



Behavioral Health Information Technology and Standards (BHITS) Project

Consent2Share Version 3 Development Guide

August 2018

Prepared by FEi Systems



This Consent2Share Version 3 Development Guide was developed by FEI Systems for the Behavioral Health Information Technologies & Standards (BHITS) contract, funded by the Substance Abuse and Mental Health Services Administration (SAMHSA).

Contents

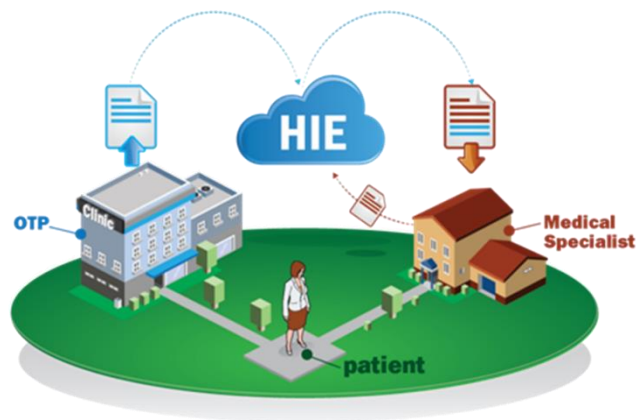
INTRODUCTION.....	5
Overview	5
Introduction	5
 ROADMAP.....	 6
 1 JAVA DEVELOPMENT ENVIRONMENT SETUP	 7
1.1 Java JDK	7
1.1.1 Install Java JDK	7
1.1.2 Set JAVA_HOME System Environment Variable and System Path.....	7
Set Environment Variables	7
1.1.3 Install Java Cryptography Extension (JCE)	9
1.2 Maven	9
1.2.1 Install Maven.....	9
1.2.2 Set M2_HOME System Environment Variable and System Path	9
1.3 Tomcat 8.....	10
1.3.1 Install Tomcat.....	10
1.3.2 Configure JVM and Other Options for Tomcat	10
1.3.3 Configure User to Operate the "/manager/html" Web Application	10
1.4 MySql Server and Workbench	10
1.4.1 Install.....	10
1.4.2 Set up System Environment Variable and System Path	10
1.5 Install Git for Windows	10
1.6 Gradle.....	10
1.6.1 Install.....	10
1.6.2 Set GRADLE_HOME System Environment Variable and System Path.....	10
1.6.3 The Gradle Wrapper.....	11
1.7 Install Flyway (Optional)	11
1.7.1 Prerequisites	11
1.7.2 Windows Installation:	11
 2 FRONTEND DEVELOPMENT ENVIRONMENT SETUP	 12
2.1 Install Node.js Platform	12
2.2 Install Angular CLI	12
 3 INSTALL AND CONFIGURE INTELLIJ IDEA FOR CONSENT2SHARE V3 DEVELOPMENT.....	 13
3.1 Install	13
3.1.1 Configure/Update JDK.....	13
3.1.2 Configure Maven.....	14
3.1.3 Set up Tomcat	15
3.1.4 Configure Github.....	16
3.1.5 Configure for Frontend Development.....	16

3.1.6	Configure Default Settings for File Template	16
3.1.7	Multirun Plugin	16
3.1.8	Configure Settings for Importing Java Classes	16
3.1.9	Install Lombok Plugin	18
4	IMPORT AND DEVELOP CONSENT2SHARE V3 PROJECTS IN INTELLIJ IDEA	19
4.1	Open Multiple Projects in the Same Window in IntelliJ IDEA	19
4.1.1	Create an Empty Project	19
4.1.2	Clone Project Git Repositories	20
4.1.3	Import C2S Projects as Modules in IntelliJ IDEA.....	21
4.2	Create Schemas in MySQL Workbench	25
4.3	Run/Debug Source Code	25
4.3.1	Run/Debug Configuration with Tomcat for UAA.....	25
4.3.2	Run/Debug Configuration with NPM for UI Projects	31
4.3.3	Use Multirun Plugin to Run Consent2Share.....	31
4.3.4	Add Environment Variables	33
4.3.5	Update Configuration for EdgeServerApplication.....	34
4.3.6	common-libraries Artifacts	34
4.3.7	Run Consent2Share Applications	34
4.3.8	Run SQL Scripts to Insert Data	36
	REFERENCES	37
	FLYWAY INSTALLATION INSTRUCTIONS.....	38

Introduction

Overview

Specially protected health information (PHI) covered under the federal confidentiality regulation 42 CFR Part 2 (health information from federally assisted drug and alcohol treatment programs) has generally not been included in the electronic exchange of patient information between health care providers. One of the primary reasons is the lack of technology options for patients to share part of their health information while not sharing others.



To address this issue, the Federal Office of the National Coordinator for Health Information Technology (ONC) developed the Data Segmentation for Privacy (DS4P) initiative to allow patients to share portions of an electronic medical record while not sharing others. In collaboration with the ONC, the Substance Abuse and Mental Health Services Administration (SAMHSA) developed the Consent2Share application to address the specific privacy protections for substance use treatment patients covered by the federal confidentiality regulation 42 CFR Part 2.

Consent2Share is an open source application for data segmentation and consent management. It is designed to integrate with existing FHIR (Fast Health Interoperability Resources) systems. Initially, SAMHSA funded the Open Behavioral Health Information Technology Architecture (OBHITA) contract to develop the Consent2Share application. Subsequently, SAMHSA funded the Behavioral Health Information Technology and Standards (BHITS) contract to further develop and conduct pilot testing of Consent2Share. Through a process of electronic consent, the patient controls how his or her sensitive health data will be shared by selecting categories.

Introduction

The main goal of this development guide is to provide clear direction for developers to set up their local technical environment to prepare to develop Consent2Share and contribute to code repository on GitHub. And if there is no explicit mention for the development environment operating system (OS), it is Windows 10 by default.



Roadmap

To describe the development environment setup, this development guide is organized in the following chapters:

- Chapter 1 describes the steps required to set up the Java (backend) development environment
- Chapter 2 describes the steps to set up the frontend development environment
- Chapter 3 describes how to install and configure IntelliJ IDEA for Consent2Share V3 development
- Chapter 4 focuses on Importing and developing Consent2Share V3 in IntelliJ IDEA

1 Java Development Environment Setup

The following development environment setups should be carried out in the following order.

1.1 Java JDK

Java 8 is used for Consent2Share development.

1.1.1 Install Java JDK

Go to the [Oracle JDK download site](#) to download the appropriate JDK for your operating systems.

1.1.2 Set JAVA_HOME System Environment Variable and System Path

After you have installed Java 8, java.exe could come from three places,

```
C:\Users\tao.lin>where java
C:\Program Files\Java\jdk1.8.0_51\bin\java.exe
C:\ProgramData\Oracle\Java\javapath\java.exe
C:\Windows\System32\java.exe
```

In order to control where java.exe should come from, do the following steps:

