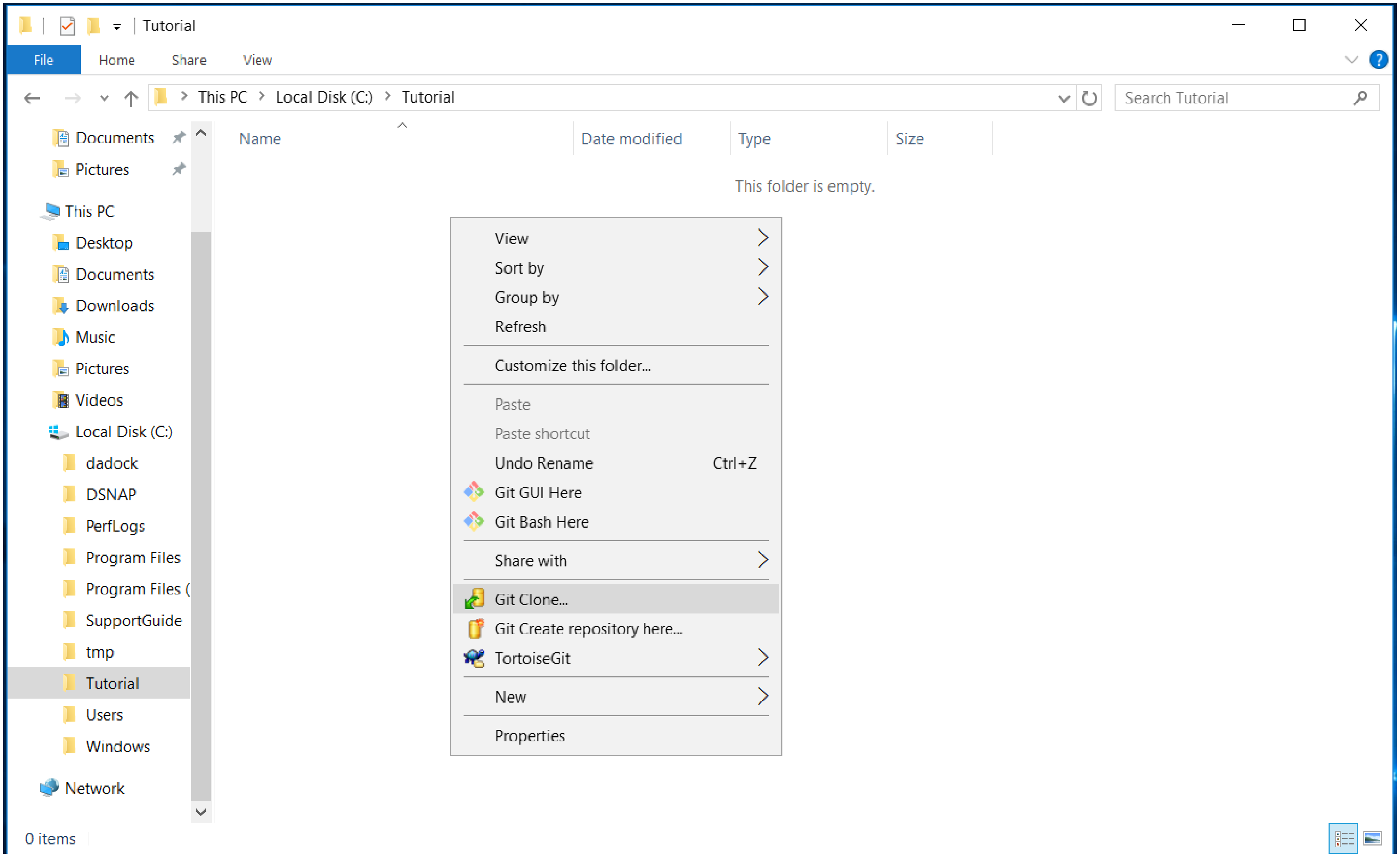
TutorialLib /

Initial commit
root authored 7 minutes ago
d3dbea82

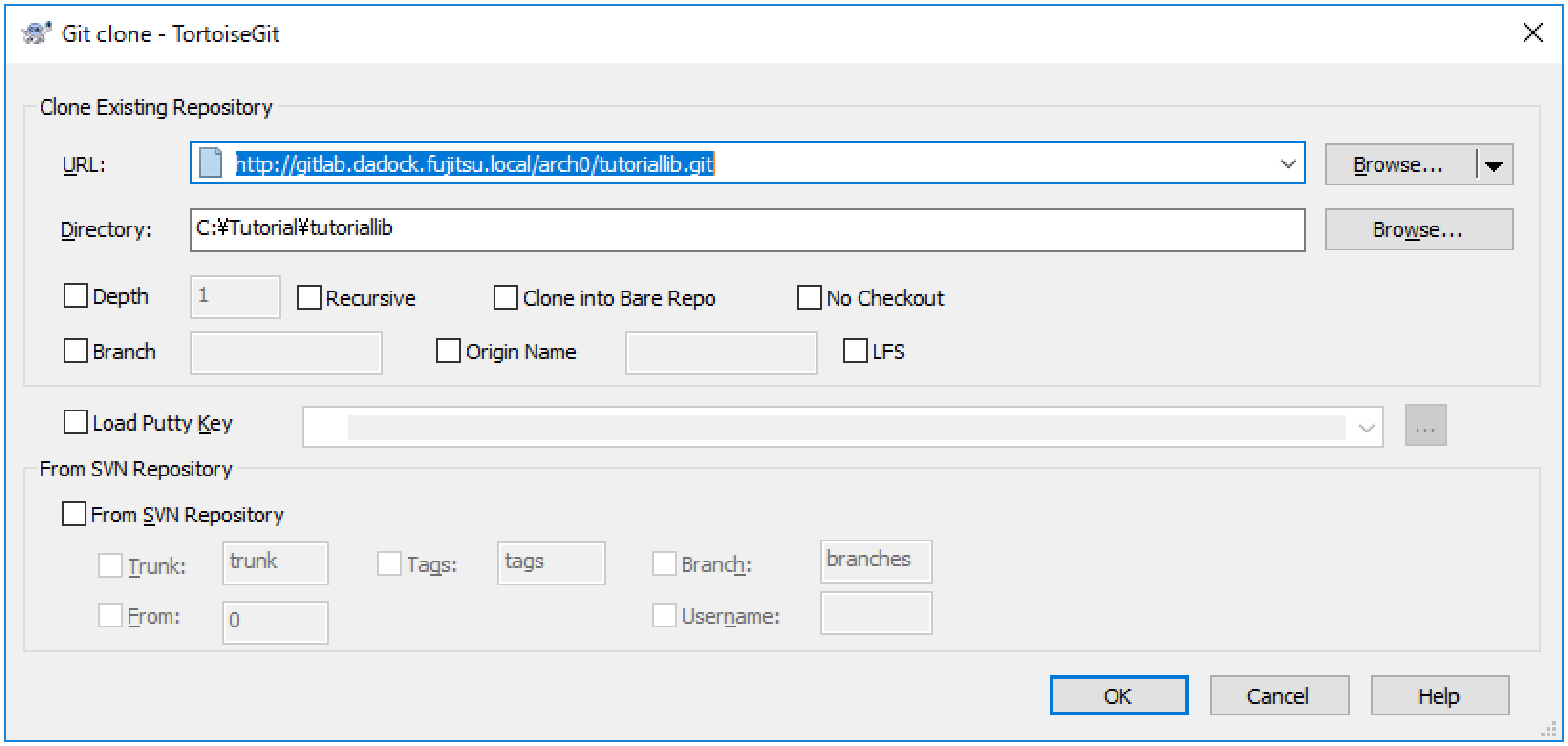
Name	Last commit	Last update
gradle/wrapper	Initial commit	7 minutes ago
src	Initial commit	7 minutes ago
.gitignore	Initial commit	7 minutes ago
.gitlab-ci.yml	Initial commit	7 minutes ago
README.md	Initial commit	7 minutes ago
build.gradle	Initial commit	7 minutes ago
gradlew	Initial commit	7 minutes ago
gradlew.bat	Initial commit	7 minutes ago
settings.gradle	Initial commit	7 minutes ago

6.In the client machine, create a directory naming 'C:\Tutorial'.

7.In the created folder, right click and select [Git Clone] from [TortoiseGit] context Menu.



8.In the Git Clone Page of TortoiseGit, input the information of the GitLab repository and click the [OK] button.



URL: http://gitlab.dadock.fujitsu.local/arch0/tutoriallib.git

Directory: C:¥Tutorial¥tutoriallib

Depth: 1

Trunk: trunk

Tags: tags

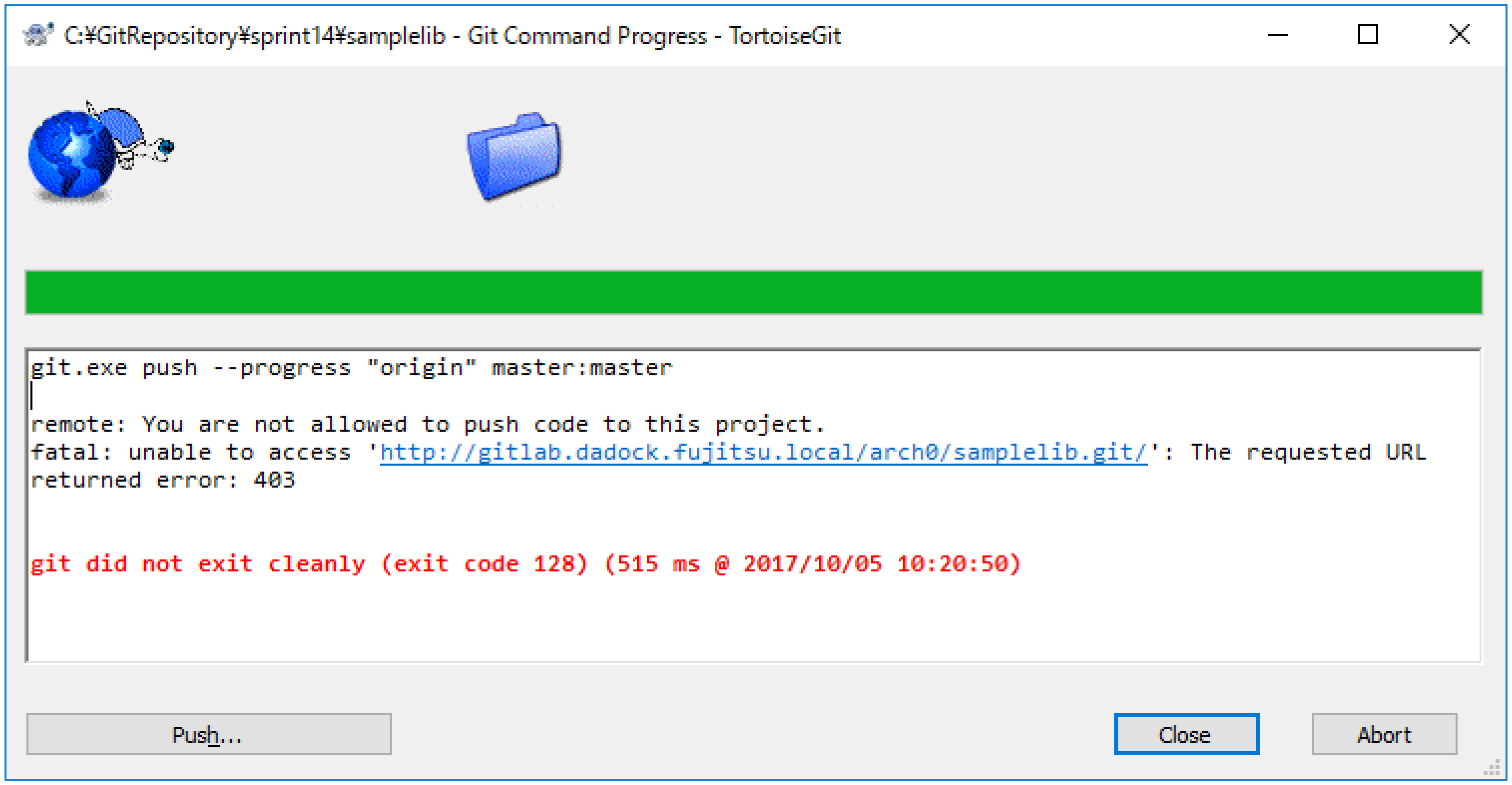
Branch: branches

From: 0

Item Field	**Value**
URL:	The value copied in the clipboard (leave it as it is since this is the default value)
Directory:	C:\Tutorial\tutoriallib (leave it as it is since this is the default value)

In case authentication with GitLab fails

If in case the authentication information with GitLab is saved in the used client machine, there is a chance that the following error will appear and authentication with GitLab will fail. From the Control Panel, open the Credential Manager and delete the authentication information related to the defined URL.

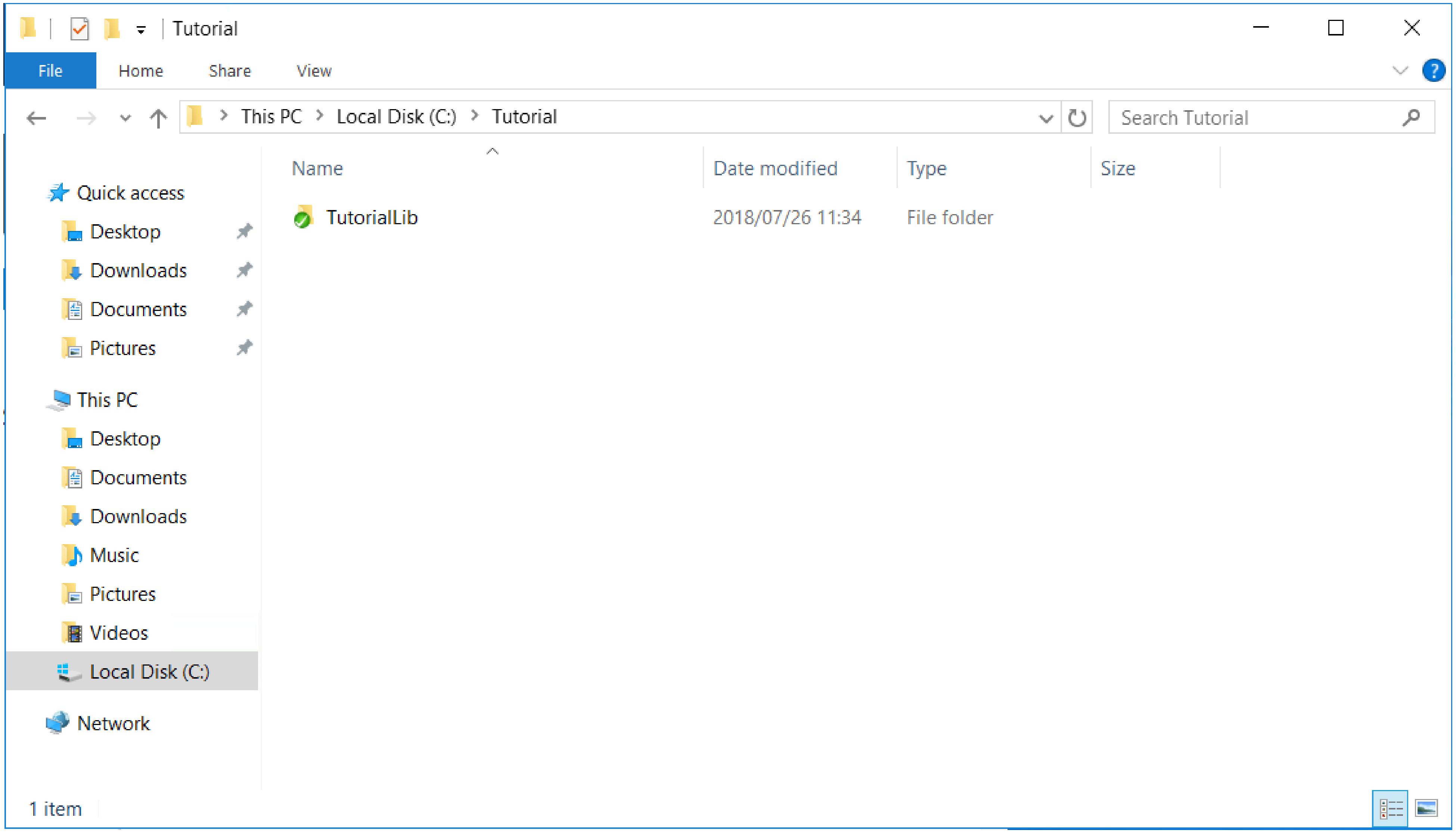

C:¥GitRepository¥sprint14¥samplelib - Git Command Progress - TortoiseGit
Push...
Close
Abort

```
git.exe push --progress "origin" master:master

remote: You are not allowed to push code to this project.
fatal: unable to access 'http://gitlab.dadock.fujitsu.local/arch0/samplelib.git/': The requested URL
returned error: 403


git did not exit cleanly (exit code 128) (515 ms @ 2017/10/05 10:20:50)
```

Inputting Information Credentials

When the username and password are required from TortoiseGit, use the credentials of [arch0] as an input information.

9.After the clone process, the folder 'C:\Tutorial\tutoriallib' will be created and the development assets is downloaded from the repository.

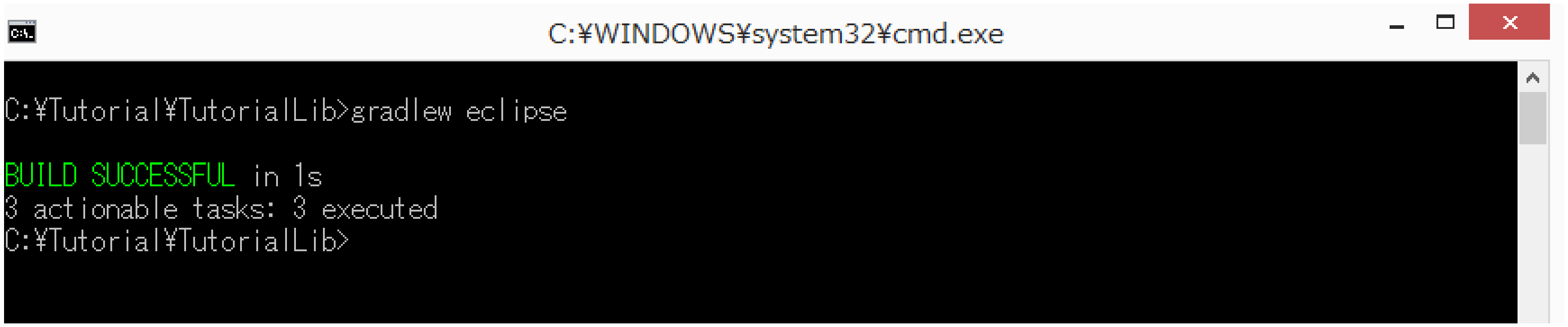


Name	Date modified	Type	Size
TutorialLib	2018/07/26 11:34	File folder	

4.2. Import the development assets in Eclipse

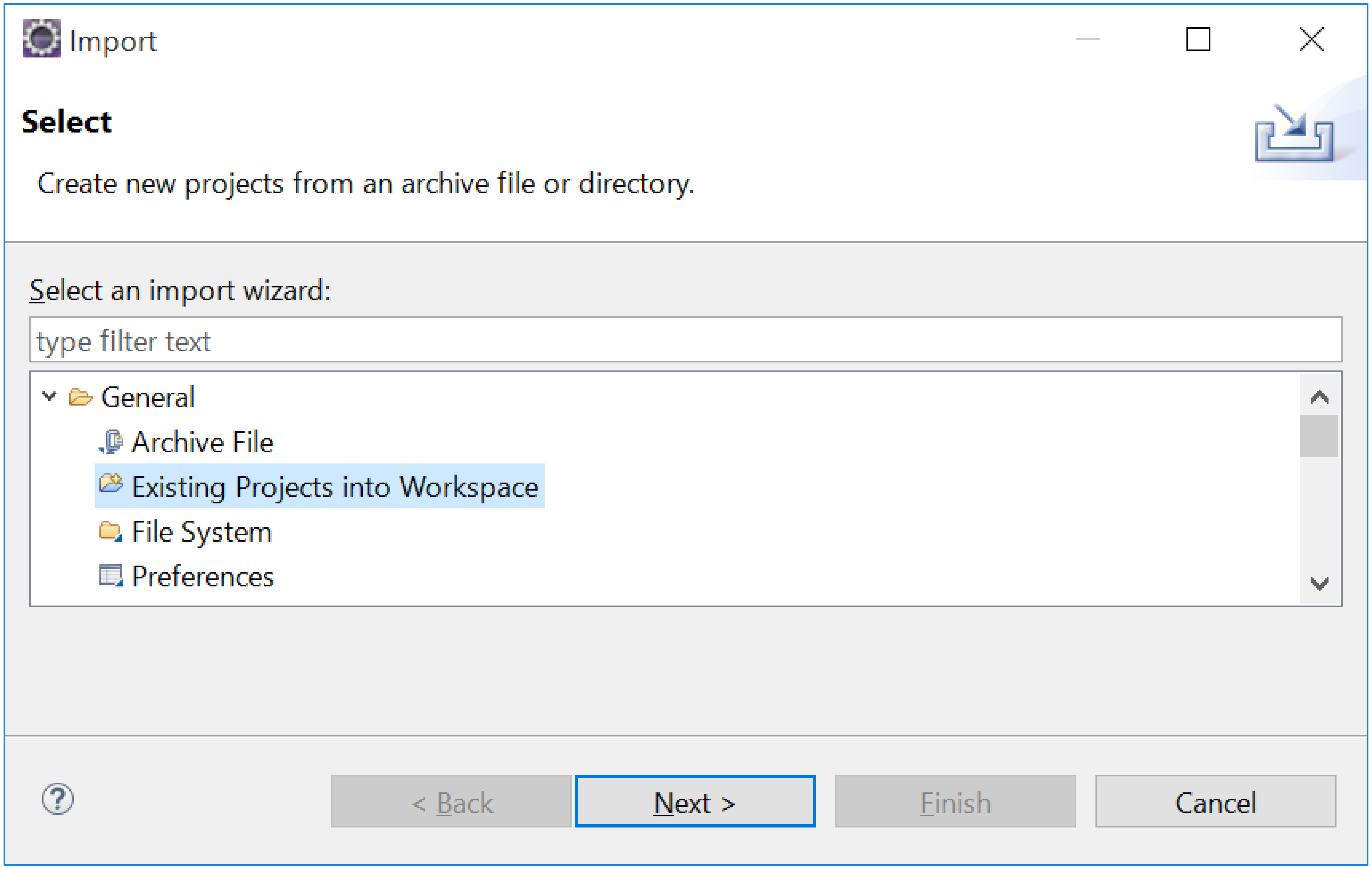
In this section, the development assets are imported in Eclipse so that development can now be done in Eclipse. Execute beforehand the procedure in Proxy settings for the various tools.

1.Start up the command prompt and move the current directory to 'C:\Tutorial\tutoriallib', then execute the command 'gradlew eclipse'.

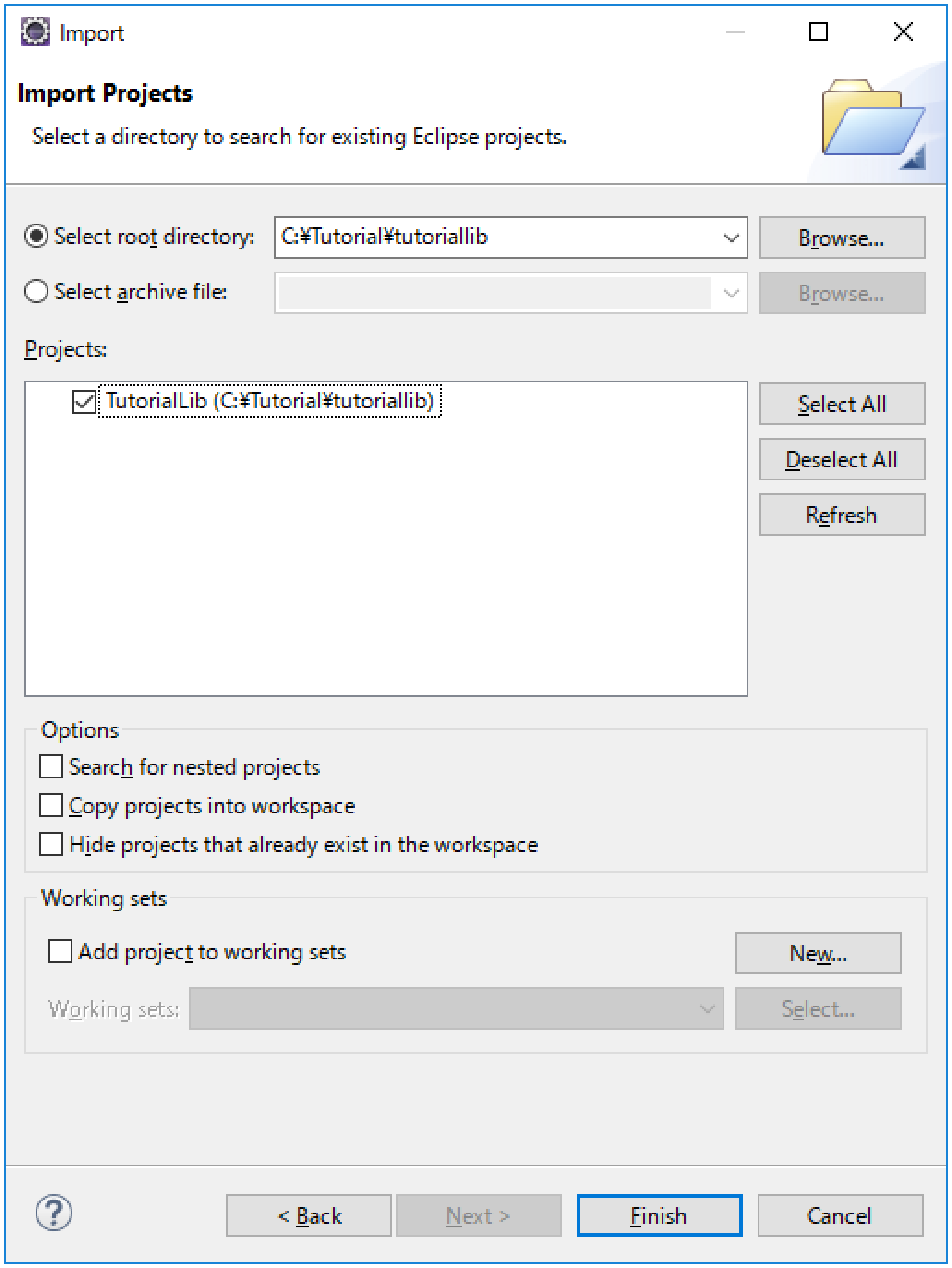


```
C:¥Tutorial¥TutorialLib>gradlew eclipse

BUILD SUCCESSFUL in 1s
3 actionable tasks: 3 executed
C:¥Tutorial¥TutorialLib>
```

2.Start up Eclipse and from the File menu, click [Import…], select [General] → [Existing Projects into Workspace] and click [Next >].

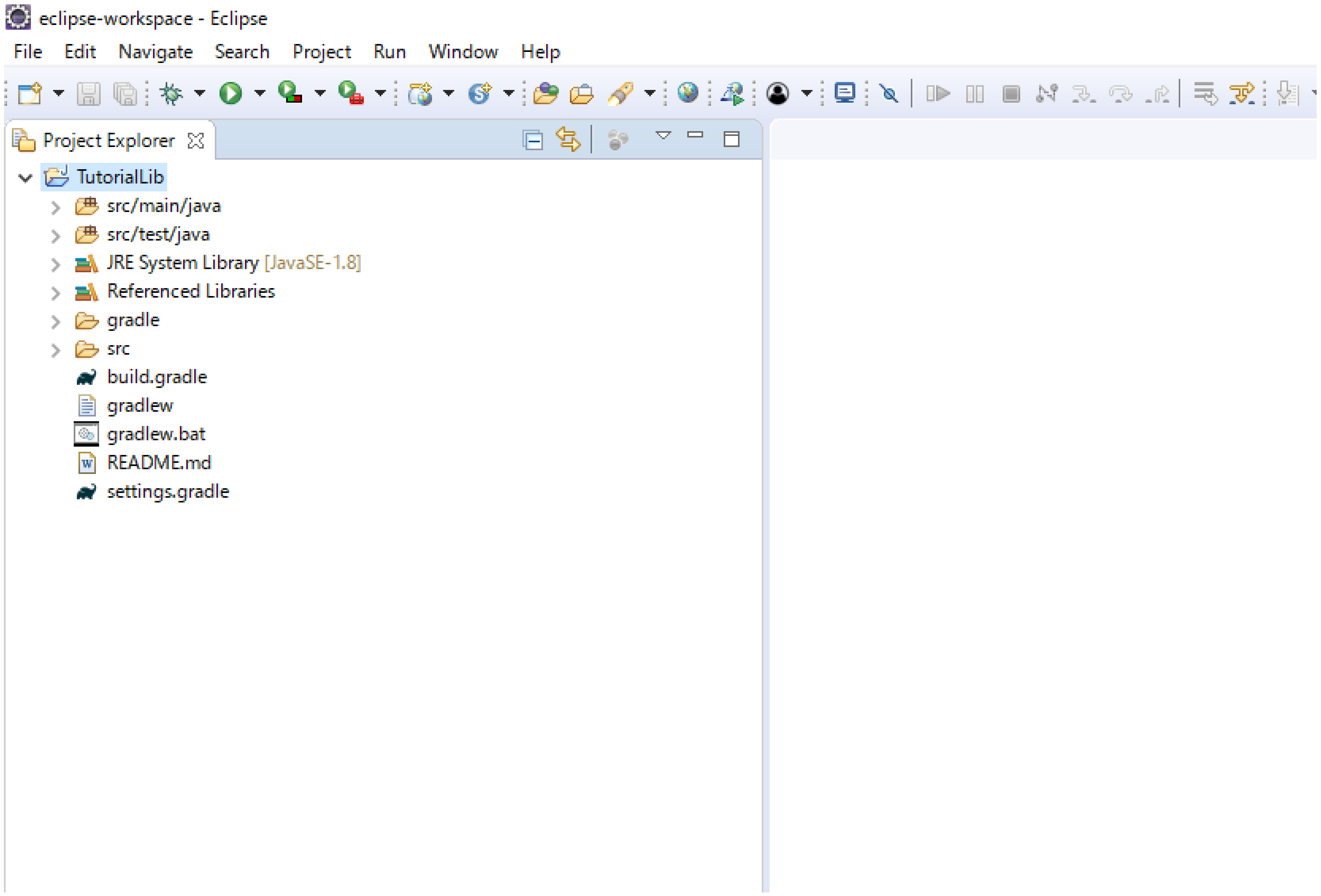


3.Select C:\Tutorial\tutoriallib and click [Finish] button.


Import
Import Projects
Select a directory to search for existing Eclipse projects.
Select root directory: C:¥Tutorial¥tutoriallib
Browse...
Select archive file:
Browse...
Projects:
TutorialLib (C:¥Tutorial¥tutoriallib)
Select All
Deselect All
Refresh
Options
Search for nested projects
Copy projects into workspace
Hide projects that already exist in the workspace
Working sets
Add project to working sets
New...
Working sets:
Select...
< Back
Next >
Finish
Cancel

Item Field	Value
Select root directory:	C:\Tutorial\tutoriallib
Projects:	*When the value in the root directory is inputted, this is automatically inputted so leave this field as it is.

4.The project will then be displayed in Eclipse.


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer

TutorialLib
- src/main/java
- src/test/java
- JRE System Library [JavaSE-1.8]
- Referenced Libraries
- gradle
- src
- build.gradle
- gradlew
- gradlew.bat
- README.md
- settings.gradle

4.3. Modification of source Code

In this section, modifications of the automatically generated source code will be done and to reflect all the changes made in the repository.

1.From the project explorer of Eclipse, open the following 2 files.

- StringUtil.java, located under the directory src/main/java, which is included in project package
- StringUtilTest.java, located under the directory src/test/java, which is included in project package

2.In the file StringUtil.java, make the following modifications.

- Change the class name to CheckUtil.
- Implement the private constructor in CheckUtil. (dealing with the Code Smells shown in SonarQube)
- Implement the method isNumber() to check if the inputted string represents a number.
- Edit the implementation of isNullOrEmpty method so that the [return true] code inside the method will only appear once. (dealing with the Code Smells shown in SonarQube)

After the modifications of the source code, the code will be as follows:

list 3. CheckUtil.java

JAVA

```java
package project;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

/**
 * This is a utility class that performs string operations.
 */
public final class CheckUtil {

    /**
     * This class is a utility class and can not be instantiated.
     */
    private CheckUtil() {
    }

    /**
     * check whether arguments is null or empty.
     *
     * @param target
     *            target string
     * @return whether arguments is null or empty
     */
    public static boolean isNullOrEmpty(String target) {
        if (target == null || target.trim().isEmpty()) {
            return true;
        } else {
            return false;
        }
    }

    /**
     * check whether arguments is integer value.
     *
     * @param target
     *            target string
     * @return whether arguments is integer value
     */
    public static boolean isNumber(String target) {
        String regex = "^-?[0-9]*.?[0-9]+$";
        Pattern p = Pattern.compile(regex);
        Matcher m = p.matcher(target);
        return m.find();
    }
}
```

3.In the StringUtilTest.java file, make the following changes.

- Change the class name to CheckUtilTest.
- Add the following test codes for isNumber() method.

After the modifications in the source code, the code will be as follows:

list 4. CheckUtilTest.java

JAVA

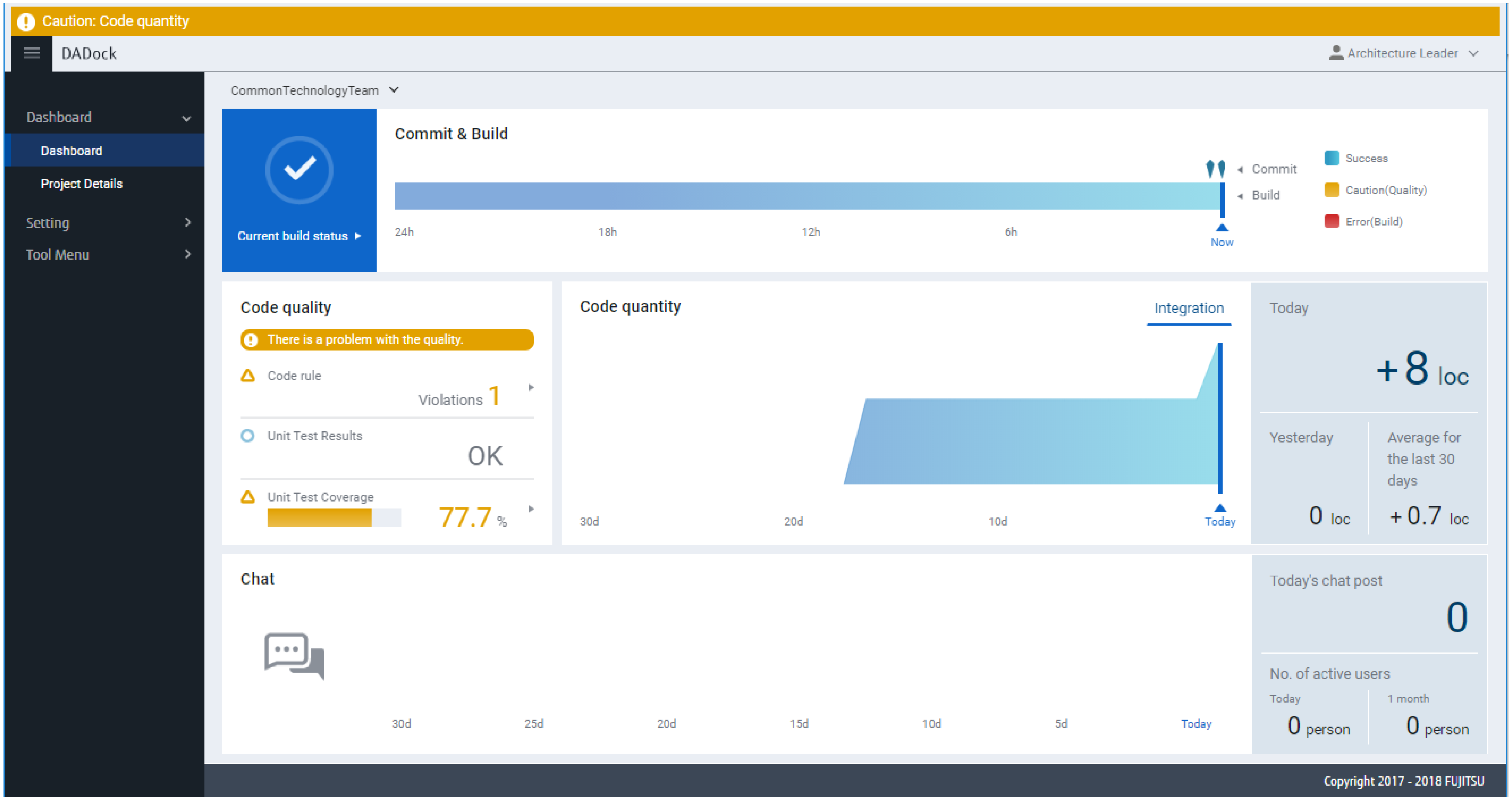
```java
package project;

import static org.junit.Assert.*;

import org.junit.Test;

public class CheckUtilTest {

    @Test
    public void isNullOrEmpty_valid() {
        assertFalse(CheckUtil.isNullOrEmpty("valid"));
    }

    @Test
    public void isNullOrEmpty_nullValue() {
        assertTrue(CheckUtil.isNullOrEmpty(null));
    }

    @Test
    public void isNullOrEmpty_emptyValue() {
        assertTrue(CheckUtil.isNullOrEmpty(""));
    }

    @Test
    public void isNullOrEmpty_blankValue() {
        assertTrue(CheckUtil.isNullOrEmpty("  "));
    }

    @Test
    public void isNumber_number() {
        assertTrue(CheckUtil.isNumber("1"));
    }

    @Test
    public void isNumber_minusNumber() {
        assertTrue(CheckUtil.isNumber("-1"));
    }

    @Test
    public void isNumber_string() {
        assertFalse(CheckUtil.isNumber("test"));
    }

}
```

4.From the context menu of TortoiseGit, click [Git Commit] then input the commit contents then click the [Commit & Push] button.（can be selected from the pulldown of the Commit button）

Item Field	**Velue**
Message	Fix util class
Changes made (double-click on file for diff)	Put a check mark in all the files

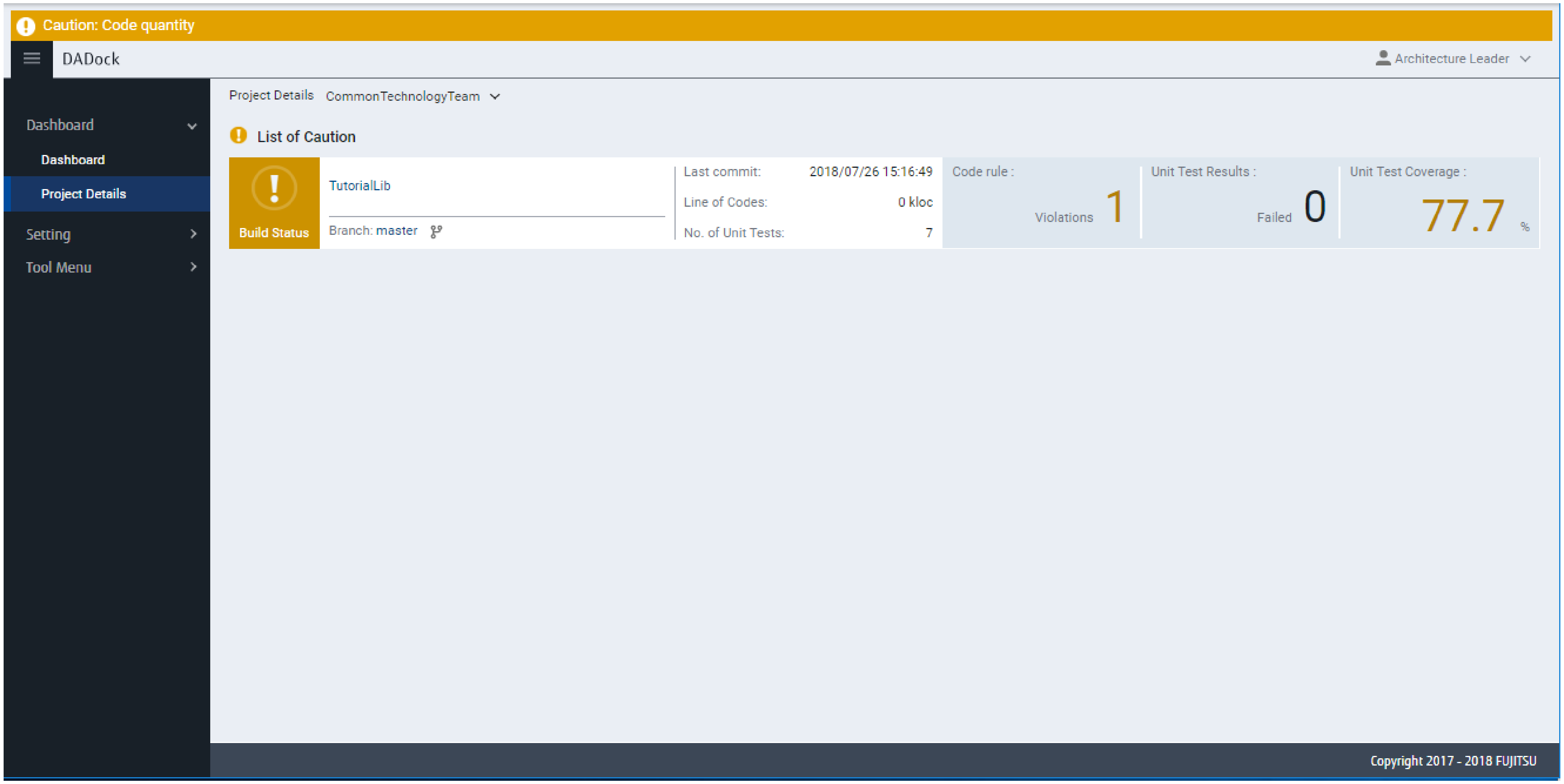
5.Login as arch0(Common Technology Team Leader) and the Dashboard page will be displayed.

6.The Quality status will be changed.



7.From the Menu Bar of DADock Portal, click [Project Details] and the Project Details Page will be displayed.

8.The same in the Dashboard Page, the quality status will be changed.



Merge Request （Standard user）

When developing a project as a standard user, create first a local branch of the project and make the modifications of the source code in that local branch. (*note: DADock recommends the assets approval process by merge request, so a standard user doesn't have a right to update the master branch) When done with the changes and the source code is already working, the standard user will submit a merge request to the project leader. When the leader approves the changes made in the source code, the changes will be reflect the changes to the master branch.

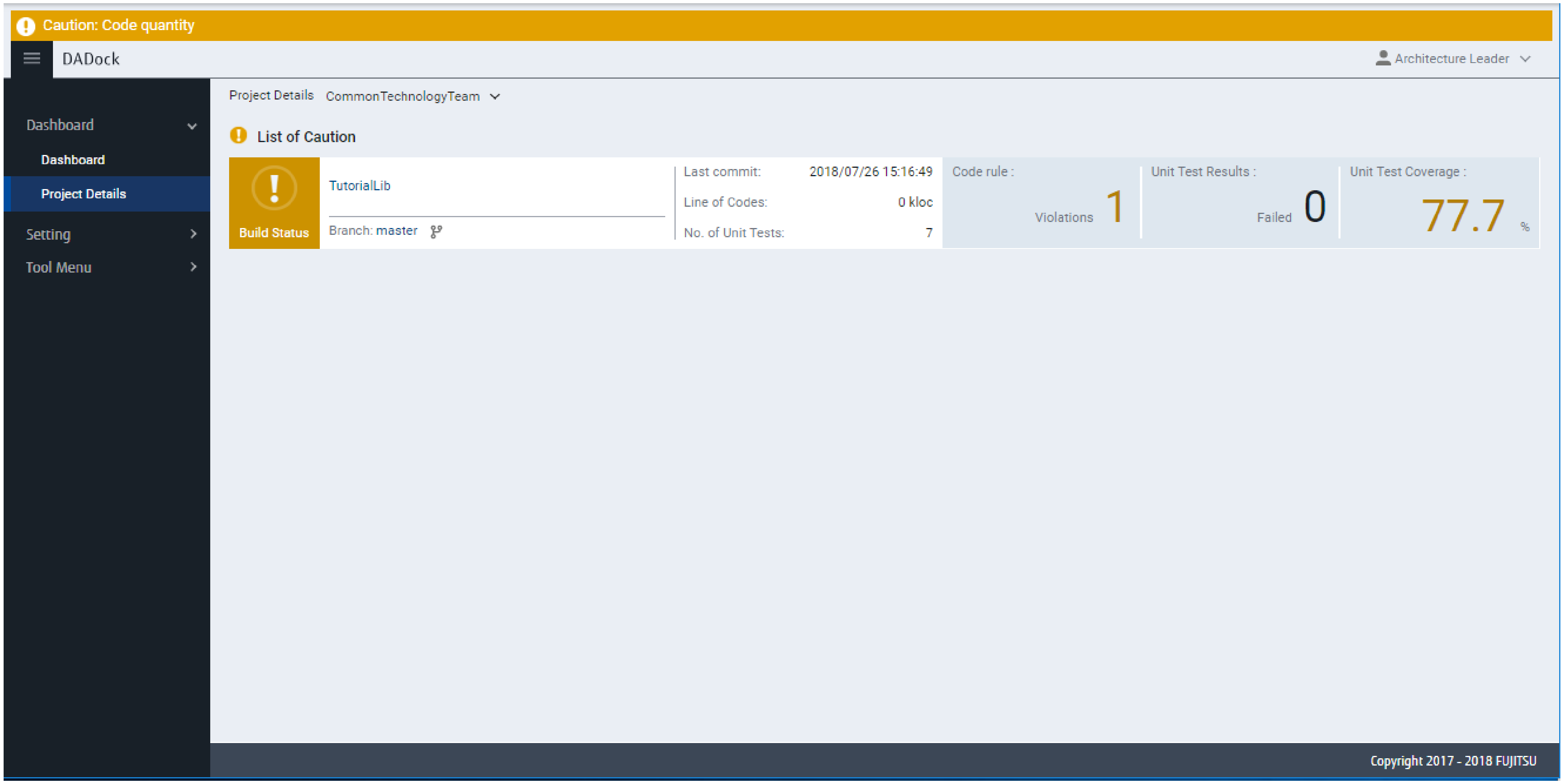
4.4. Storing of build artifacts in Artifactory

In this section, storing of the build artifacts in Artifactory and releasing the common components will be done.

> Either attach a GitLab Tag in the corresponding repository or comment out the following entries below inside the gitlab-ci.yml file located in the job_package stage.
>
> ```
> only:
>   - tags
> ```
>
> For the way on how to grant GitLab Tag, refer to the manuals below and execute the necessary settings.
>
> Separate Manual [DADock - Reference Guide] > How to attach GitLab Tag
> Separate Manual [DADock - Reference Guide] > GitLab Protected Tags settings procedures

> *Important*
>
> In storing the build artifacts in the Artifactory, the following settings are needed beforehand.
> Refer to the manual below and execute the settings.
>
> - Separate Manual [DADock - Reference Guide] > [JavaLibrary] > [JavaLibrary Pre-setting]

1.After reloading the Project Details Page of DADock Portal, click the [build status] of [TutorialLib] project.

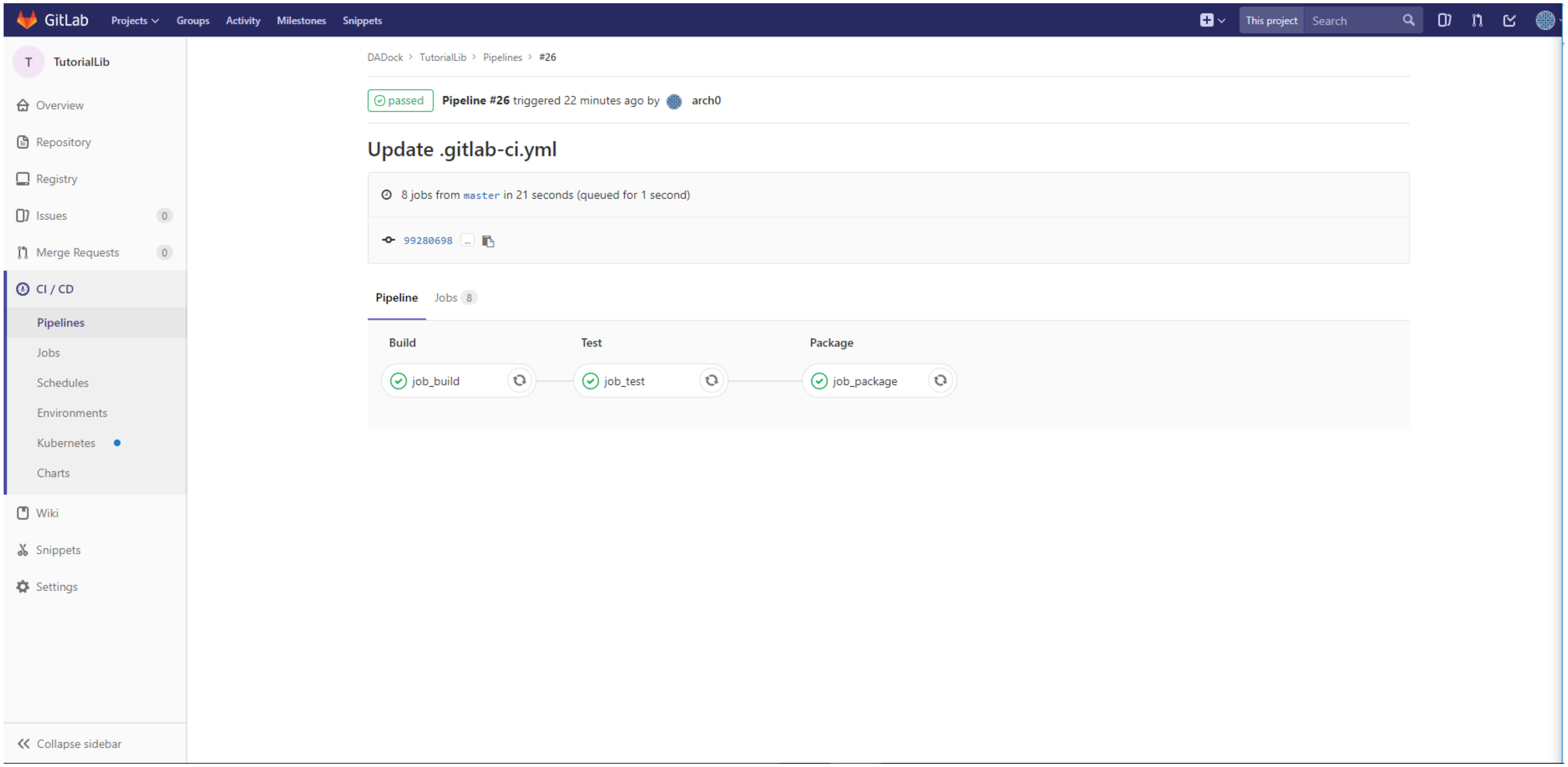


TutorialLib	Last commit: 2018/07/26 15:16:49	Code rule :	Unit Test Results :	Unit Test Coverage :	
Branch: master	Line of Codes: 0 kloc	Violations 1	Failed 0	77.7 %	
	No. of Unit Tests: 7				

2.Move into the GitLab Login Page.

3.Login as arch0(Common Technology Team Leader).

4.Move to the detailed page of the Pipeline status.



Pipeline #26 triggered 22 minutes ago by arch0

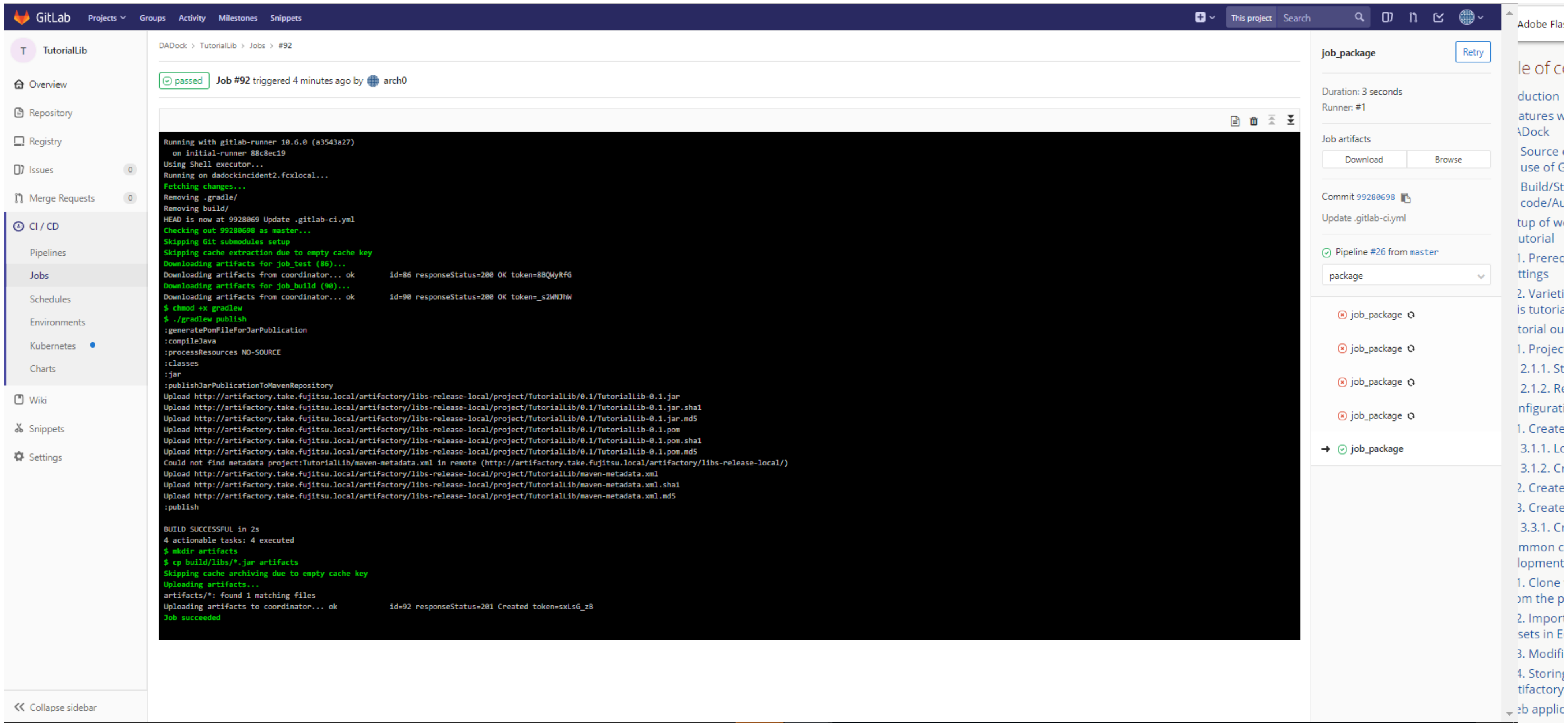
Update .gitlab-ci.yml

8 jobs from master in 21 seconds (queued for 1 second)

99280698

5.Click the [job_package] button to display the console page of the job, which stores the built deliverables.

6.Check that the execution of storing the build artifacts job is successful(the job status shown at the left side of the page is passed).



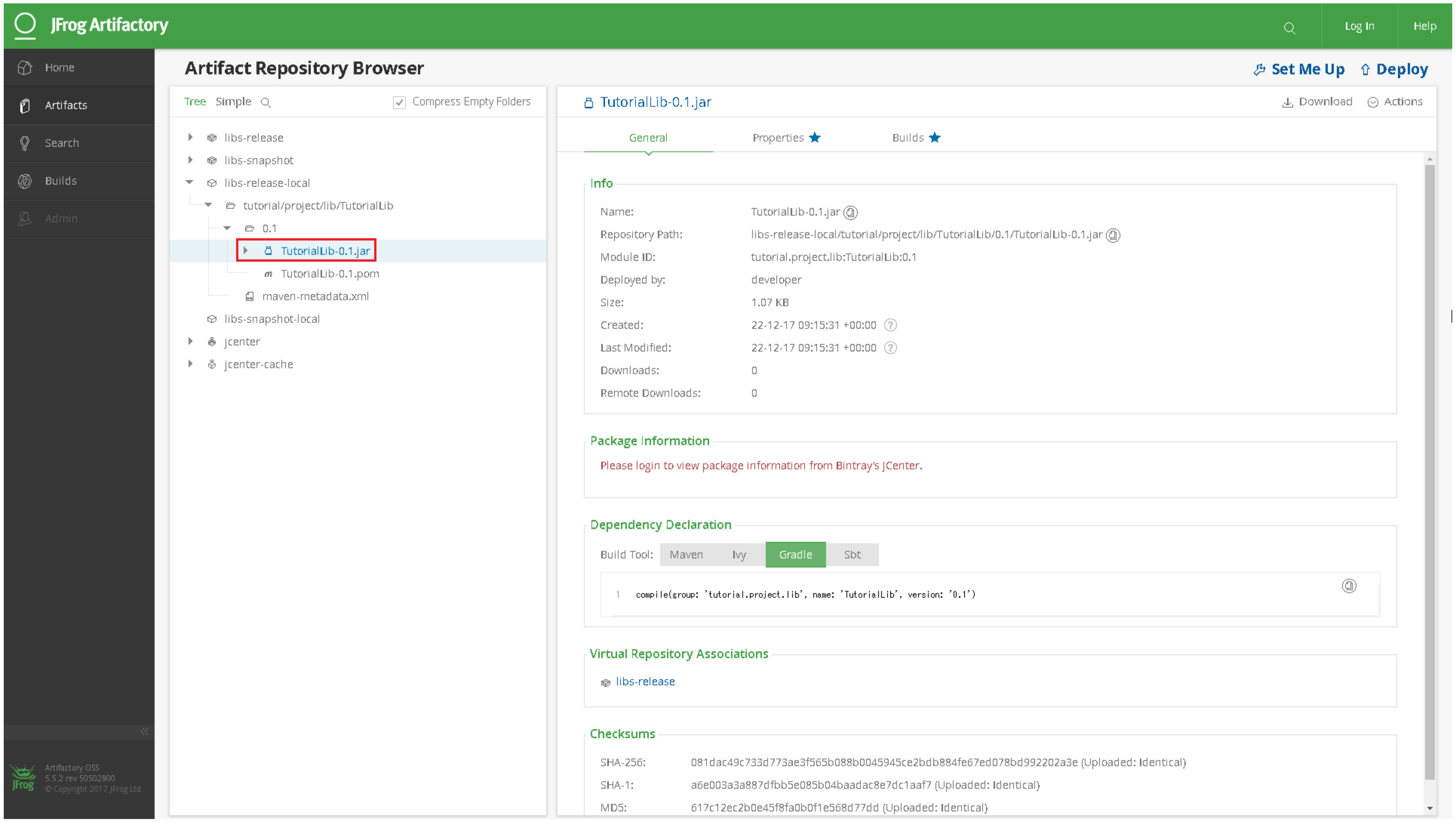
```
Running with gitlab-runner 10.6.0 (a3543a27)
  on initial-runner 88c8ec19
Using Shell executor...
Running on dadockincident2.fcxlocal...
Fetching changes...
Removing .gradle/
Removing build/
HEAD is now at 9928069 Update .gitlab-ci.yml
Checking out 99280698 as master...
Skipping Git submodules setup
Skipping cache extraction due to empty cache key
Downloading artifacts for job_test (86)...
Downloading artifacts from coordinator... ok        id=86 responseStatus=200 OK token=8BQWyRfG
Downloading artifacts for job_build (90)...
Downloading artifacts from coordinator... ok        id=90 responseStatus=200 OK token=_s2WNJhW
$ chmod +x gradlew
$ ./gradlew publish
:generatePomFileForJarPublication
:compileJava
:processResources NO-SOURCE
:classes
:jar
:publishJarPublicationToMavenRepository
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.jar
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.jar.sha1
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.jar.md5
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.pom
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.pom.sha1
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/0.1/TutorialLib-0.1.pom.md5
Could not find metadata project:TutorialLib/maven-metadata.xml in remote (http://artifactory.take.fujitsu.local/artifactory/libs-release-local/)
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/maven-metadata.xml
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/maven-metadata.xml.sha1
Upload http://artifactory.take.fujitsu.local/artifactory/libs-release-local/project/TutorialLib/maven-metadata.xml.md5
:publish

BUILD SUCCESSFUL in 2s
4 actionable tasks: 4 executed
$ mkdir artifacts
$ cp build/libs/*.jar artifacts
Skipping cache archiving due to empty cache key
Uploading artifacts...
artifacts/*: found 1 matching files
Uploading artifacts to coordinator... ok            id=92 responseStatus=201 Created token=sxLsG_zB
Job succeeded
```

7.Move back to the Dasboard Page of DADock Portal and click [Tool Menu] tab to expand the and the Tool Menu List Page will be displayed.

8.Click the [Artifactory] button to display the Top Page of the Artifactory.

9.At the left side of the page, select [Artifacts] to display the Artifacts Repository Page.

10.Check if the build artifacts [libs-release-local] is saved under the TutolialLib project.



Artifact Repository Browser

- libs-release
- libs-snapshot
- libs-release-local
  - tutorial/project/lib/TutorialLib
    - 0.1
      - TutorialLib-0.1.jar
      - TutorialLib-0.1.pom
    - maven-metadata.xml
- libs-snapshot-local
- jcenter
- jcenter-cache

TutorialLib-0.1.jar

Info

Name:	TutorialLib-0.1.jar
Repository Path:	libs-release-local/tutorial/project/lib/TutorialLib/0.1/TutorialLib-0.1.jar
Module ID:	tutorial.project.lib:TutorialLib:0.1
Deployed by:	developer
Size:	1.07 KB
Created:	22-12-17 09:15:31 +00:00
Last Modified:	22-12-17 09:15:31 +00:00
Downloads:	0
Remote Downloads:	0

Package Information

Please login to view package information from Bintray's JCenter.

Dependency Declaration

Build Tool: Maven Ivy [✓ Gradle] Sbt

```
compile(group: 'tutorial.project.lib', name: 'TutorialLib', version: '0.1')
```

Virtual Repository Associations

libs-release

Checksums

SHA-256:	081dac49c733d773ae3f565b088b0045945ce2bdb884fe67ed078bd992202a3e (Uploaded: Identical)
SHA-1:	a6e003a3a887dfbb5e085b04baadac8e7dc1aaf7 (Uploaded: Identical)
MD5:	617c12ec2b0e45f8fa0b0f1e568d77dd (Uploaded: Identical)

5. Web application development

In this section, how the development flow of two development teams working on the same web application development repository will be learned.

Team A and Team B will develop a web application with the use of TutorialWeb, a repository created in the section Configuration management. Refer to Tutorial outline for the details about repository and team structures.

The following is the outline for the development flow:

- The developer from team B starts with the development by adding a low quality source code
- The developer from team A starts with the development by adding a high quality source code
- The development leader of team A will use Team Directory settings of DADock to set that only the quality status information of team A source code can be checked/viewed.

5.1. Team B development

5.1.1. Development preparation

Create a directory for the source codes of team B.

1.Create the following directory in the local machine.
`C:/Tutorial/TutorialWeb_B0`

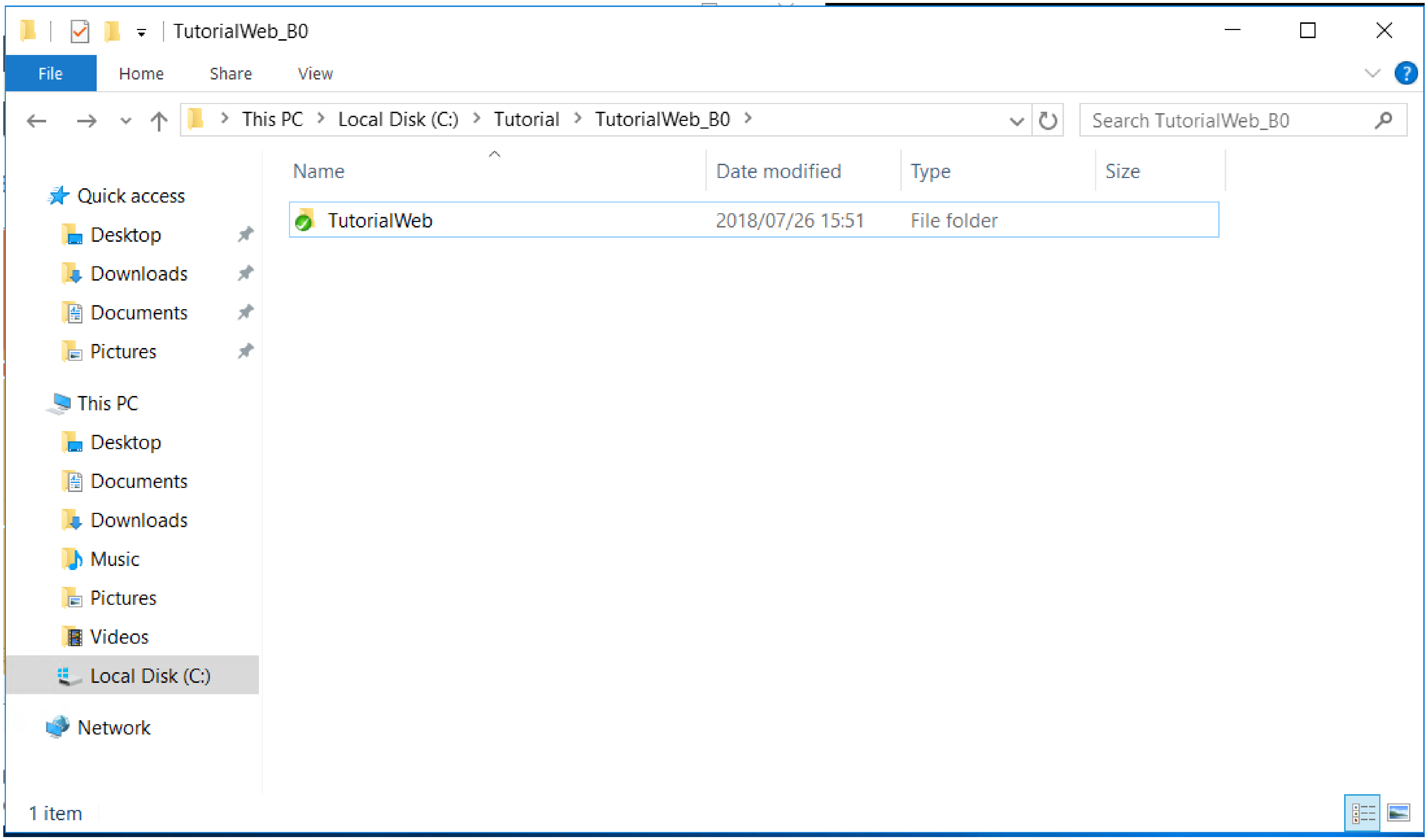
2.Login as app.b0 in DADock Portal then clone TutorialWeb repository in the newly created directory.

Refer to Clone the development assets from the project.

When username and password are required

When the username and password of TortoiseGit are required, input the user creadentials of [app.b0].

After cloning, the directory `C:/Tutorial/TutorialWeb_B0/tutorialweb` will be automatically created.



Name	Date modified	Type	Size
TutorialWeb	2018/07/26 15:51	File folder	

3.Import the cloned development assets in Eclipse.

Refer to Import the development assets in Eclipse.

5.1.2. Initial changes in the assets

1.Since the following source codes will not be used in this tutorial, delete the following files.

```
src/main/java/project/CalculationInputCalculateServlet.java
src/main/java/project/CalculationInputCreateServlet.java
src/main/java/project/CalculationUtil.java
src/main/webapp/WEB-INF/jsp/CalculationInput.jsp
src/main/webapp/WEB-INF/jsp/CalculationResult.jsp
src/test/java/project/CalculationUtilTest.java
```

2.In order to switch pages for the implementation of team A and team B, edit the source code `src/main/webapp/index.jsp` as follows.

list 5. index.jsp

HTML

```html
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<title>TutorialWeb</title>
<meta charset="UTF-8">
</head>
<body>
        <input type="button" value="Team A - Calculation Input" onClick="document.location='/TutorialWeb/TeamA/CalculationInput.Create'">
        <input type="button" value="Team B - Calculation Input" onClick="document.location='/TutorialWeb/TeamB/CalculationInput.Create'">
</body>
</html>
```

3.[Commit & Push] the edited source code and reflect all the changes in the repository. From the context menu of TortoiseGit, click [Git Commit] and input the description of the commit process then click the [Commit & Push] button (can be selected from the pull-down menu of the commit button).

Item Field	Input Value
Message	Fix initial assets
Changes made (double-click on file for diff)	Put a check mark in all of the files

When username and password are required

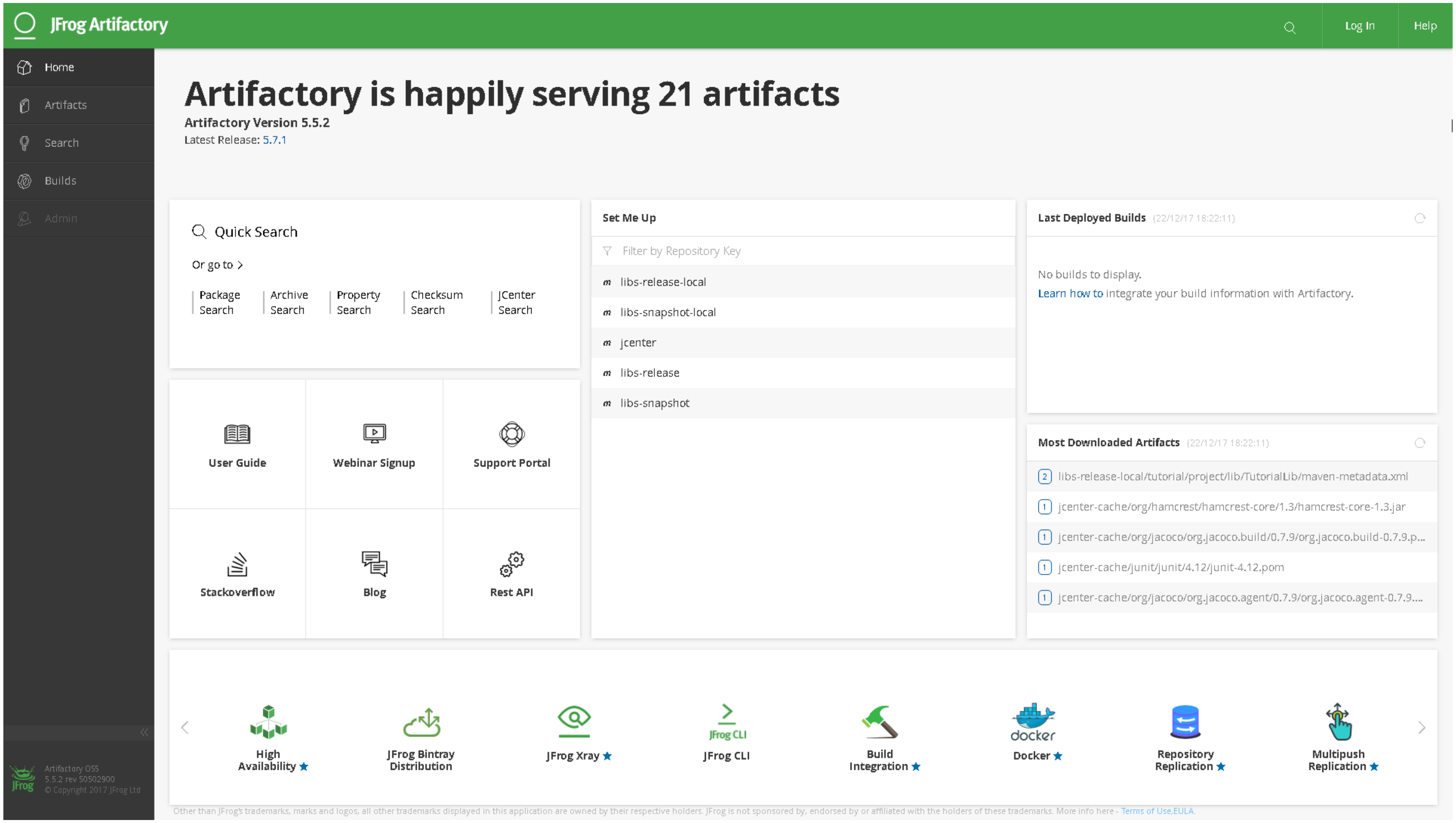
When the username and password of TortoiseGit are required, input the user credentials of [app.b0].

5.1.3. Reference settings of the common components project

Web application refers to the common components that is created in Common components development section.

With the following procedures, add the existing dependencies of the common components.

1.Login to DADock Portal as [app.b0] then, expand the [Tool Menu] tab click [Artifactory] to display the Top Page of Artifactory.



Artifactory is happily serving 21 artifacts
Artifactory Version 5.5.2
Latest Release: 5.7.1

Set Me Up
- libs-release-local
- libs-snapshot-local
- jcenter
- libs-release
- libs-snapshot

Last Deployed Builds (22/12/17 18:22:11)
No builds to display.
Learn how to integrate your build information with Artifactory.

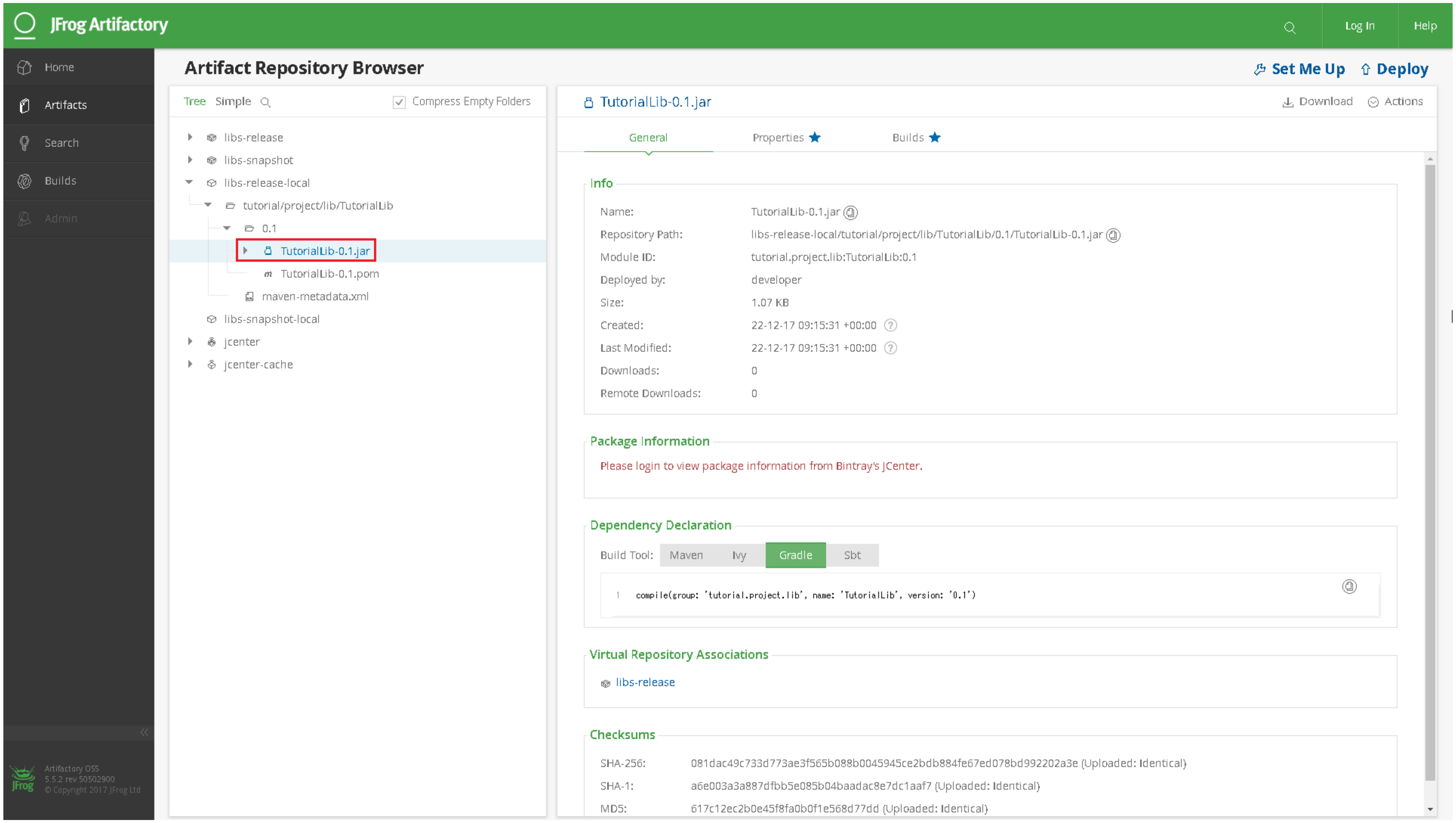
Most Downloaded Artifacts (22/12/17 18:22:11)

2	libs-release-local/tutorial/project/lib/TutorialLib/maven-metadata.xml
1	jcenter-cache/org/hamcrest/hamcrest-core/1.3/hamcrest-core-1.3.jar
1	jcenter-cache/org/jacoco/org.jacoco.build/0.7.9/org.jacoco.build-0.7.9.p...
1	jcenter-cache/junit/junit/4.12/junit-4.12.pom
1	jcenter-cache/org/jacoco/org.jacoco.agent/0.7.9/org.jacoco.agent-0.7.9....

Artifactory OSS
5.5.2 rev 50502900
© Copyright 2017 JFrog Ltd

Other than JFrog's trademarks, marks and logos, all other trademarks displayed in this application are owned by their respective holders. JFrog is not sponsored by, endorsed by or affiliated with the holders of these trademarks. More info here - Terms of Use,EULA

2.Expand the [libs-release-local] tree and select the file [project/lib/TutorialLib/0.1/TutorialLib-0.1.jar].



TutorialLib-0.1.jar

Info

Name:	TutorialLib-0.1.jar
Repository Path:	libs-release-local/tutorial/project/lib/TutorialLib/0.1/TutorialLib-0.1.jar
Module ID:	tutorial.project.lib:TutorialLib:0.1
Deployed by:	developer
Size:	1.07 KB
Created:	22-12-17 09:15:31 +00:00
Last Modified:	22-12-17 09:15:31 +00:00
Downloads:	0
Remote Downloads:	0

Package Information

Please login to view package information from Bintray's JCenter.

Dependency Declaration

Build Tool: Maven Ivy [✓ Gradle] Sbt

```
compile(group: 'tutorial.project.lib', name: 'TutorialLib', version: '0.1')
```

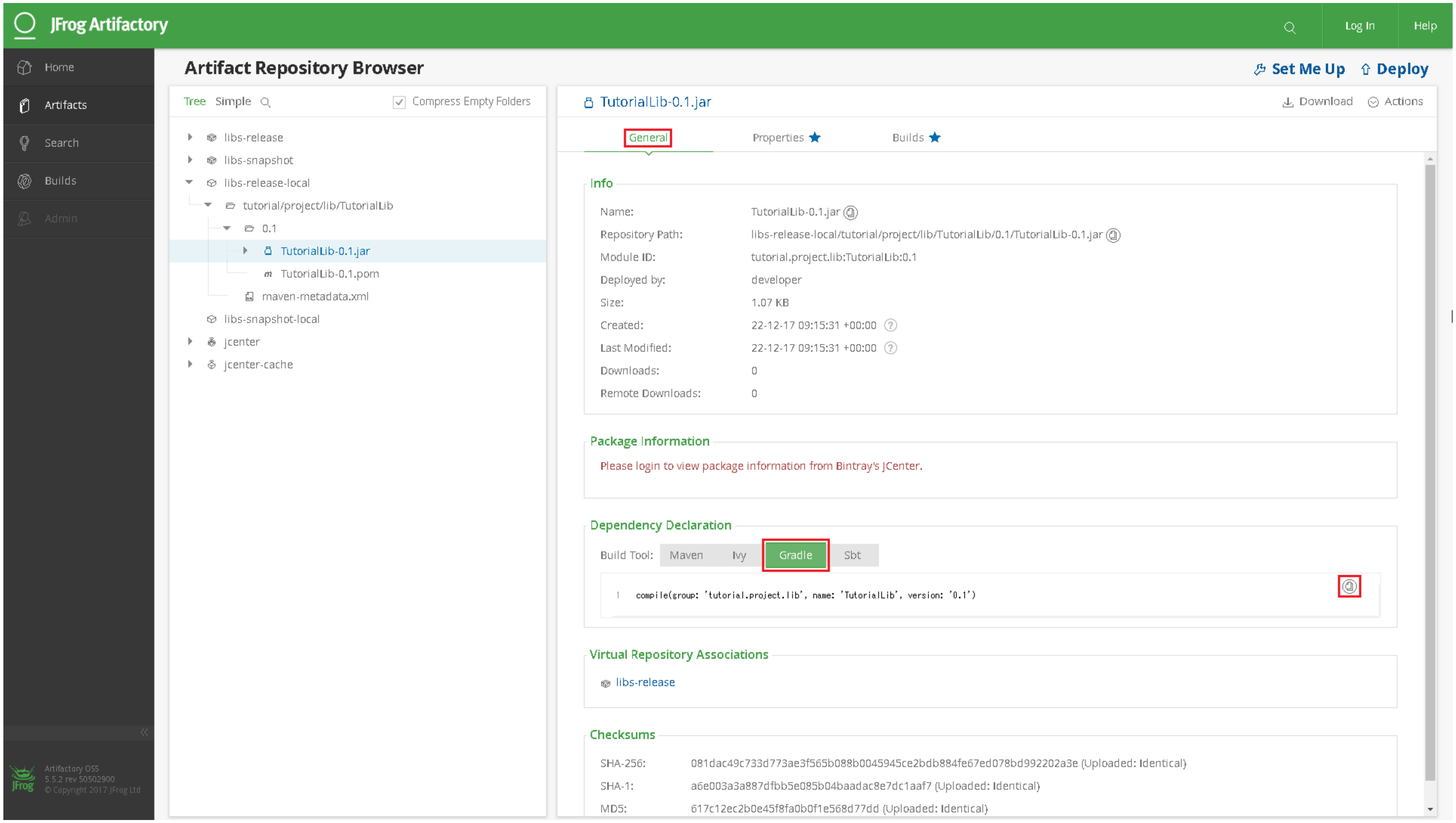
Virtual Repository Associations

libs-release

Checksums

SHA-256:	081dac49c733d773ae3f565b088b0045945ce2bdb884fe67ed078bd992202a3e (Uploaded: Identical)
SHA-1:	a6e003a3a887dfbb5e085b04baadac8e7dc1aaf7 (Uploaded: Identical)
MD5:	617c12ec2b0e45f8fa0b0f1e568d77dd (Uploaded: Identical)

3.From the [General] Tab, select [Dependency Declaration] then, select [Gradle]. At the right side of the page, click the button that displays the message "Copy snippet to clipboard" when the cursor is hovered.



TutorialLib-0.1.jar

Info

Name:	TutorialLib-0.1.jar
Repository Path:	libs-release-local/tutorial/project/lib/TutorialLib/0.1/TutorialLib-0.1.jar
Module ID:	tutorial.project.lib:TutorialLib:0.1
Deployed by:	developer
Size:	1.07 KB
Created:	22-12-17 09:15:31 +00:00
Last Modified:	22-12-17 09:15:31 +00:00
Downloads:	0
Remote Downloads:	0

Package Information

Please login to view package information from Bintray's JCenter.

Dependency Declaration

Build Tool: Maven Ivy [✓ Gradle] Sbt

```
compile(group: 'tutorial.project.lib', name: 'TutorialLib', version: '0.1')
```

Virtual Repository Associations

libs-release

Checksums

SHA-256:	081dac49c733d773ae3f565b088b0045945ce2bdb884fe67ed078bd992202a3e (Uploaded: Identical)
SHA-1:	a6e003a3a887dfbb5e085b04baadac8e7dc1aaf7 (Uploaded: Identical)
MD5:	617c12ec2b0e45f8fa0b0f1e568d77dd (Uploaded: Identical)

4.In Eclipse, go to [Project Explorer] windows and open the [TutorialWeb] repository then open the following file.

- build.gradle

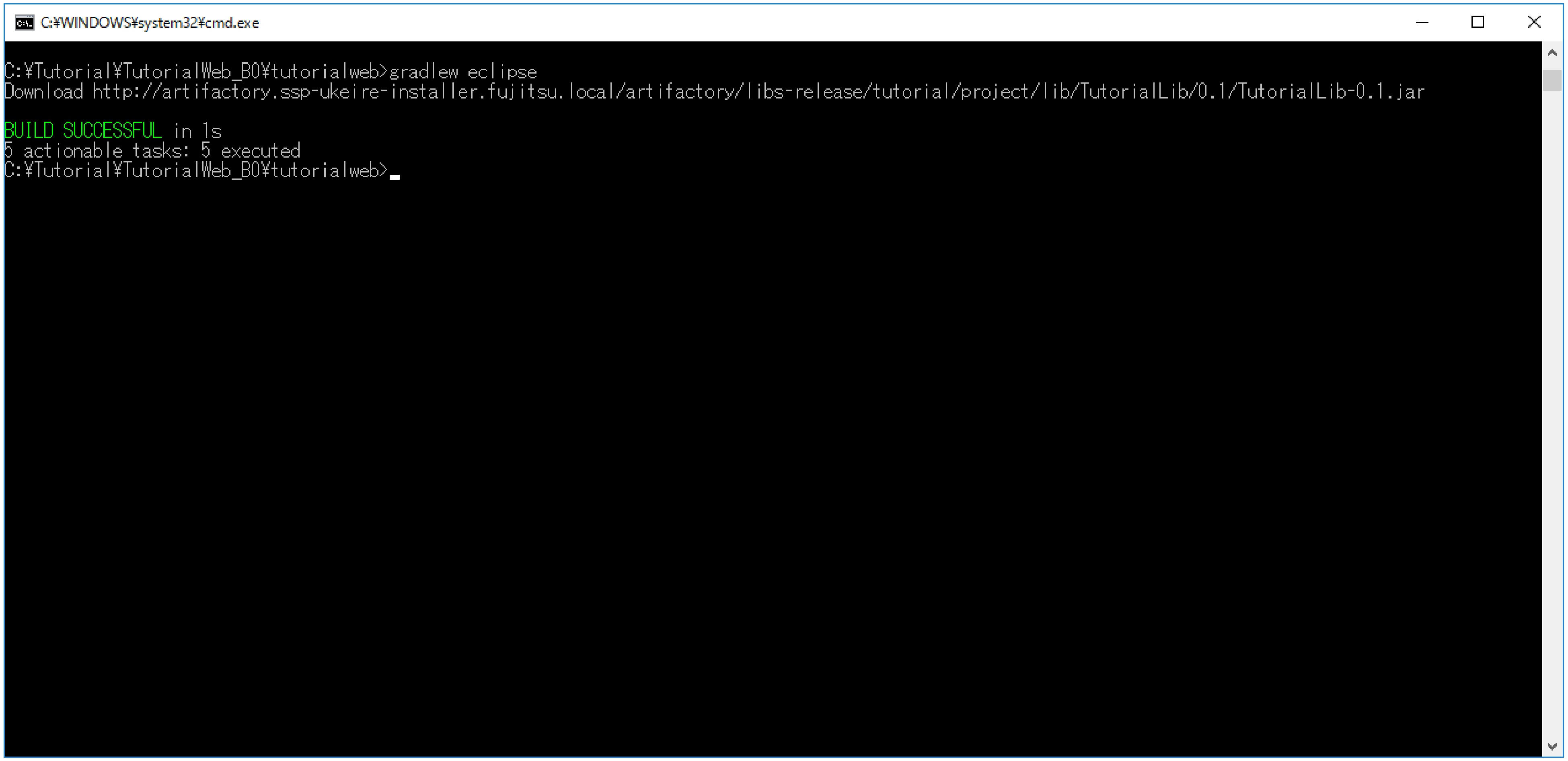
5.In the [dependencies] task, add the dependencies to the artifacts of Java Library project.

After editing, the description of build.gradle file will be changed as follows.

list 6. build.gradle

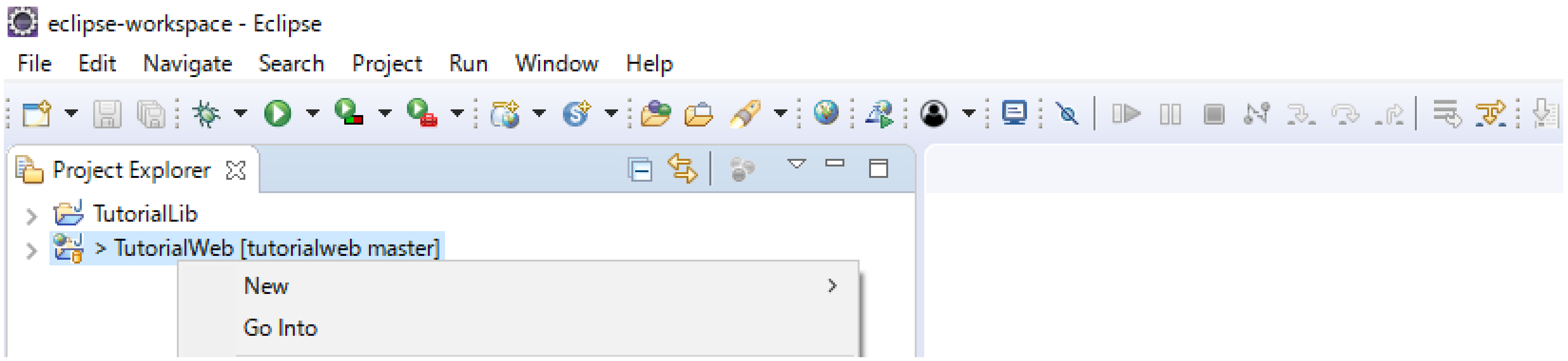
```
- ommitted -

dependencies {
    compile(group: 'javax.servlet', name: 'javax.servlet-api', version: '3.1.0')
    compile(group: 'project', name: 'TutorialLib', version: '0.1')
    testCompile(group: 'junit', name: 'junit', version: '4.12')
}

- ommitted -
```

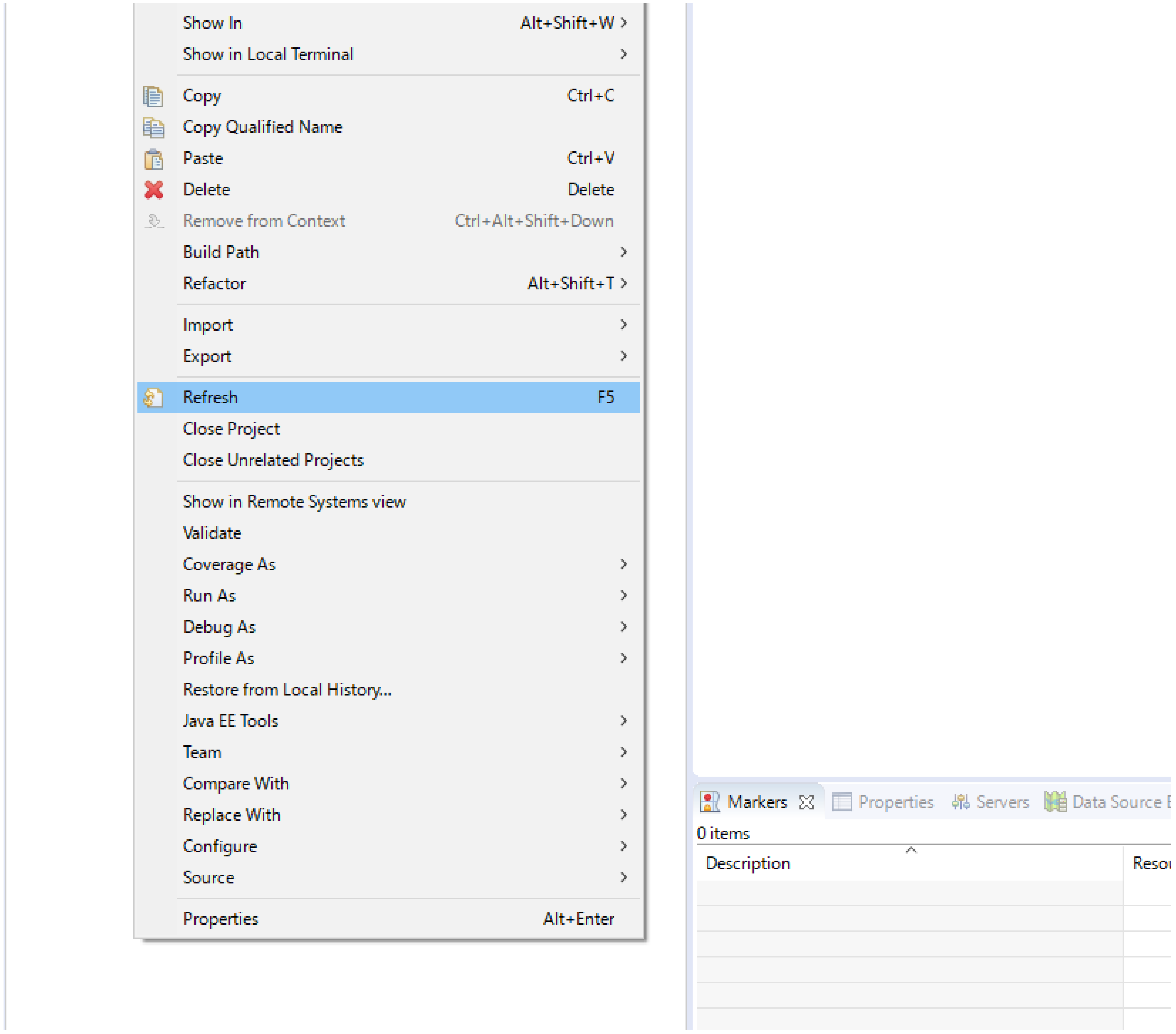
6.In command prompt, move the current directory to `C:/Tutorial/TutorialWeb_B0/tutorialweb` and execute `gradlew eclipse` command.



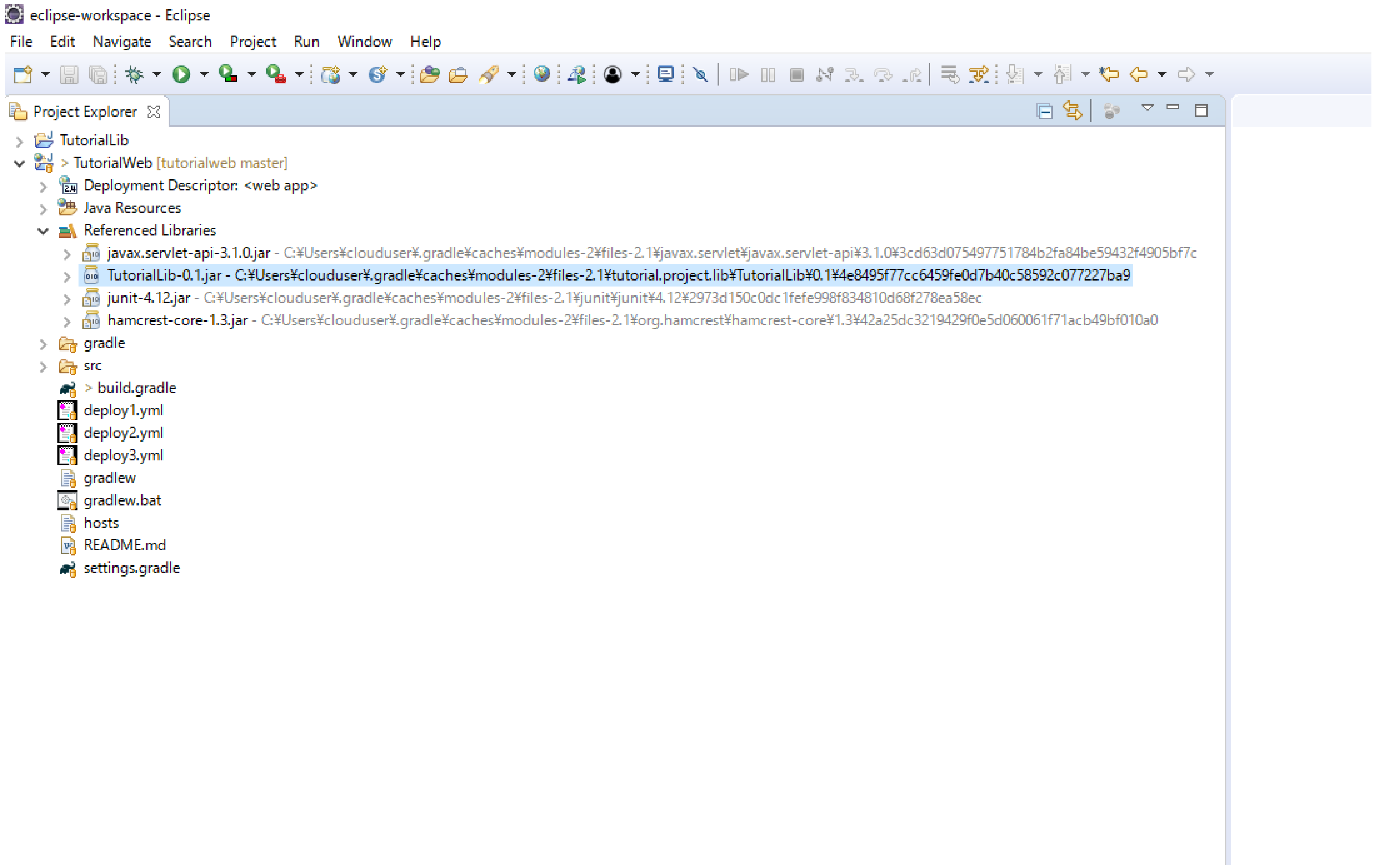
```
C:¥Tutorial¥TutorialWeb_BO¥tutorialweb>gradlew eclipse
Download http://artifactory.ssp-ukeire-installer.fujitsu.local/artifactory/libs-release/tutorial/project/lib/TutorialLib/0.1/TutorialLib-0.1.jar

BUILD SUCCESSFUL in 1s
5 actionable tasks: 5 executed
C:¥Tutorial¥TutorialWeb_BO¥tutorialweb>_
```

7.In Eclipse, open the [Project Explorer] windows and then right click the [TutorialWeb] repository. From the context menu, click [Refresh].


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
TutorialLib
> TutorialWeb [tutorialweb master]
New
Go Into


Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import
Export
Refresh F5
Close Project
Close Unrelated Projects
Show in Remote Systems view
Validate
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Java EE Tools
Team
Compare With
Replace With
Configure
Source
Properties Alt+Enter
Markers
Properties
Servers
Data Source E
0 items
Description
Resou

8.In Eclipse, open the [Project Explorer] windows and open the [Referenced Libraries] tree. Check if TutorialLib-0.1.jar file is included in the directory.


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer

```
> TutorialLib
v > TutorialWeb [tutorialweb master]
  > Deployment Descriptor: <web app>
  > Java Resources
  v Referenced Libraries
    > javax.servlet-api-3.1.0.jar - C:¥Users¥clouduser¥.gradle¥caches¥modules-2¥files-2.1¥javax.servlet¥javax.servlet-api¥3.1.0¥3cd63d075497751784b2fa84be59432f4905bf7c
    > TutorialLib-0.1.jar - C:¥Users¥clouduser¥.gradle¥caches¥modules-2¥files-2.1¥tutorial.project.lib¥TutorialLib¥0.1¥4e8495f77cc6459fe0d7b40c58592c077227ba9
    > junit-4.12.jar - C:¥Users¥clouduser¥.gradle¥caches¥modules-2¥files-2.1¥junit¥junit¥4.12¥2973d150c0dc1fefe998f834810d68f278ea58ec
    > hamcrest-core-1.3.jar - C:¥Users¥clouduser¥.gradle¥caches¥modules-2¥files-2.1¥org.hamcrest¥hamcrest-core¥1.3¥42a25dc3219429f0e5d060061f71acb49bf010a0
  > gradle
  > src
    > build.gradle
    deploy1.yml
    deploy2.yml
    deploy3.yml
    gradlew
    gradlew.bat
    hosts
    README.md
    settings.gradle
```

With these procedures, the common components project can now be used.

5.1.4. Web Application Implementation

Team B will implement a web application that checks the results of Mathematical addition.
At this point, it is expected that the developer of team B will write a low quality source code.

1.Under the directory `C:/Tutorial/TutorialWeb_B0/TutorialWeb` , create the following new directories for the source codes of team B.

```
src/main/java/tutorial/project/web/teamb
src/main/webapp/WEB-INF/jsp/teamb
```

2.Add the following 3 java files under `src/main/java/tutorial/project/web/teamb` directory.

list 7. CalculationUtilB.java

```java
package project.web.teamb;

/**
 * This is a utility class that performs calculations.
 */
public final class CalculationUtilB {
        /**
         * Add arguments x and y.
         *
         * @param x
         *            argument
         * @param y
         *            argument
         * @return calculation result
         */
        public static int add(int x, int y) {
                return x + y;
        }
}
```

list 8. CalculationInputCalculateServletB.java

JAVA

```java
package project.web.teamb;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import project.lib.CheckUtil;

/**
 * A servlet class that processes the calculate button of the calculation input
 * screen.
 */
@WebServlet("/TeamB/CalculationInputCaluculate.do")
public class CalculationInputCalculateServletB extends HttpServlet {
        /**
         * Constructs a new instance.
         */
        public CalculationInputCalculateServletB() {
        }

        /**
         * Handle a POST request.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         * @throws IOException
         *             if an input or output error is detected when the servlet
         *             handles the GET request
         * @throws ServletException
         *             if the request for the GET could not be handled
         */
        @Override
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                String formArgX = request.getParameter("argX");
                String formArgY = request.getParameter("argY");
                if (CheckUtil.isNullOrEmpty(formArgX)) {
                        request.setAttribute("errorMessage", "argumentX is required.");
                        request.setAttribute("argX", formArgX);
                        request.setAttribute("argY", formArgY);
                        request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationInputB.jsp").forward(request, response);
                        return;
                }
                if (CheckUtil.isNullOrEmpty(formArgY)) {
                        request.setAttribute("errorMessage", "argumentY is required.");
                        request.setAttribute("argX", formArgX);
                        request.setAttribute("argY", formArgY);
                        request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationInputB.jsp").forward(request, response);
                        return;
                }

                int argX;
                int argY;
                if (CheckUtil.isNumber(formArgX)) {
                        argX = Integer.parseInt(formArgX);
                } else {
                        request.setAttribute("errorMessage", "argumentX must be numeric.");
                        request.setAttribute("argX", formArgX);
                        request.setAttribute("argY", formArgY);
                        request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationInputB.jsp").forward(request, response);
                        return;
                }
                if (CheckUtil.isNumber(formArgY)) {
                        argY = Integer.parseInt(formArgY);
                } else {
                        request.setAttribute("errorMessage", "argumentY must be numeric.");
                        request.setAttribute("argX", formArgX);
                        request.setAttribute("argY", formArgY);
                        request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationInputB.jsp").forward(request, response);
                        return;
                }

                int result = CalculationUtilB.add(argX, argY);
                request.setAttribute("argX", argX);
                request.setAttribute("argY", argY);
                request.setAttribute("result", result);
                request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationResultB.jsp").forward(request, response);
        }
}
```

list 9. CalculationInputCreateServletB.java

JAVA

```java
package project.web.teamb;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * A servlet class that initializes the calculation input screen.
 */
@WebServlet("/TeamB/CalculationInput.Create")
public class CalculationInputCreateServletB extends HttpServlet {
        /** serialVersionUID */
        private static final long serialVersionUID = 2972265646531623810L;

        /**
         * Constructs a new instance.
         */
        public CalculationInputCreateServletB() {
        }

        /**
         * Handle a GET request.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         * @throws IOException
         *             if an input or output error is detected when the servlet
         *             handles the GET request
         * @throws ServletException
         *             if the request for the GET could not be handled
         */
        @Override
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                request.getRequestDispatcher("/WEB-INF/jsp/teamb/CalculationInputB.jsp").forward(request, response);
        }
}
```

Also, add the following 2 jsp files under the directory `src/main/webapp/WEB-INF/jsp/teamb` .

list 10. CalculationInputB.jsp

```java
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Team B - Calculation Input</title>
        <meta charset="UTF-8">
</head>
<body>
        ${errorMessage}
        <form action="<%=request.getContextPath()%>/TeamB/CalculationInputCaluculate.do" method="POST">
                argumentX
                <input type="text" name="argX" value="${requestScope.argX}"/><br>
                argumentY
                <input type="text" name="argY" value="${requestScope.argY}"/><br>
                <input type="submit" value="Calculate">
        </form>
</body>
</html>
```

list 11. CalculationResultB.jsp

```java
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Team B - Calculation Result</title>
        <meta charset="UTF-8">
</head>
<body>
        ${requestScope.argX} + ${requestScope.argY} = ${requestScope.result}
</body>
</html>
```

3.[Commit & Push] the modified source codes and reflect all the changes in the repository. From the context menu of [TortoiseGit], click [Git Commit] and input the desciption of the commit then click the [Commit & Push] button (can be selected from the pull-down menu of Commit button).

Item Field	**Value**
Message	Add team B implementation
Changes made (double-click on file for diff)	Put a check mark in all the files

When username and password are required

When the username and password of TortoiseGit are required, input the user credentials of [app.b0].

5.2. Team A development

5.2.1. Development preparation

Create directory for cloning the development assets from the repository and storing the source code of team A.

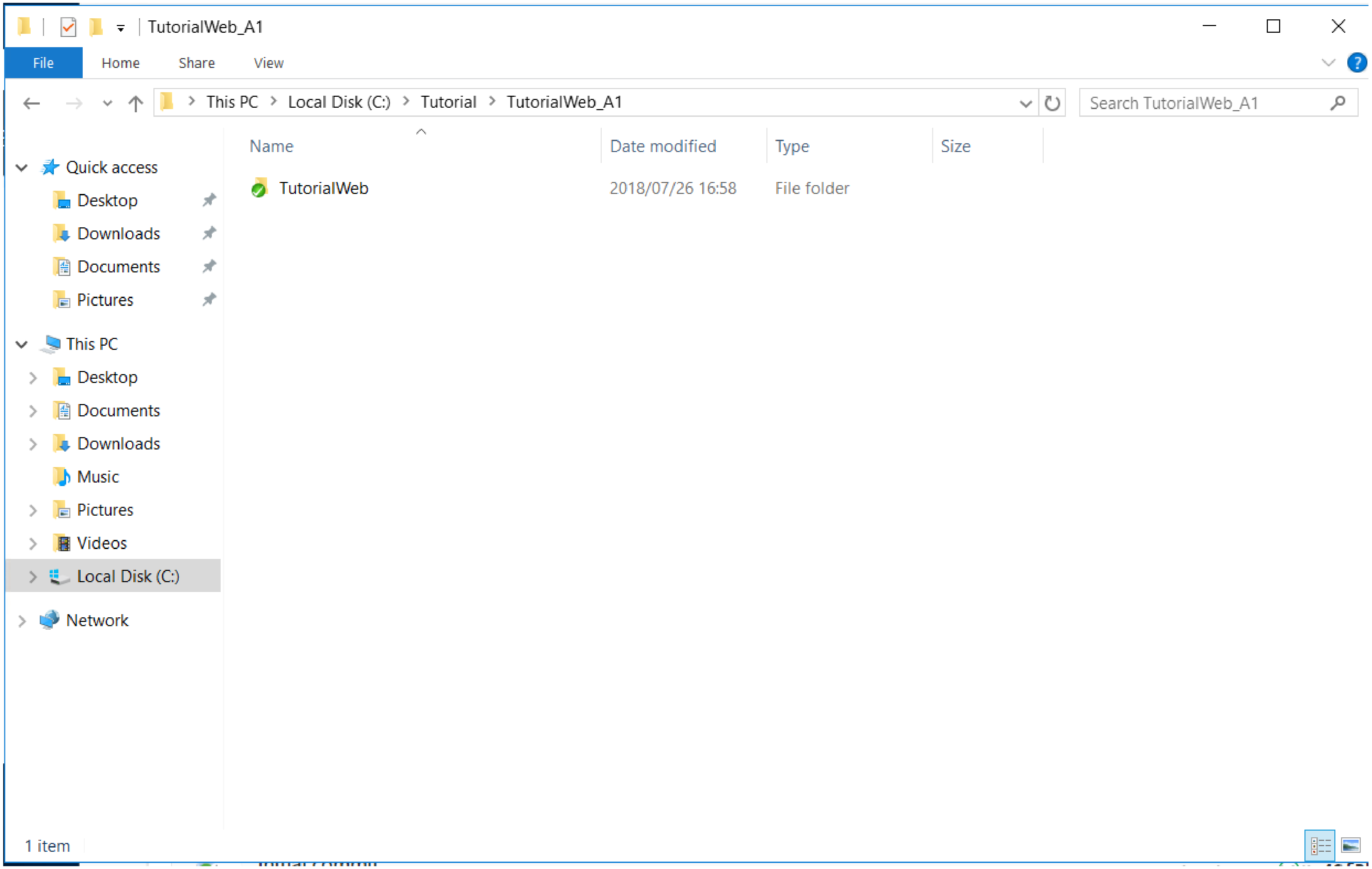
1.In the local machine, create the following directory.

```
C:/Tutorial/TutorialWeb_A1
```

2.Login to DADock Portal as [app.a1] user and clone the [TutorialWeb] repository in the newly created directory.

Refer to Clone the development assets from the project.

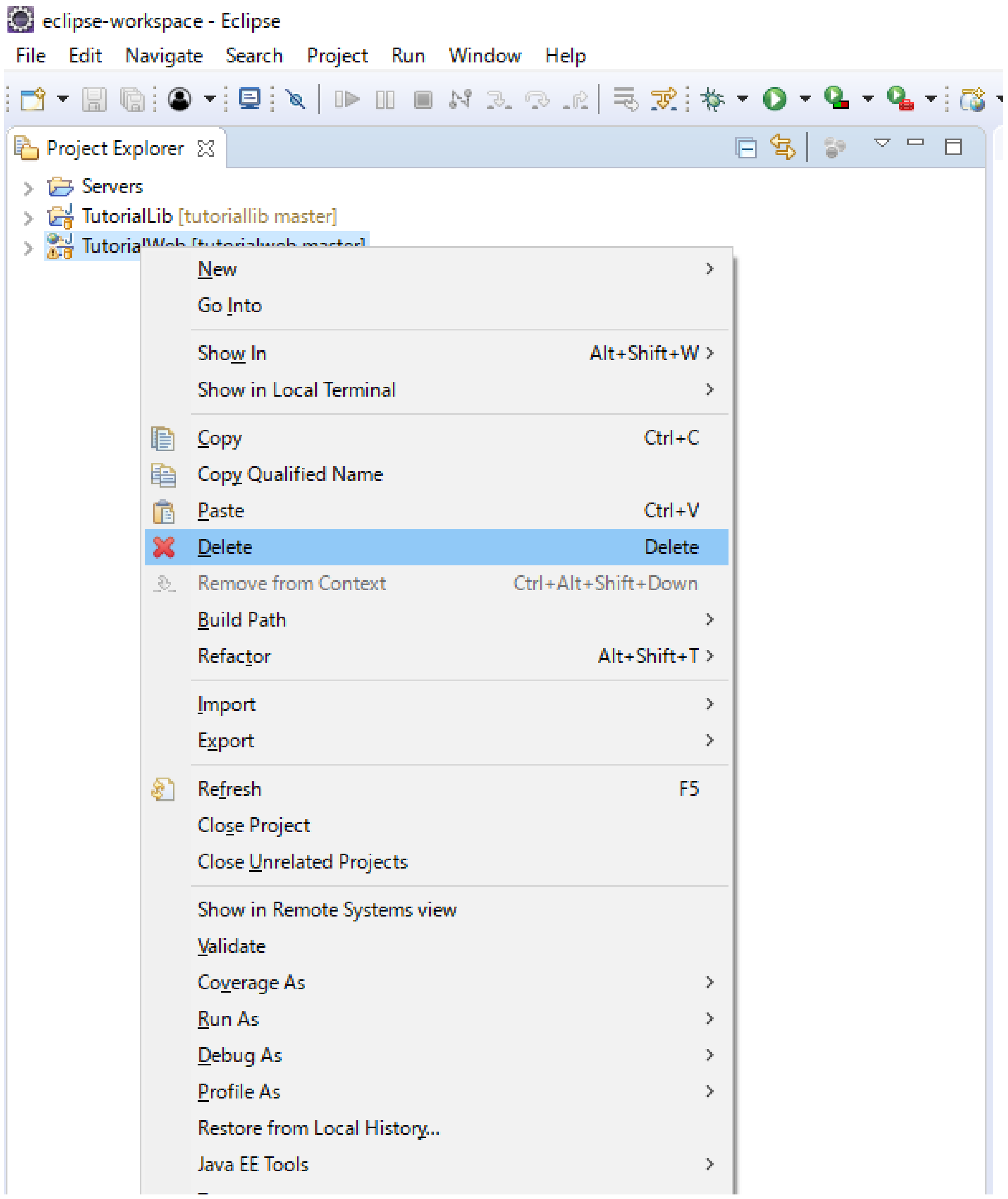
After cloning, the directory `C:/Tutorial/TutorialWeb_A1/tutorialweb` will be automatically created.

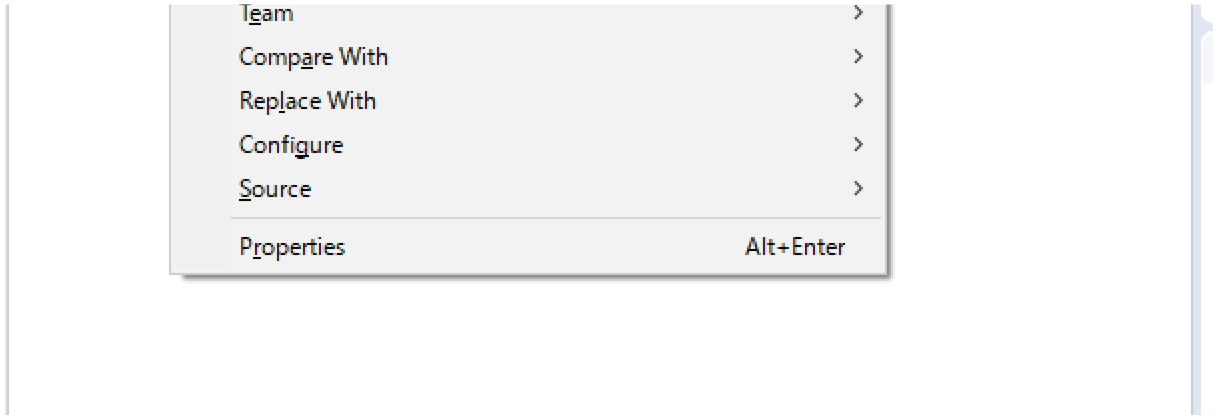


Name	Date modified	Type	Size
TutorialWeb	2018/07/26 16:58	File folder	

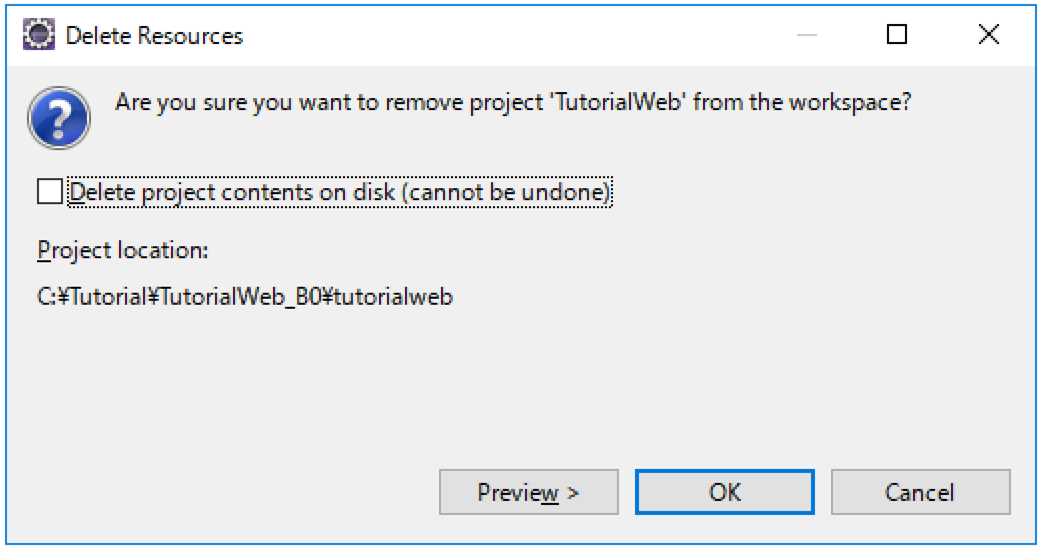
3.In order to import the development assets of team A in Eclipse, delete first [TutorialWeb] project Development preparation in Eclipse.

In Eclipse, select [Project Explorer] then right click [TutorialWeb] project then from the context menu, click the [Delete] button.


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
Servers
TutorialLib [tutoriallib master]
TutorialWeb [tutorialweb master]
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import
Export
Refresh F5
Close Project
Close Unrelated Projects
Show in Remote Systems view
Validate
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Java EE Tools



Check if the [Delete project contents on disk (cannot be undone)] does not have a check mark and click the [OK] button.


Delete Resources

Are you sure you want to remove project 'TutorialWeb' from the workspace?

☐ Delete project contents on disk (cannot be undone)

Project location:

C:¥Tutorial¥TutorialWeb_B0¥tutorialweb

4.Import the cloned development assets in Eclipse.

Refer to Importing the development assets in Eclipse.

5.2.2. Web application Implementation

Team A will implement a web application that checks the results of Mathematical multiplication.
At this point, it is expected that the developer of team A will write a high quality source code.

1.Under the directory `C:/Tutorial/TutorialWeb_A1/tutorialweb` , create the following new directories for the source codes of team A.

```
src/main/java/project/web/teama
src/main/webapp/WEB-INF/jsp/teama
src/test/java/project/web/teama
```

2.Add the following 3 java files under `src/main/java/project/web/teama` directory.

list 12. CalculationUtilA.java

JAVA

```java
package project.web.teama;

/**
 * This is a utility class that performs calculations.
 */
public final class CalculationUtilA {
	/**
	 * This class is a utility class and can not be instantiated.
	 */
	private CalculationUtilA() {
	}

	/**
	 * Multiply arguments x and y.
	 *
	 * @param x
	 *            argument
	 * @param y
	 *            argument
	 * @return calculation result
	 */
	public static int multiply(int x, int y) {
		return x * y;
	}
}
```

list 13. CalculationInputCalculateServletA.java

JAVA

```java
package project.web.teama;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import project.lib.CheckUtil;

/**
 * A servlet class that processes the calculate button of the calculation input
 * screen.
 */
@WebServlet("/TeamA/CalculationInputCaluculate.do")
public class CalculationInputCalculateServletA extends HttpServlet {
        private static final String ERROR_MESSAGE_KEY = "errorMessage";
        /** serialVersionUID */
        private static final long serialVersionUID = 2972265646531623810L;

        /**
         * Constructs a new instance.
         */
        public CalculationInputCalculateServletA() {
                // No operation
        }

        /**
         * Handle a POST request.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         * @throws IOException
         *             if an input or output error is detected when the servlet
         *             handles the GET request
         * @throws ServletException
         *             if the request for the GET could not be handled
         */
        @Override
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
```

```
		String formArgX = request.getParameter("argX");
		String formArgY = request.getParameter("argY");
		// You can link with the Java Library Project of DevOps Project
		// Initializer
		// by releasing the following comment out and modifying the dependencies
		// of build.gradle.
		if (CheckUtil.isNullOrEmpty(formArgX)) {
			request.setAttribute(ERROR_MESSAGE_KEY, "argumentX is required.");
			request.setAttribute("argX", formArgX);
			request.setAttribute("argY", formArgY);
			forwardToInputPage(request, response);
			return;
		}
		if (CheckUtil.isNullOrEmpty(formArgY)) {
			request.setAttribute(ERROR_MESSAGE_KEY, "argumentY is required.");
			request.setAttribute("argX", formArgX);
			request.setAttribute("argY", formArgY);
			forwardToInputPage(request, response);
			return;
		}

		int argX = 0;
		int argY = 0;
		if (CheckUtil.isNumber(formArgX)) {
			try {
				argX = Integer.parseInt(formArgX);
			} catch (NumberFormatException e) {
				// cannot be reached
			}
		} else {
			request.setAttribute(ERROR_MESSAGE_KEY, "argumentX must be numeric.");
			request.setAttribute("argX", formArgX);
			request.setAttribute("argY", formArgY);
			forwardToInputPage(request, response);
			return;
		}
		if (CheckUtil.isNumber(formArgY)) {
			try {
				argY = Integer.parseInt(formArgY);
			} catch (NumberFormatException e) {
				// cannot be reached
			}
		} else {
			request.setAttribute(ERROR_MESSAGE_KEY, "argumentY must be numeric.");
			request.setAttribute("argX", formArgX);
			request.setAttribute("argY", formArgY);
			forwardToInputPage(request, response);
			return;
```

```
                }

                int result = CalculationUtilA.multiply(argX, argY);
                request.setAttribute("argX", argX);
                request.setAttribute("argY", argY);
                request.setAttribute("result", result);
                forwardToResultPage(request, response);
        }

        /**
         * Forward the request to Calculation Input page.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         */
        private void forwardToInputPage(HttpServletRequest request, HttpServletResponse response) {
                try {
                        request.getRequestDispatcher("/WEB-INF/jsp/teama/CalculationInputA.jsp").forward(request, response);
                } catch (ServletException | IOException e) {
                        e.printStackTrace();
                }
        }

        /**
         * Forward the request to Calculation Result page.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         */
        private void forwardToResultPage(HttpServletRequest request, HttpServletResponse response) {
                try {
                        request.getRequestDispatcher("/WEB-INF/jsp/teama/CalculationResultA.jsp").forward(request, response);
                } catch (ServletException | IOException e) {
                        e.printStackTrace();
                }
        }
}
```

list 14. CalculationInputCreateServletA.java

JAVA

```java
package project.web.teama;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * A servlet class that initializes the calculation input screen.
 */
@WebServlet("/TeamA/CalculationInput.Create")
public class CalculationInputCreateServletA extends HttpServlet {
        /** serialVersionUID */
        private static final long serialVersionUID = 2972265646531623810L;

        /**
         * Constructs a new instance.
         */
        public CalculationInputCreateServletA() {
                // No operation
        }

        /**
         * Handle a GET request.
         *
         * @param request
         *            an {@link HttpServletRequest} object that contains the request
         *            the client has made of the servlet
         * @param response
         *            an {@link HttpServletResponse} object that contains the
         *            response the servlet sends to the client
         * @throws IOException
         *             if an input or output error is detected when the servlet
         *             handles the GET request
         * @throws ServletException
         *             if the request for the GET could not be handled
         */
        @Override
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                try {
                        request.getRequestDispatcher("/WEB-INF/jsp/teama/CalculationInputA.jsp").forward(request, response);
                } catch (ServletException | IOException e) {
                        e.printStackTrace();
                }
        }
}
```

Also, add the following 2 jsp files under the directory `src/main/webapp/WEB-INF/jsp/teama`.

list 15. CalculationInputA.jsp

JAVA

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Team A - Calculation Input</title>
        <meta charset="UTF-8">
</head>
<body>
        ${errorMessage}
        <form action="<%=request.getContextPath()%>/TeamA/CalculationInputCaluculate.do" method="POST">
                argumentX
                <input type="text" name="argX" value="${requestScope.argX}"/><br>
                argumentY
                <input type="text" name="argY" value="${requestScope.argY}"/><br>
                <input type="submit" value="Calculate">
        </form>
</body>
</html>
```

list 16. CalculationResultA.jsp

JAVA

```
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Team A - Calculation Result</title>
        <meta charset="UTF-8">
</head>
<body>
        ${requestScope.argX} * ${requestScope.argY} = ${requestScope.result}
</body>
</html>
```

Also, add the following java unit test source code under the directory `src/test/java/project/web/teama` .

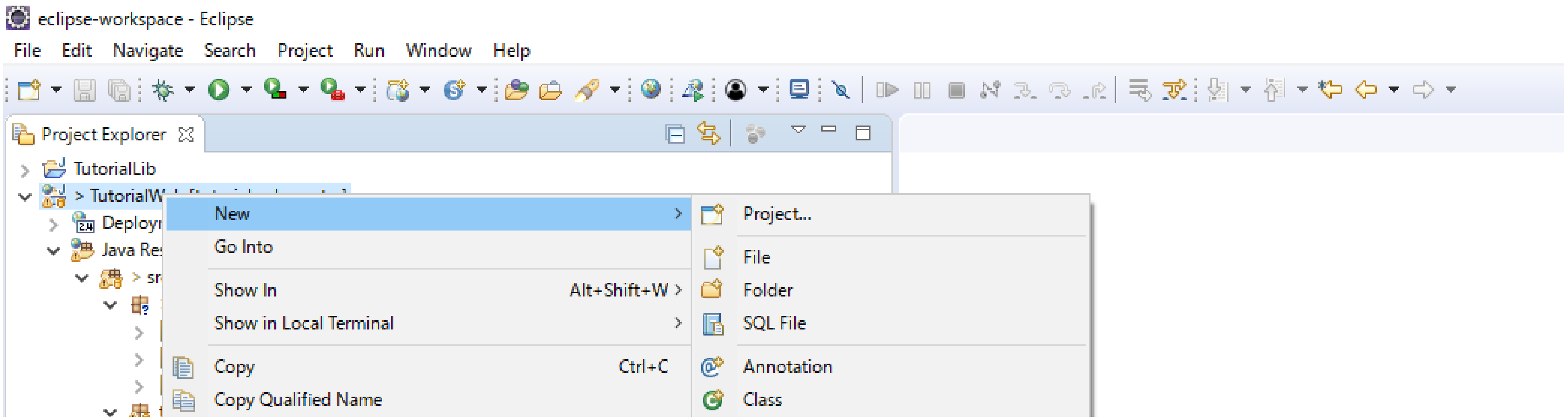
list 17. CalculationUtilTestA.java

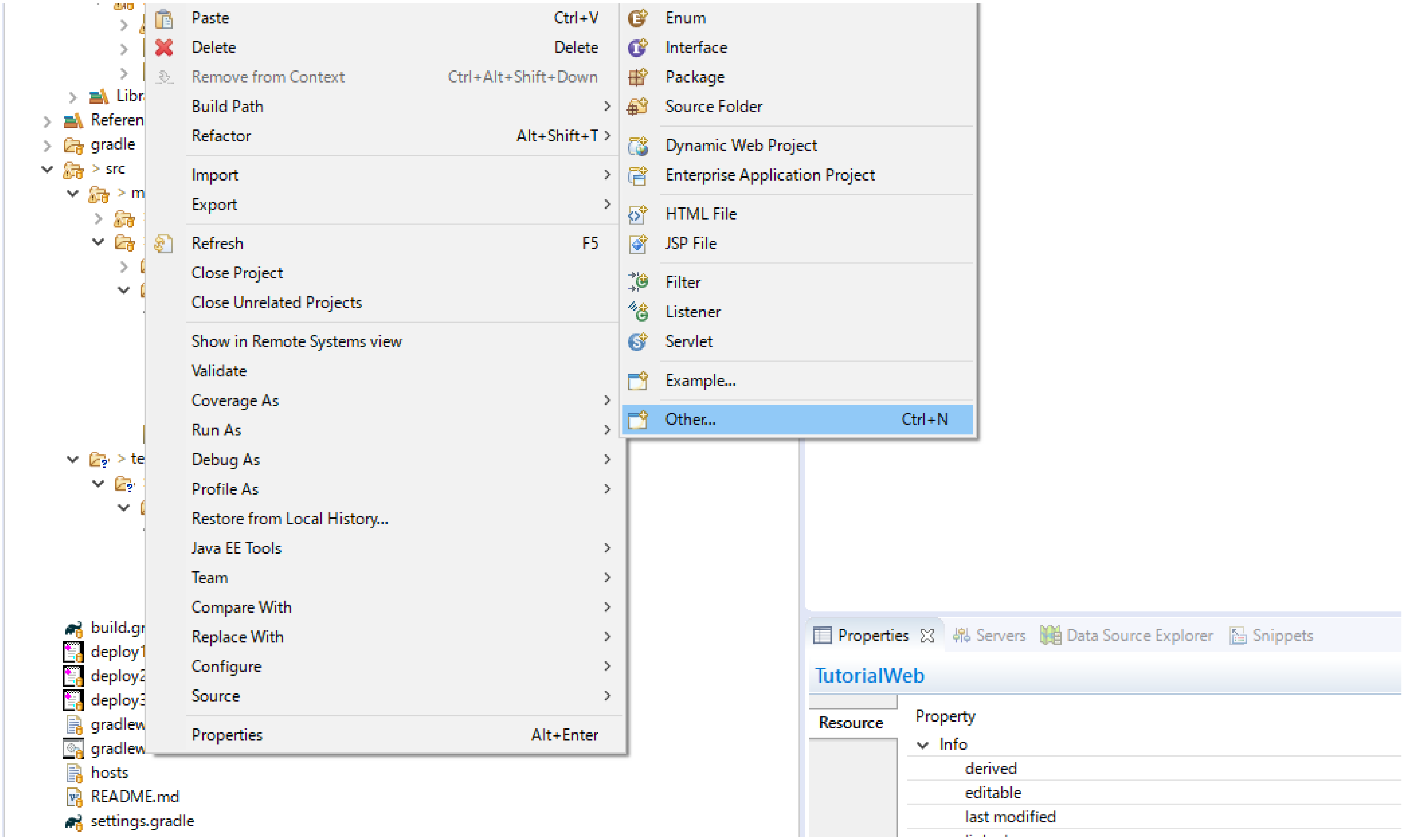
JAVA

```java
package project.web.teama;

import static org.junit.Assert.*;

import org.junit.Test;

public class CalculationUtilTestA {
        @Test
        public void testMultiply() {
                assertEquals(2, CalculationUtilA.multiply(1, 2));
        }

        @Test
        public void testMultiply_Zero() {
                assertEquals(0, CalculationUtilA.multiply(0, 0));
        }
}
```

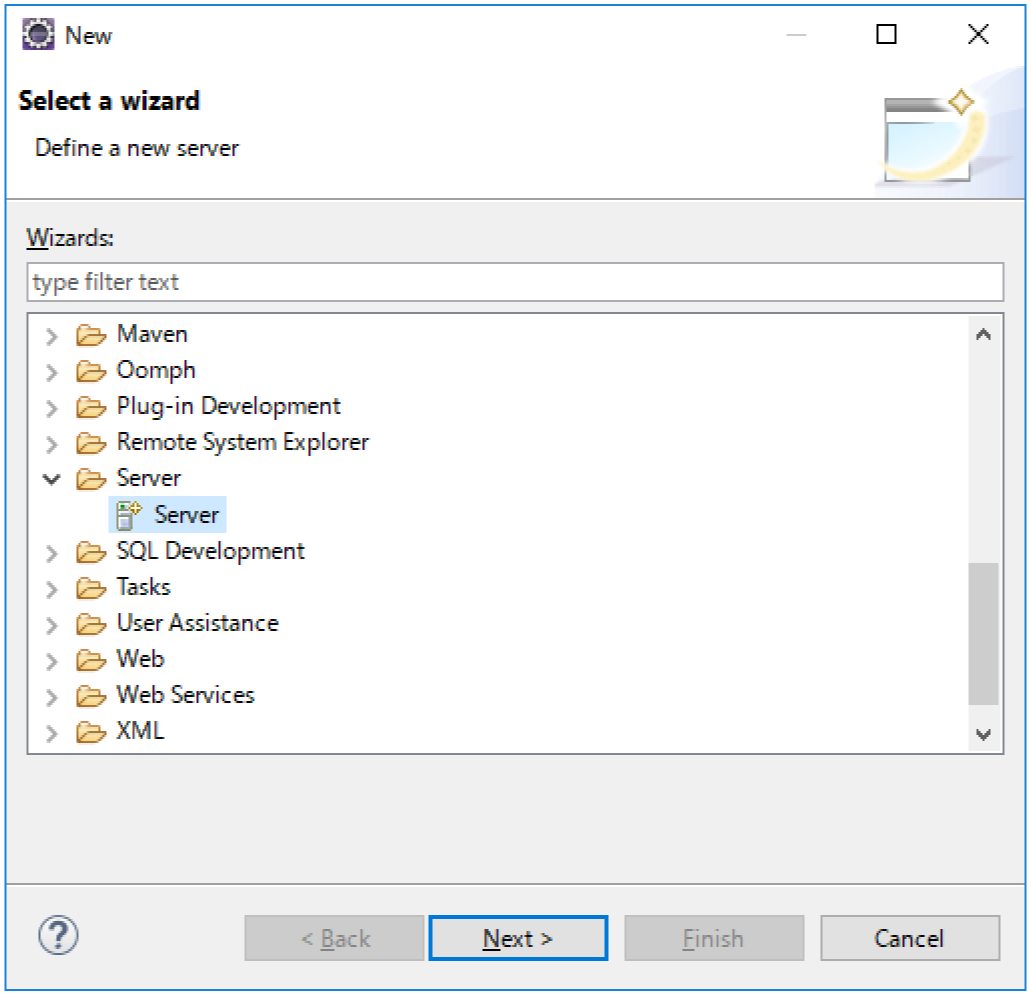
3.Check the operations of the implemented web application in the local machine.

Start Eclipse, from the menu select [File] → [New] then click [Other…] option.

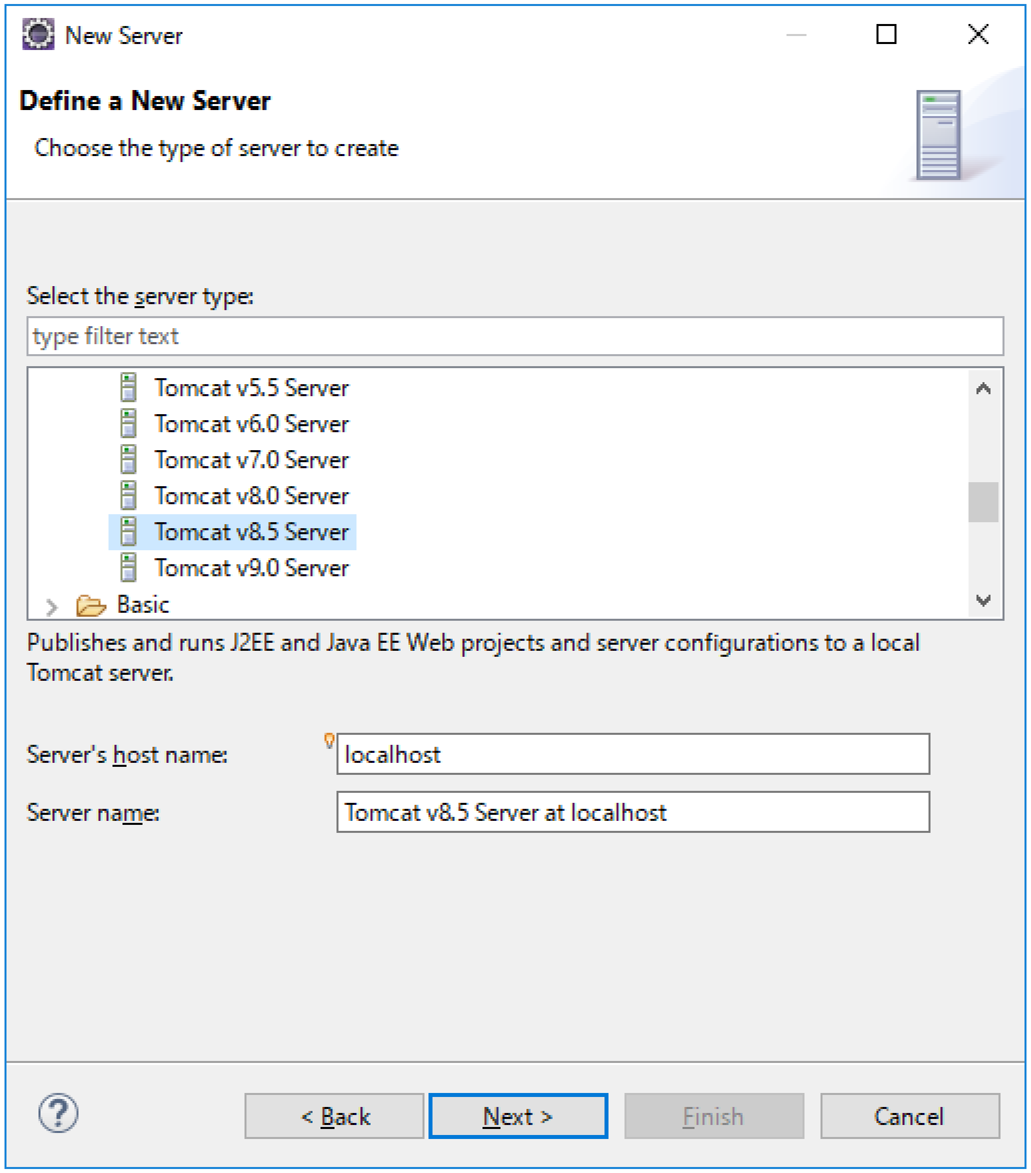




Select [Server] then click the [Next>] button.



From [Apache] directory select the version of Tomcat installed in Tomcat installation and click [Next >] button.



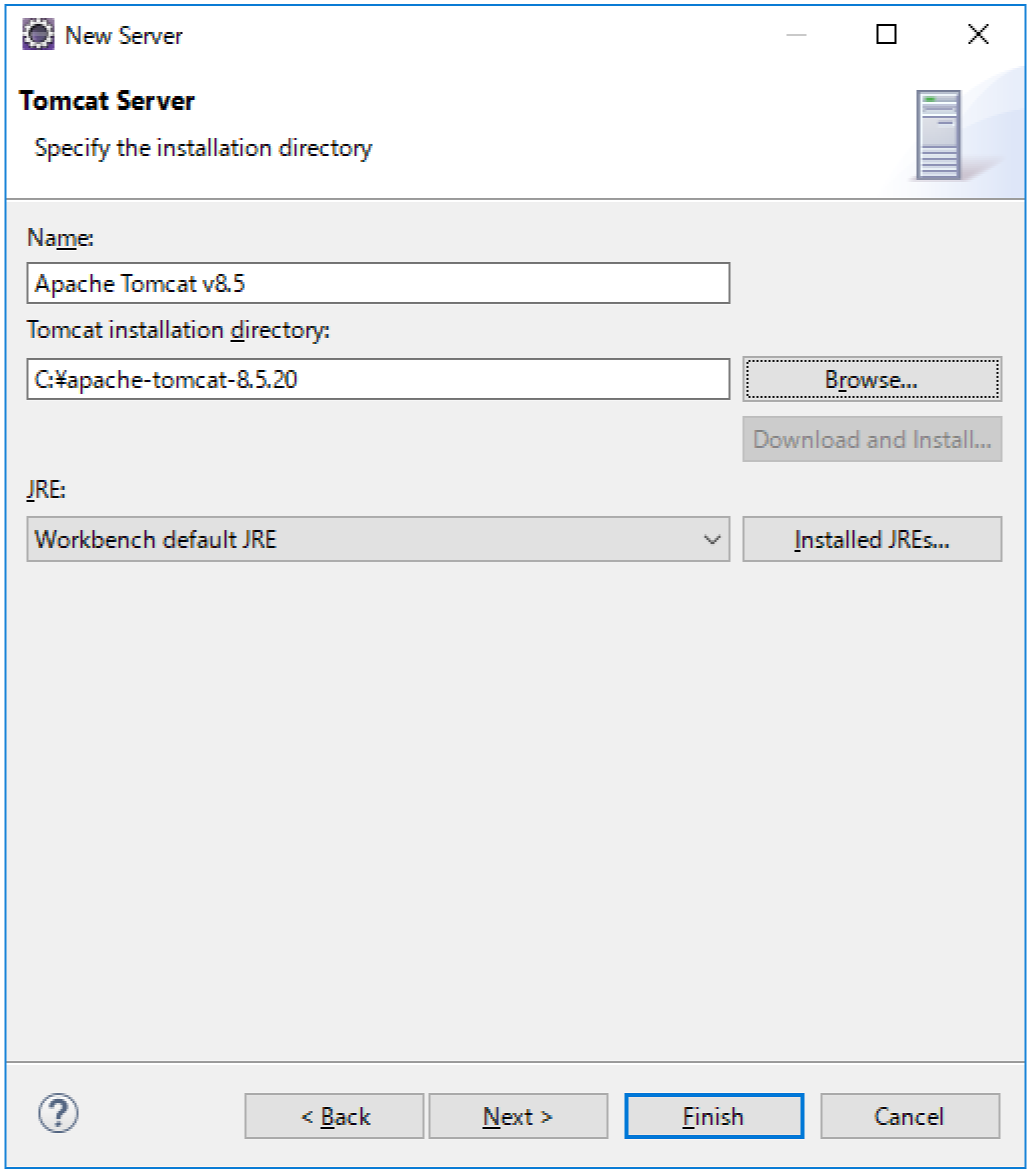
Select the server type: type filter text

Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server.

Server's host name: localhost

Server name: Tomcat v8.5 Server at localhost

In the field Tomcat installation directory, set the folder of the Tomcat installed in Tomcat installation then click [Finish] button.



Tomcat Server

Specify the installation directory

Name:

Apache Tomcat v8.5

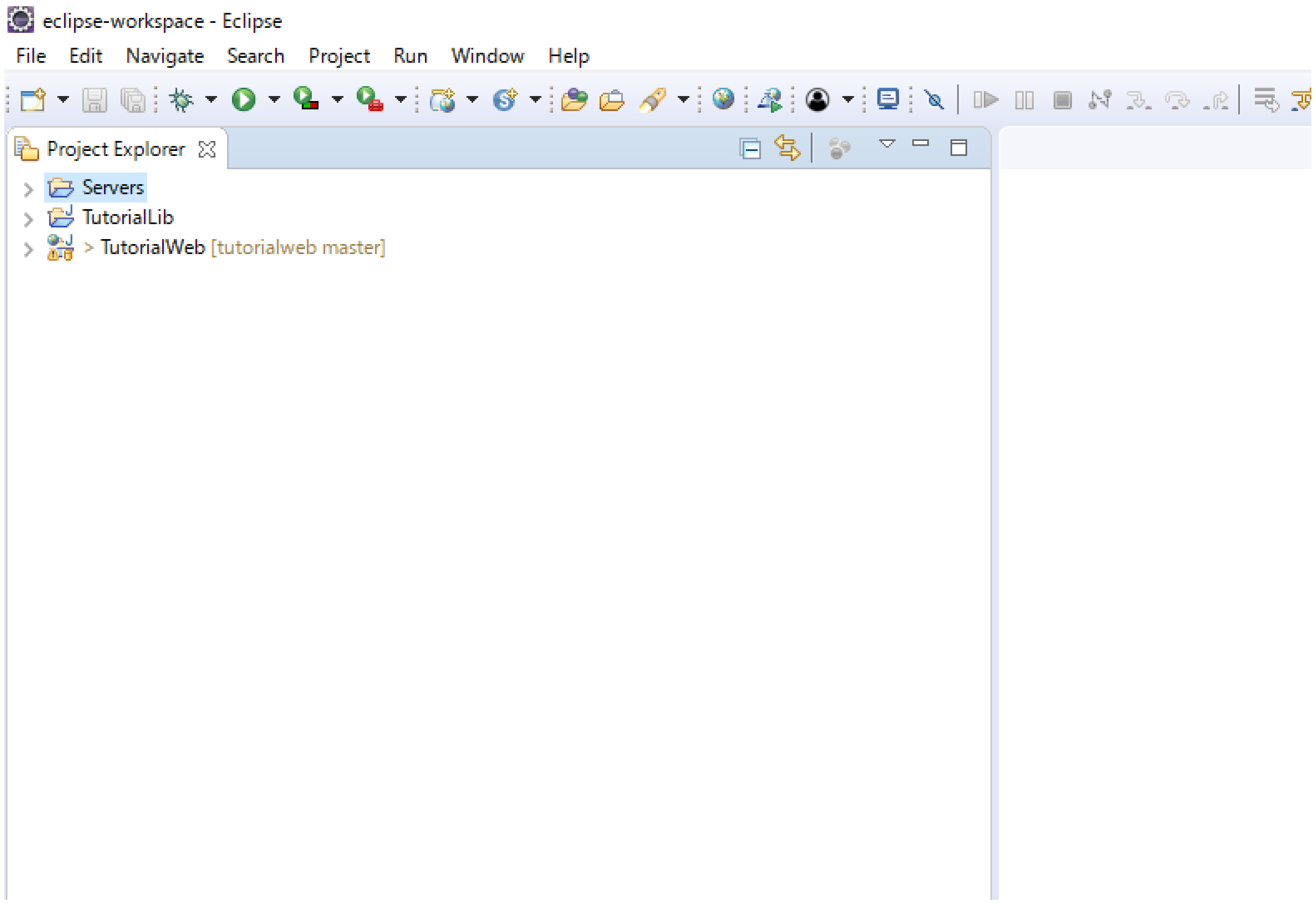
Tomcat installation directory:

C:¥apache-tomcat-8.5.20

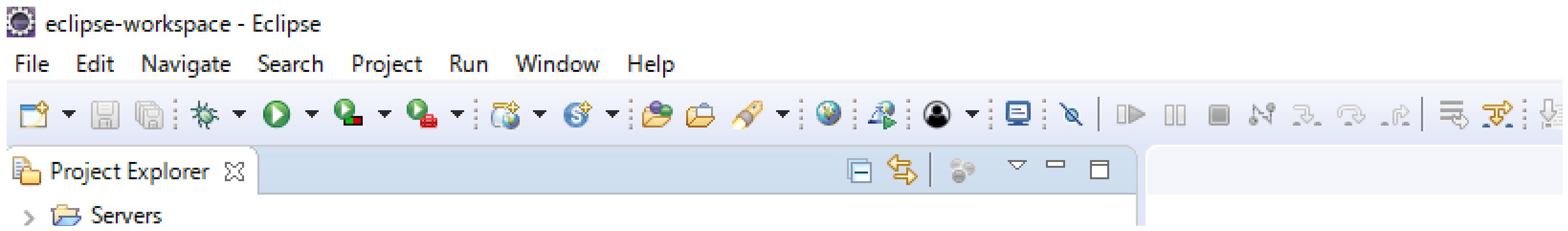
JRE:

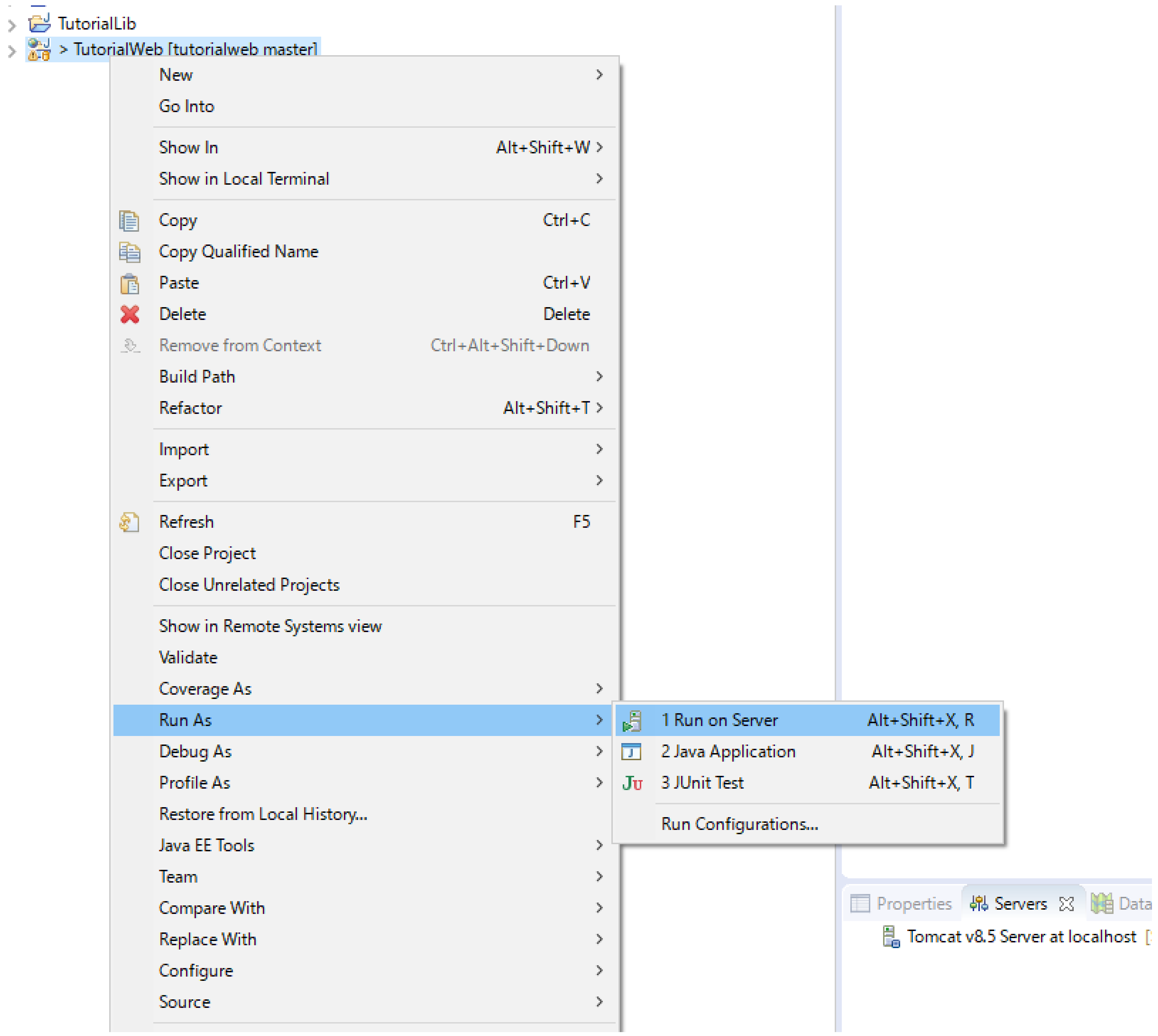
Workbench default JRE

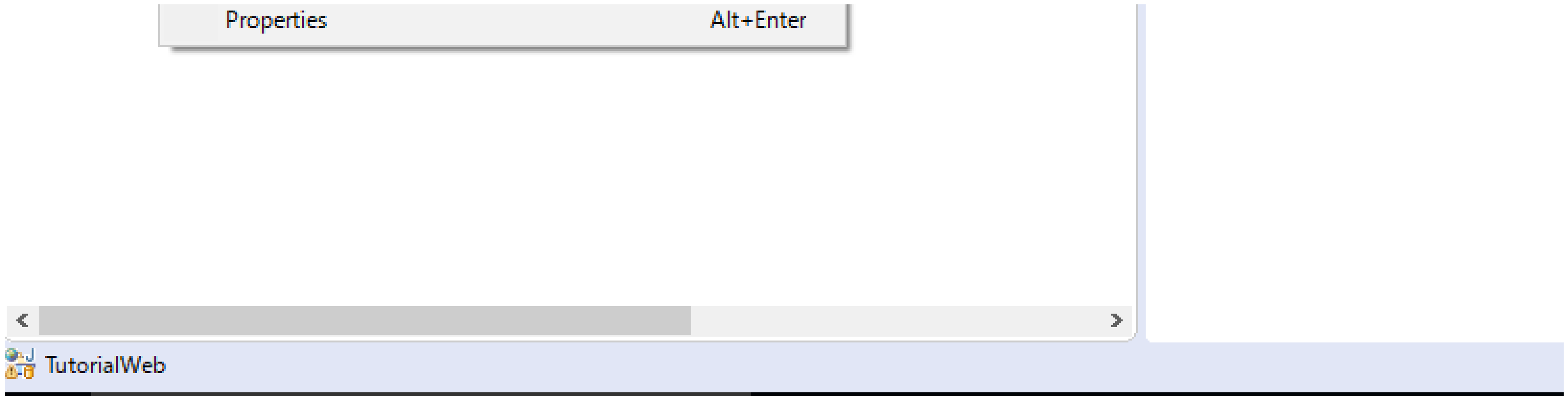
Then in the [Project Explorer] window of Eclipse, check if [Servers] is added.


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
Servers
TutorialLib
> TutorialWeb [tutorialweb master]

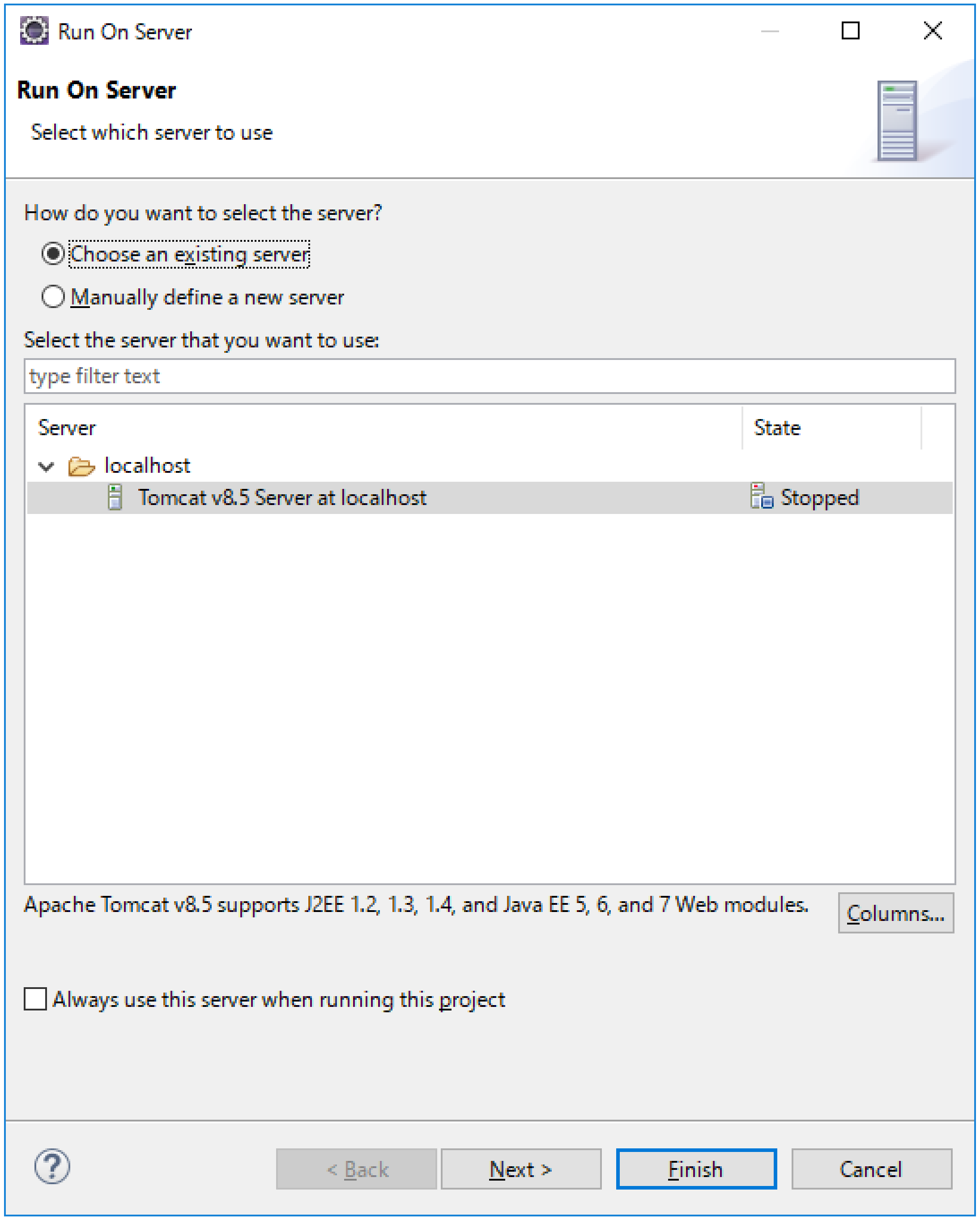
In [Project Explorer] window of Eclipse, right click [TutorialWeb] project then from the context menu, select [Run As] → [Run on Server] option.


eclipse-workspace - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
Servers


TutorialLib
> TutorialWeb [tutorialweb master]
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import
Export
Refresh F5
Close Project
Close Unrelated Projects
Show in Remote Systems view
Validate
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Java EE Tools
Team
Compare With
Replace With
Configure
Source
1 Run on Server Alt+Shift+X, R
2 Java Application Alt+Shift+X, J
3 JUnit Test Alt+Shift+X, T
Run Configurations...
Properties
Servers
Data
Tomcat v8.5 Server at localhost



Select the option [Choose an existing server] and select the Tomcat server and click [Finish] button.



Run On Server

Run On Server

Select which server to use

How do you want to select the server?

[✓ Choose an existing server]

Manually define a new server

Select the server that you want to use:

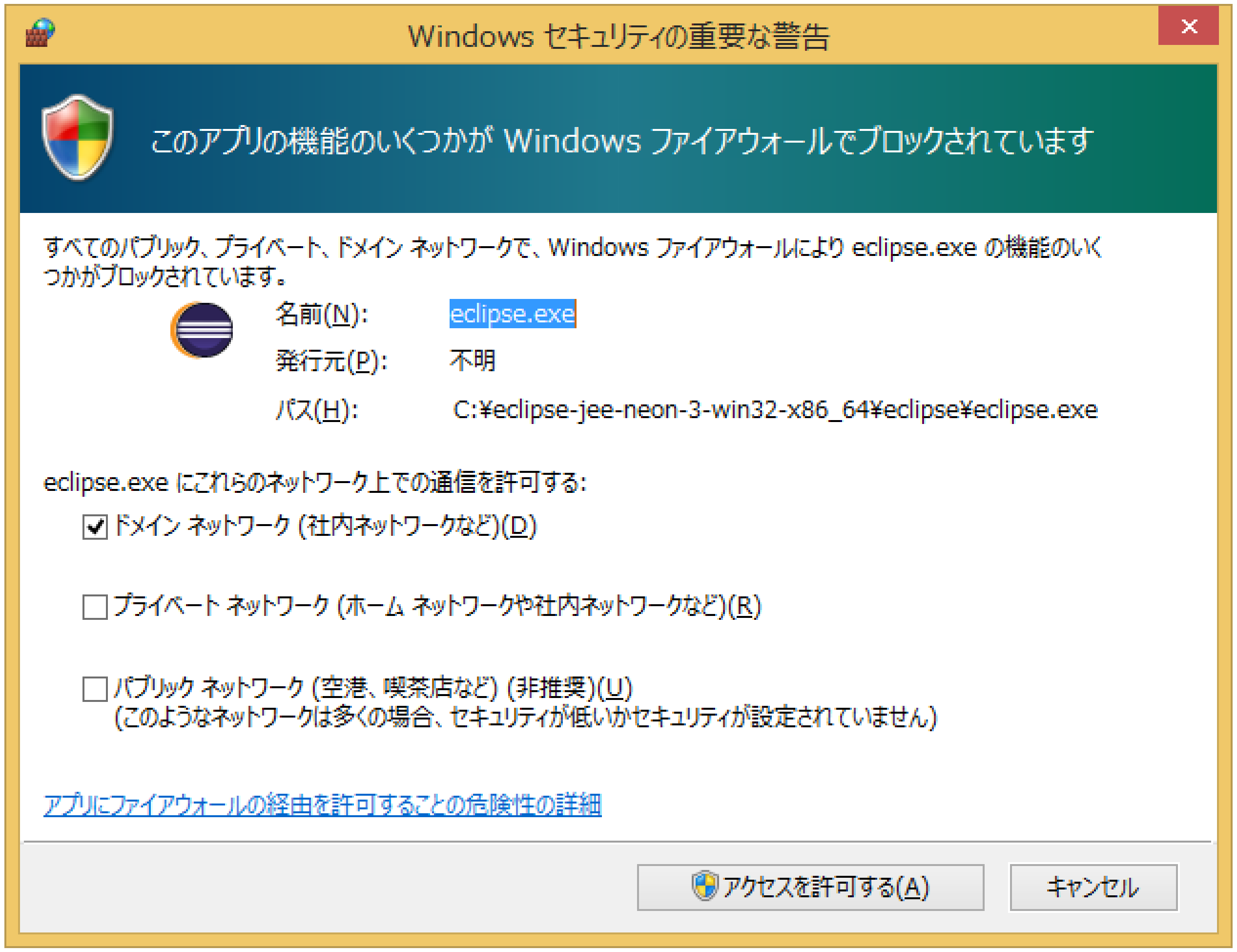
type filter text

Server	State
localhost	
Tomcat v8.5 Server at localhost	Stopped

Apache Tomcat v8.5 supports J2EE 1.2, 1.3, 1.4, and Java EE 5, 6, and 7 Web modules.

☐ Always use this server when running this project

Only during at the initial startup of Tomcat server will the security warning dialog box appear. From the dialog box, click the [Allow access] button.


Windows セキュリティの重要な警告

このアプリの機能のいくつかが Windows ファイアウォールでブロックされています

すべてのパブリック、プライベート、ドメイン ネットワークで、Windows ファイアウォールにより eclipse.exe の機能のいくつかがブロックされています。

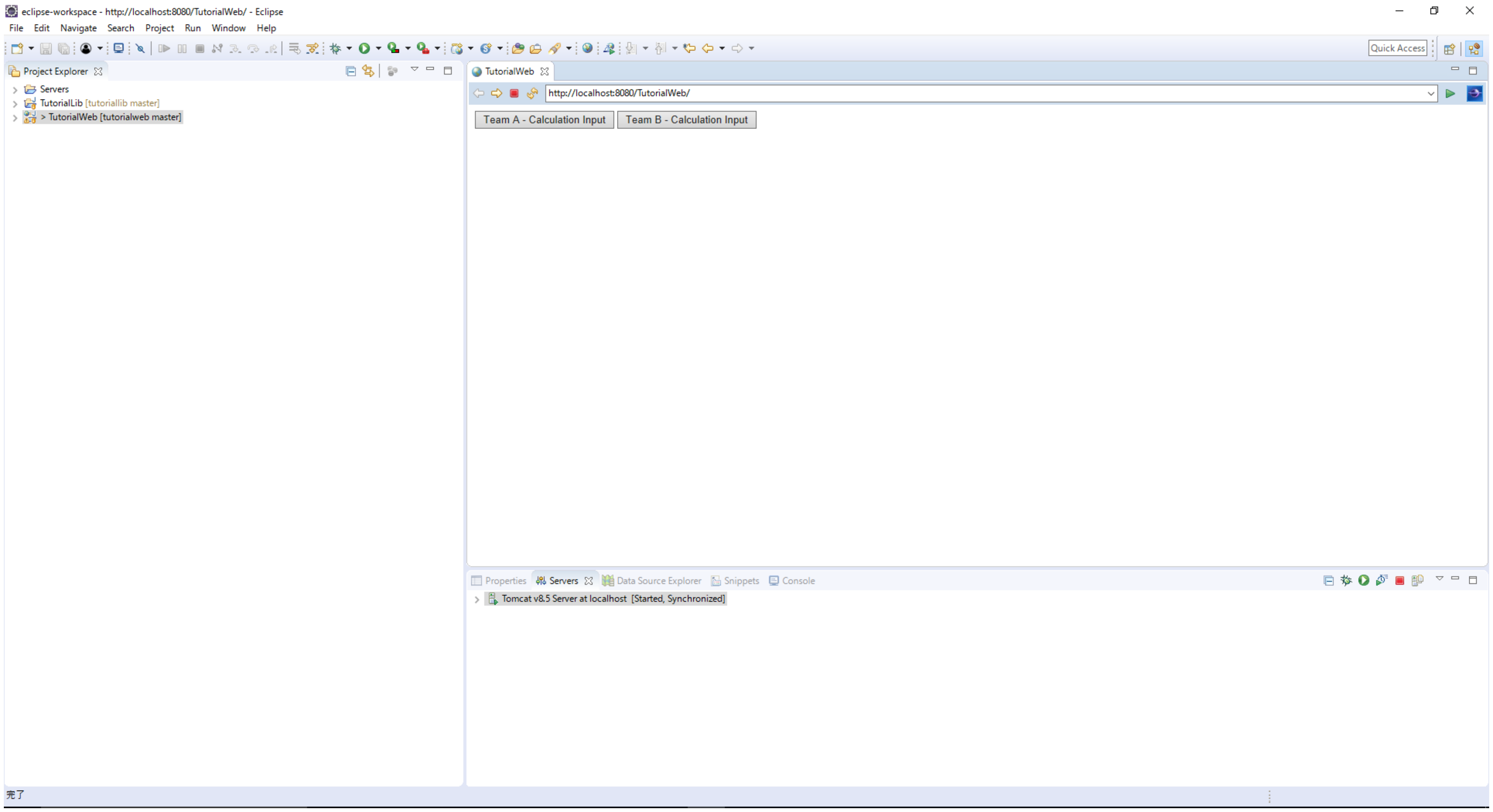
名前(N): eclipse.exe
発行元(P): 不明
パス(H): C:¥eclipse-jee-neon-3-win32-x86_64¥eclipse¥eclipse.exe

eclipse.exe にこれらのネットワーク上での通信を許可する:

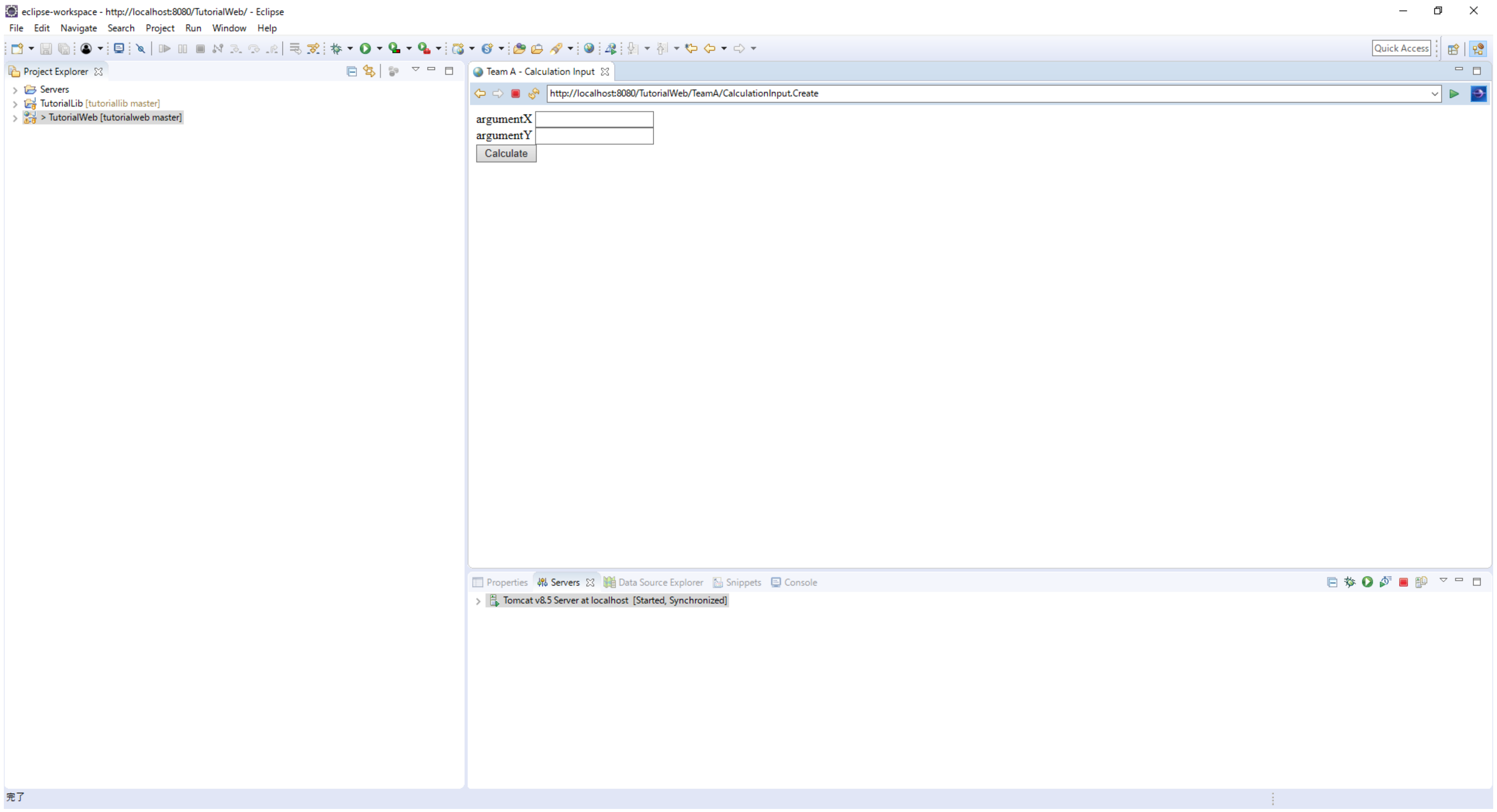
- [✓] ドメイン ネットワーク (社内ネットワークなど)(D)
- [ ] プライベート ネットワーク (ホーム ネットワークや社内ネットワークなど)(R)
- [ ] パブリック ネットワーク (空港、喫茶店など) (非推奨)(U)
  (このようなネットワークは多くの場合、セキュリティが低いかセキュリティが設定されていません)

アプリにファイアウォールの経由を許可することの危険性の詳細

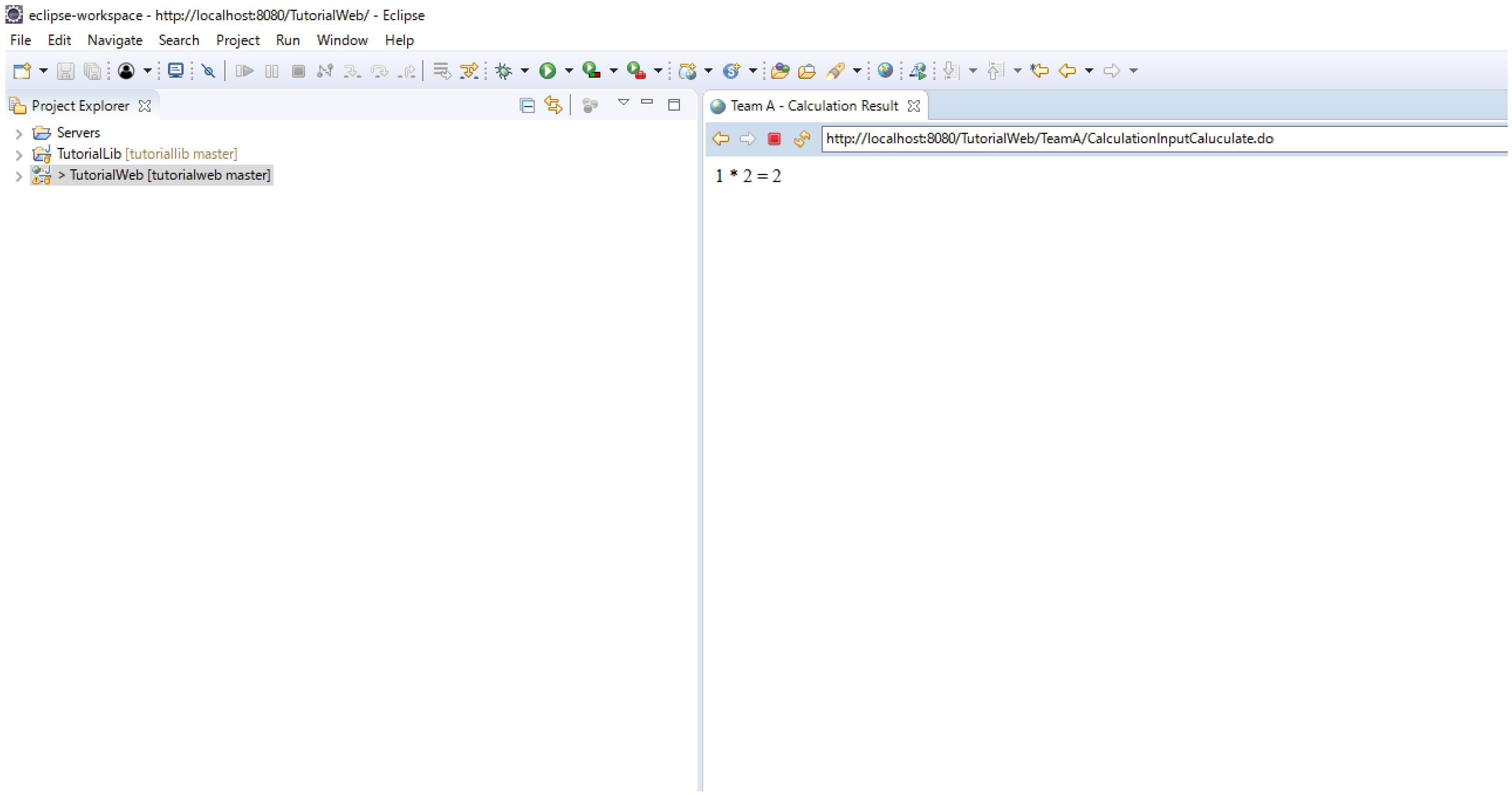
The [Calculation Input] tab will be opened, and the screen as shown below will be displayed.


eclipse-workspace - http://localhost:8080/TutorialWeb/ - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Project Explorer
Servers
TutorialLib [tutoriallib master]
> TutorialWeb [tutorialweb master]
TutorialWeb
http://localhost:8080/TutorialWeb/
Team A - Calculation Input
Team B - Calculation Input
Properties
Servers
Data Source Explorer
Snippets
Console
Tomcat v8.5 Server at localhost [Started, Synchronized]
完了

When the button [Team A - Calculation Input] is clicked, the screen as shown below will be displayed.


eclipse-workspace - http://localhost:8080/TutorialWeb/ - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Project Explorer
Servers
TutorialLib [tutoriallib master]
> TutorialWeb [tutorialweb master]
Team A - Calculation Input
http://localhost:8080/TutorialWeb/TeamA/CalculationInput.Create
argumentX
argumentY
Calculate
Properties
Servers
Data Source Explorer
Snippets
Console
Tomcat v8.5 Server at localhost [Started, Synchronized]
完了

In the fields of argumentX and argumentY, input any random number and click the [Calculate] button. Check that the result displayed is the product of the numbers inputted.


eclipse-workspace - http://localhost:8080/TutorialWeb/ - Eclipse
File Edit Navigate Search Project Run Window Help
Project Explorer
Servers
TutorialLib [tutoriallib master]
> TutorialWeb [tutorialweb master]
Team A - Calculation Result
http://localhost:8080/TutorialWeb/TeamA/CalculationInputCaluculate.do

1 * 2 = 2

4.[Commit & Push] the implemented source code and reflect all the changes in the repository. From the context menu of [TortoiseGit], click [Git Commit] and input teh desciption of commit then, click [Commit & Push] button (can be selected from the pull-down menu of Commit button).

Item Field	Value
Message	Add A team implementation

Item Field	Value
Changes made (double-click the file for diff)	Put a check mark in all the files

When username and password are required

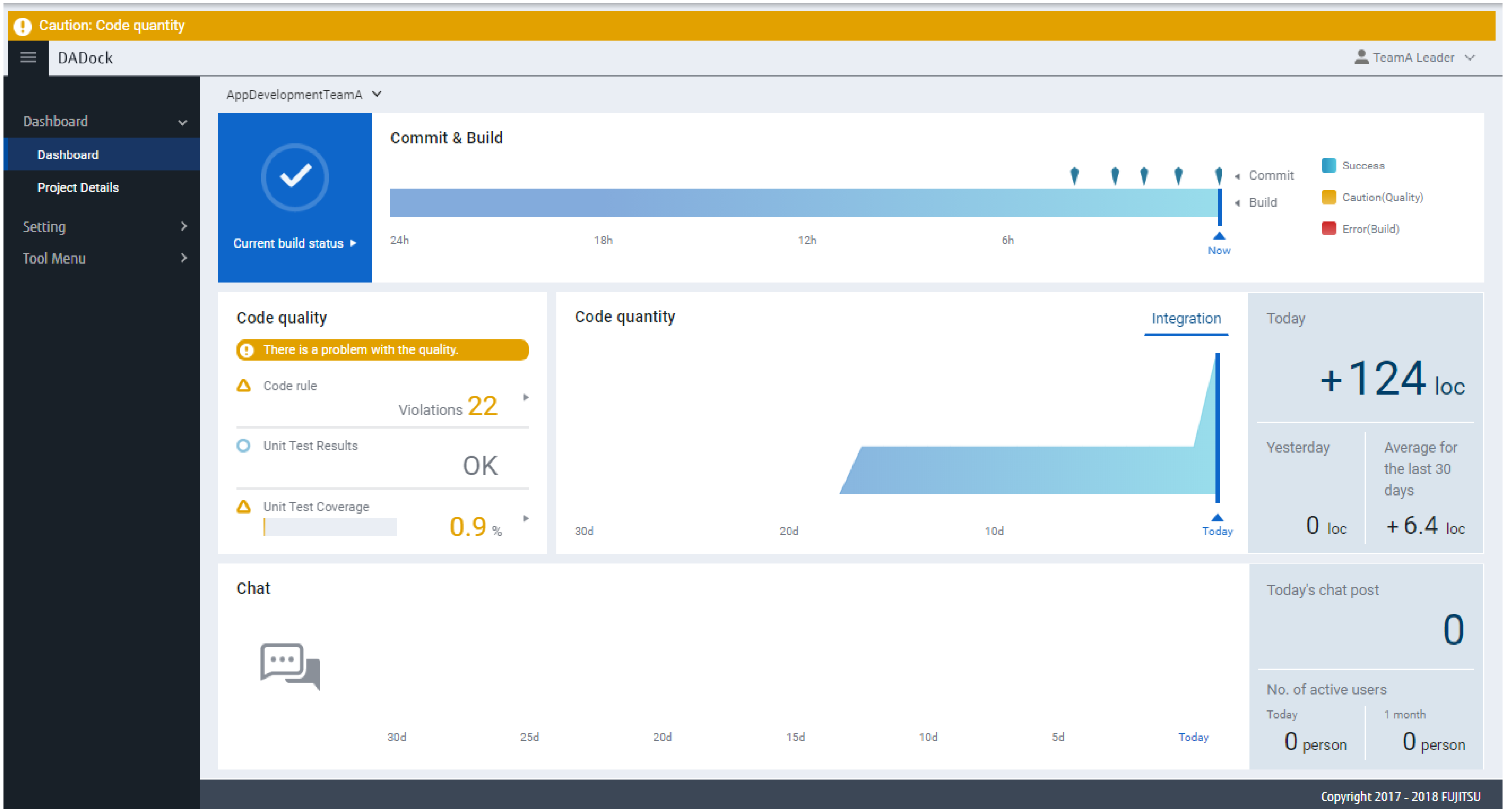
When the username and password of TortoiseGit are required, input the user information of [app.a0].

5.2.3. Source Code Quality Check

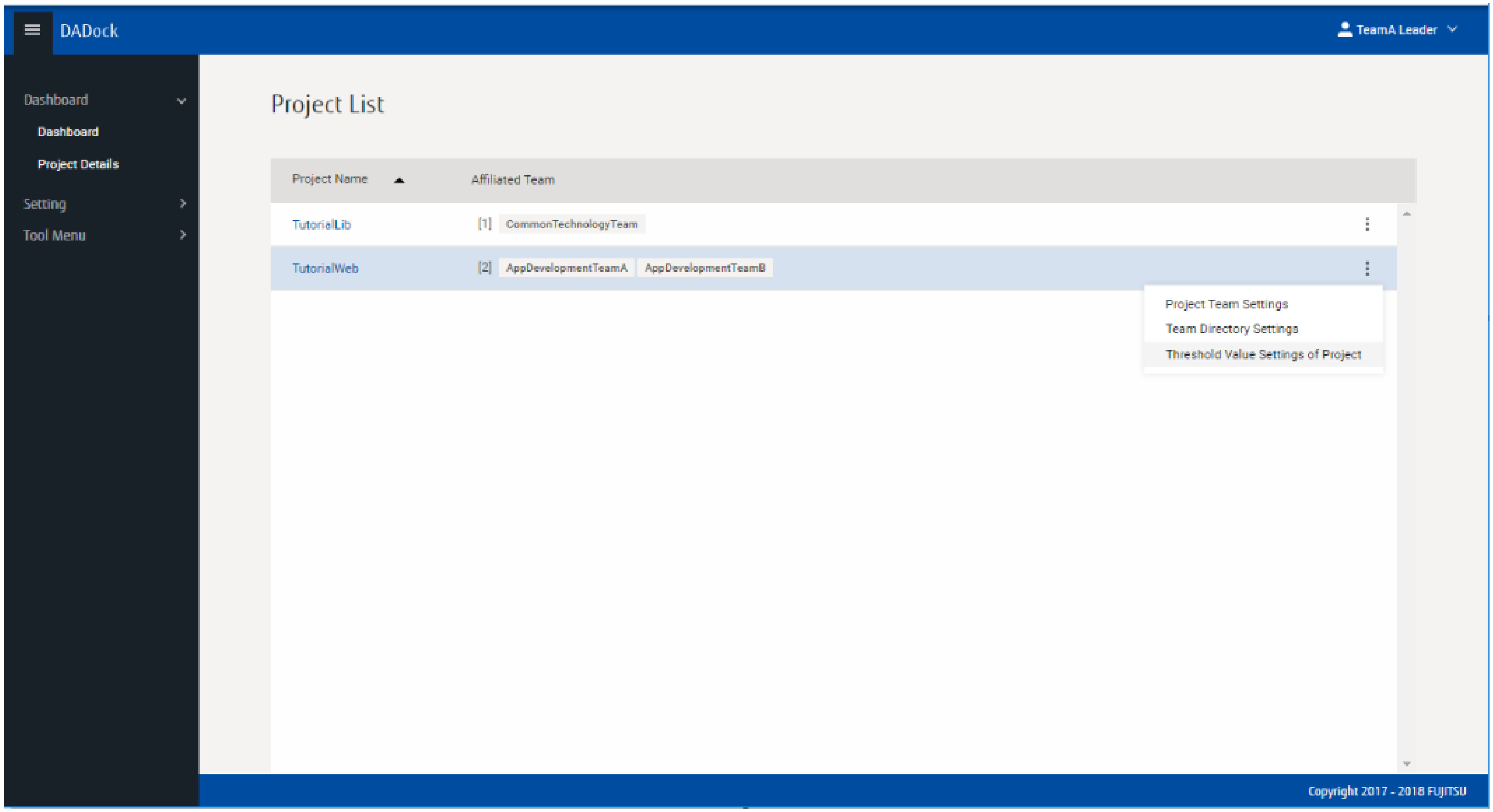
The development leader of team A will check the quality of the source code.

1.Access DADock Portal and login as [app.a0] user.

2.At the Dashboard Page of DADock Portal, check the displayed values of the code violation in the source code static analysis.

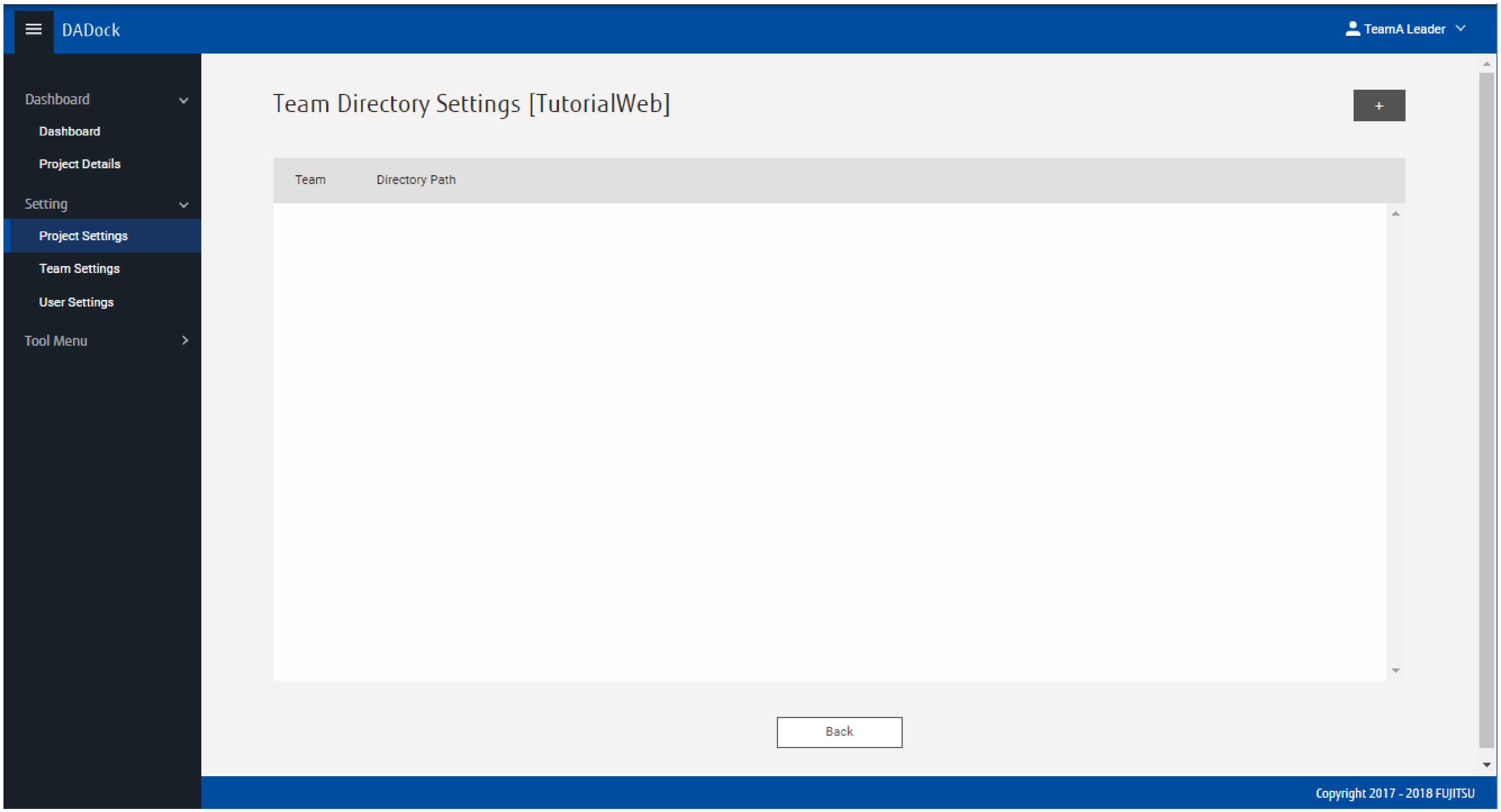


3.In order to check the values of the code violations implemented only by team A, from the menu bar expand the [Settings] tab and click [Project Settings] to display the Peoject List Page.+



Project Name	Affiliated Team
TutorialLib	[1] CommonTechnologyTeam
TutorialWeb	[2] AppDevelopmentTeamA AppDevelopmentTeamB

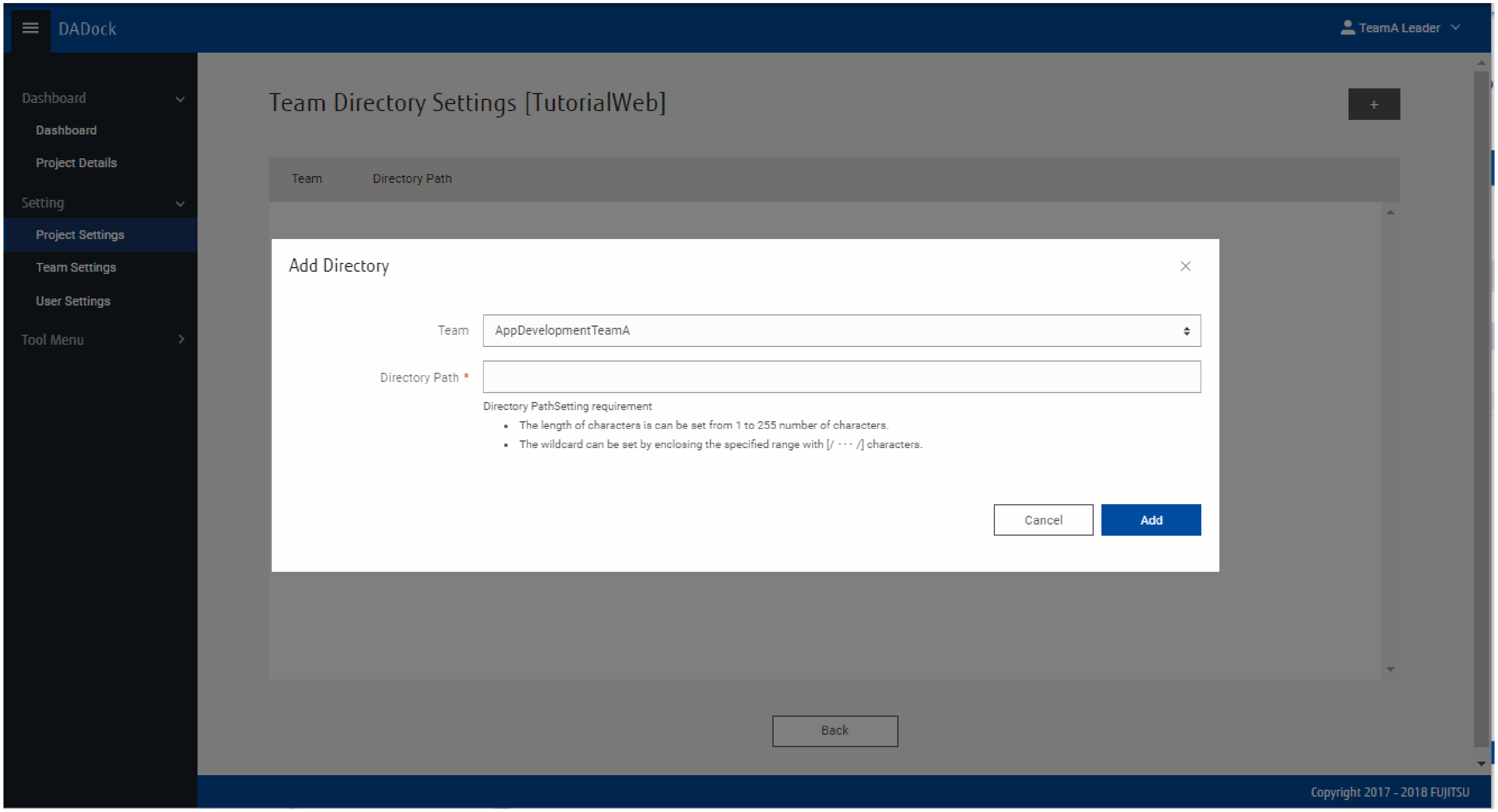
4.In order to check the values of the code violations implemented only by team A, at the right part of [TutorialWeb] project, click the context menu and select [Team Directory Settings] to display the Team Directory Settings page.



Team Directory Settings [TutorialWeb]

Team	Directory Path

5.Check if the Team Directory Settings page is displayed, click the [+] button at the upper right side of the page.



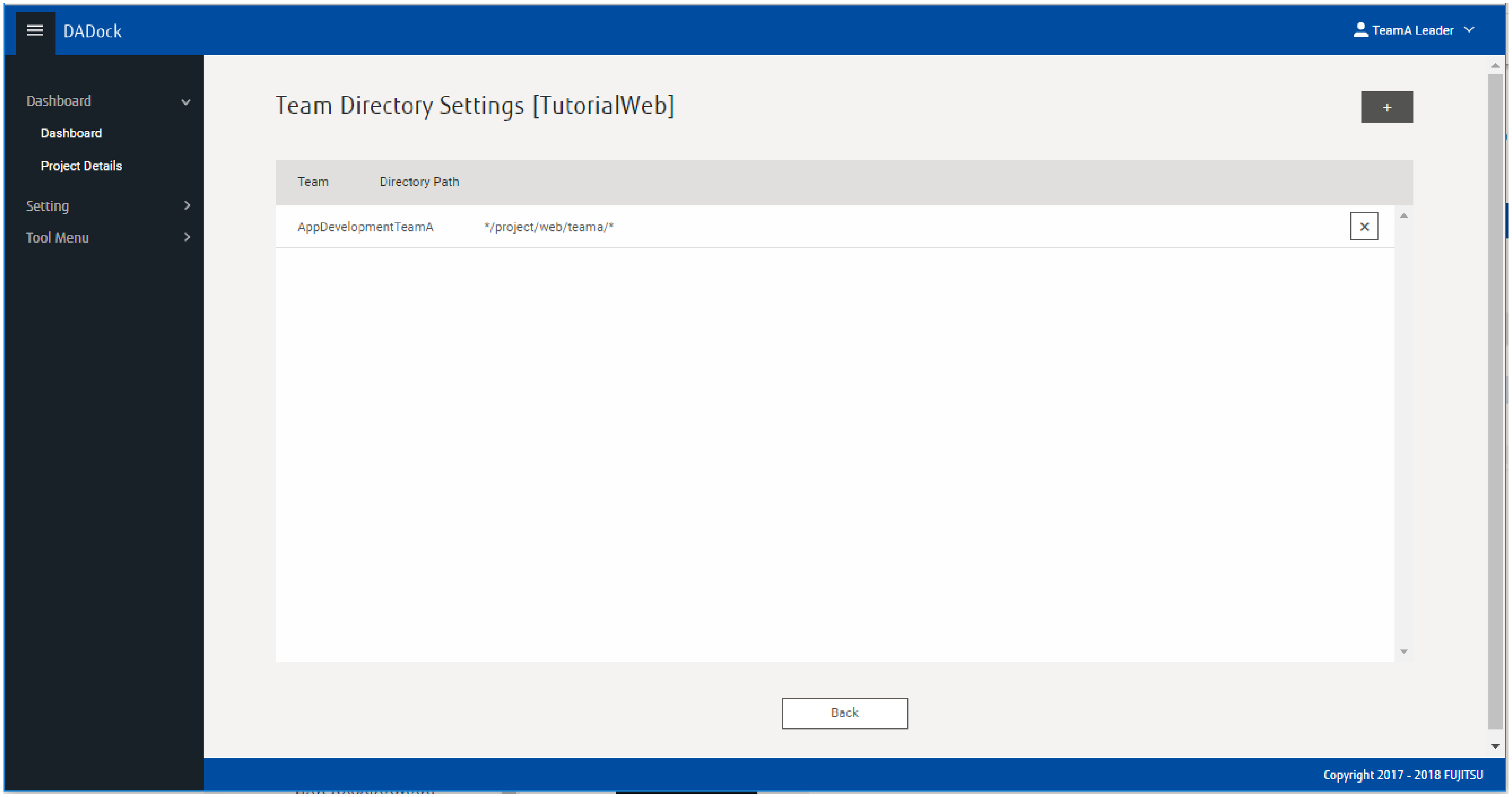
Directory PathSetting requirement

- The length of characters is can be set from 1 to 255 number of characters.
- The wildcard can be set by enclosing the specified range with [/ ··· /] characters.

6.Check that the Add Directory pop up page is displayed, then input the following information to add the team directory settings and then click the [Add] button.

Item	**Value**
Team	AppDevelopmentTeamA

Item	Value
Directory Path	*/project/web/teama/*

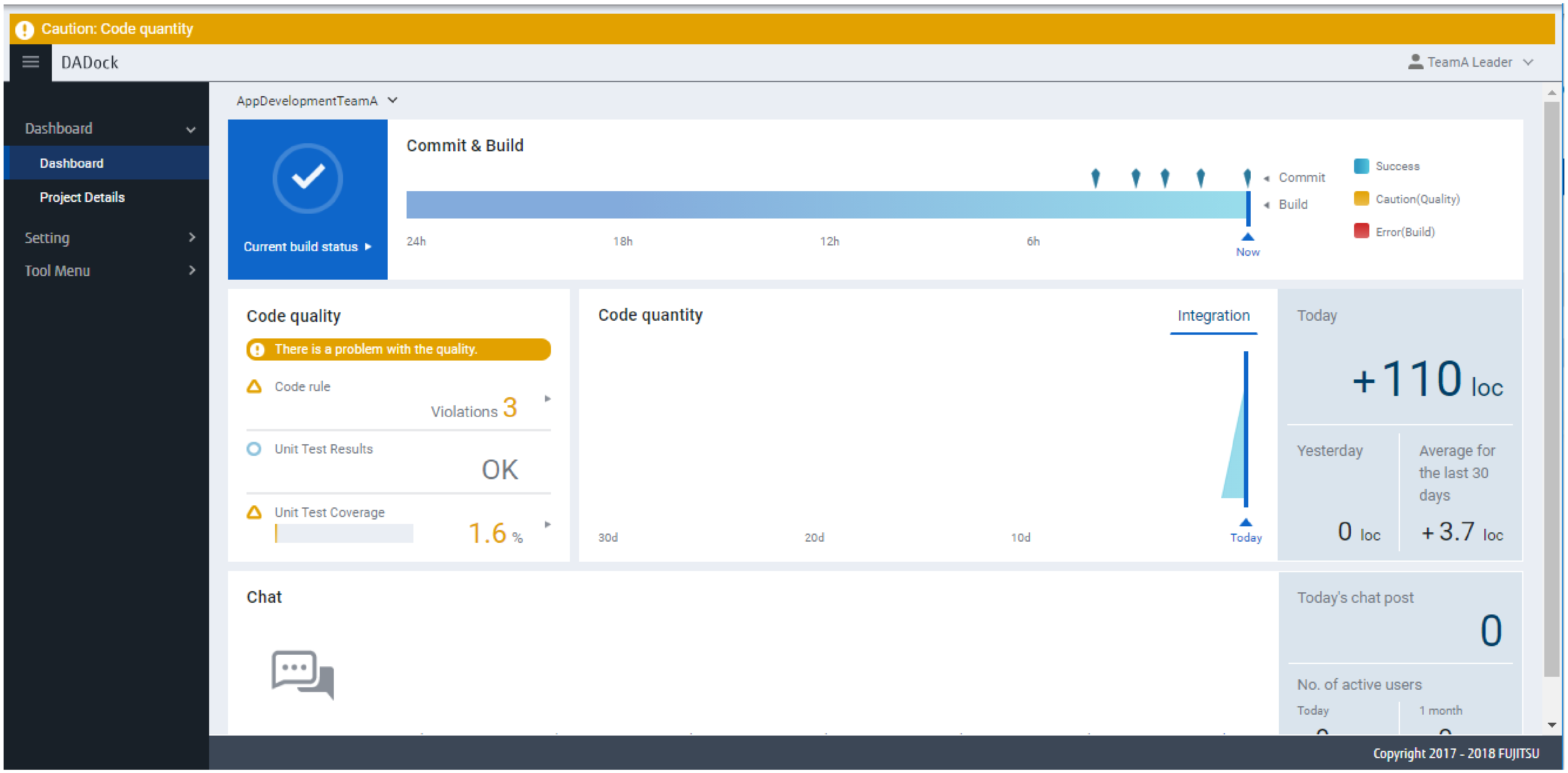


Team Directory Settings [TutorialWeb]

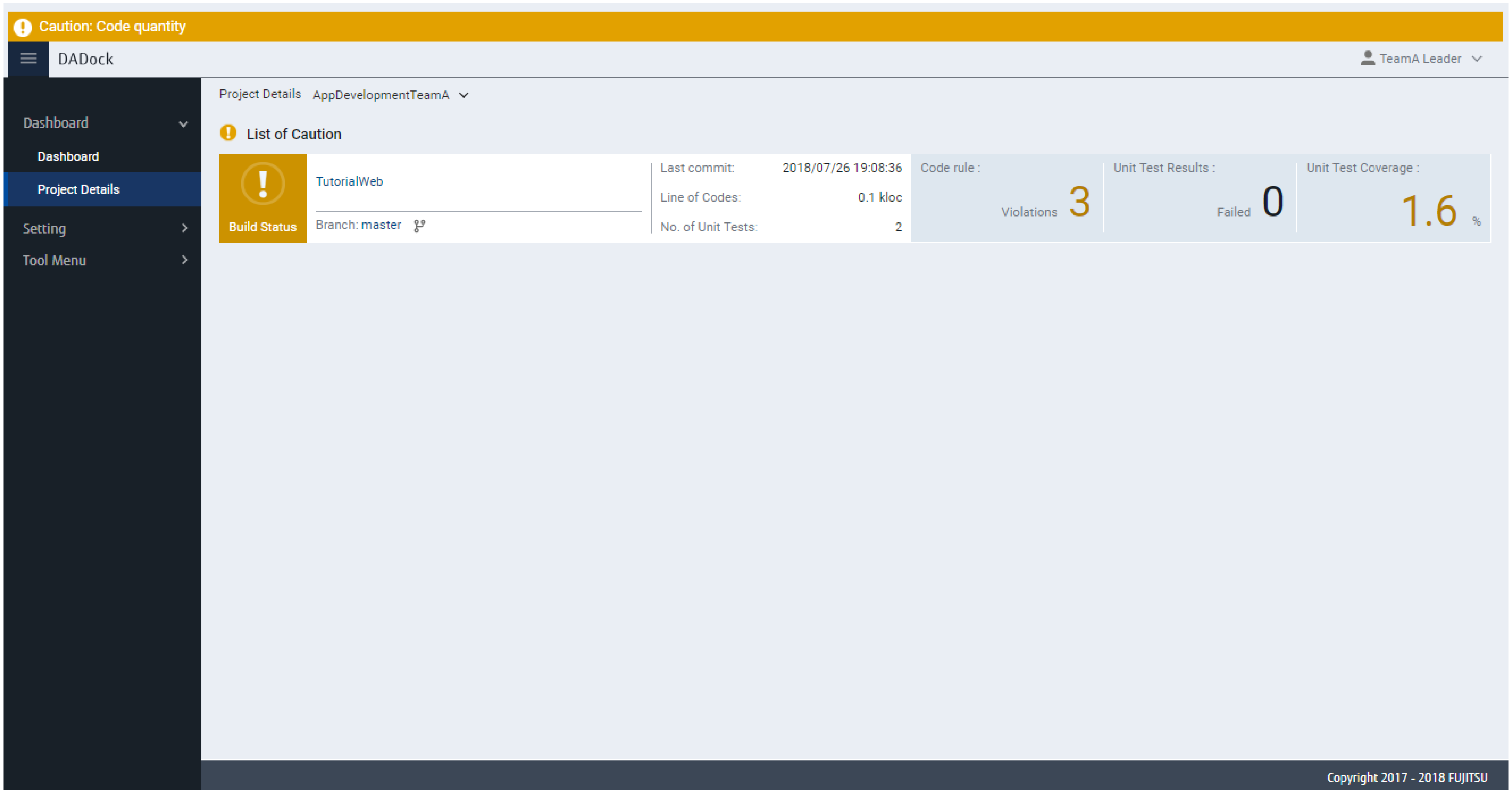
Team	Directory Path
AppDevelopmentTeamA	*/project/web/teama/*

7.Check that there is 1 entry added in the Team Directory Settings Page.

8.Go back to the Dashboard Page and check that the displayed values in the code violations in the static analysis decreased as compared to before adding the Team Directory conditions.



9.Go to the Project Details Page, and the same with the Dashboard Page, check that the displayed values in the code violations in the static analysis decreased as compared to before adding the Team Directory conditions



By adding the Team Directory settings, it is possible that only the quality status of team A can be checked.

This is the end of Tutorial.

6. Appendix

6.1. How to install various tools

This chapter explains on how to install the various tools used in this tutorial.

6.1.1. Git installation

From the home page of Git for Windows, download the installer and install the software.

6.1.2. TortoiseGit installation

From the official home page of TortoiseGit, download the installer and install the software. During installation, the username and password will be required so input the following information.

Item Field	Value
git user name	The characters before the @ mark of the email address. Example) For the email address, fujitsu_taro@jp.fujitsu.com, The username will be [fujitsu_taro]
git e-mail address	Email address

In case development is done in a proxy environment

Even if the development is done in a proxy environment, the proxy setting for TortoiseGit is not needed anynmore since DADock itself is operating in a proxy environment. On the opposite, if proxy settings is done in TortoiseGit, DADock can no longer be accessed so it is a must to be careful with this setting.

6.1.3. JDK8 installation

JDK8 (Java SE Development Kit 8) is an environment for Java development and execution. From the home page of Oracle, download the installer and install the software. After the installation, add the following settings in the environment variables.

Environment variable name	Value
JAVA_HOME	Set the file path of the installation folder of JDK8
PATH	Set the file path of bin folder under the installation folder of JDK8

6.1.4. Eclipse and other related plugin installation

From the home page of Eclipse, download the zip file of [Eclipse IDE for Java EE Developers] installer and install the software in any random directory. After installing, start the Eclipse software.

In case development is done in a proxy environment

In case the development is done with an internet connection to the proxy server, the proxy settings of Eclipse is needed so that internet can be accessed when the plugin is being downloaded. In case development is done in a proxy environment, the proxy setting of Eclipse is also needed.

Details of Settings

Refer to Eclipse proxy settings

6.1.5. Gradle settings

Since the project generated in the Project Create page uses Gradle Wrapper, there is no need to install Gradle. However, if the development is done in proxy environment, it is needed to set the proxy settings of Gradle.

In case development is done in a proxy environment

In case the development is done via internet connection to the proxy server, the proxy setting of Gradle is needed so that internet can be accessed. In case development is done in a proxy environment, the proxy setting of Gradle is also needed to set.

Details of Settings

Refer to Gradle proxy setting

6.1.6. Tomcat installation

To operate Tomcat, it is a must that Java is already installed in the local machine. Java can be installed by executing the JDK8 installation.

From the home page of Apache Tomcat, download the zip file of the installer and extract it. It is recommended that the extraction folder is under C drive.

6.2. Proxy settings for the various tools

6.2.1. Eclipse proxy setting

In Eclipse application, go to [Window] menu, open [Preference] → [General] → [Network Connections] and add the appropriate settings.

To check that the settings are properly set, restart Eclipse, go to [Window] menu, select [Help] then click [Eclipse Marketplace]. In case Eclipse Marketplace window is not displayed, check again the settings of Eclipse.

6.2.2. Gradle proxy setting

In case build is done in the development environment, Gradle and Artifactory will download the libraries from the internet and resolve the existing dependencies. In accordance to that, proxy settings for Gradle is needed.

Add the following entries in gradle.properties file. (Content of gradle.properties)

```
<UserHomeDirectory>/.gradle/gradle.properties
```

Example 1. gradle.properties Example(In case the hostname of DADock Portal is [portal.dadock.fujitsu.local])

```
systemProp.http.proxyHost=<HostName>
systemProp.http.proxyPort=<Port>
systemProp.http.proxyUser=<UseName>
systemProp.http.proxyPassword=<PassWord>
systemProp.http.nonProxyHosts=gitlab.dadock.fujitsu.local|sonarqube.dadock.fujitsu.local|artifactory.dadock.fujitsu.local

systemProp.https.proxyHost=<HostName>
systemProp.https.proxyPort=<Port>
systemProp.https.proxyUser=<UseName>
systemProp.https.proxyPassword=<PassWord>
systemProp.https.nonProxyHosts=gitlab.dadock.fujitsu.local|sonarqube.dadock.fujitsu.local|artifactory.dadock.fujitsu.local
```

V2_1806
Last update 2018-08-17 03:57:31 UTC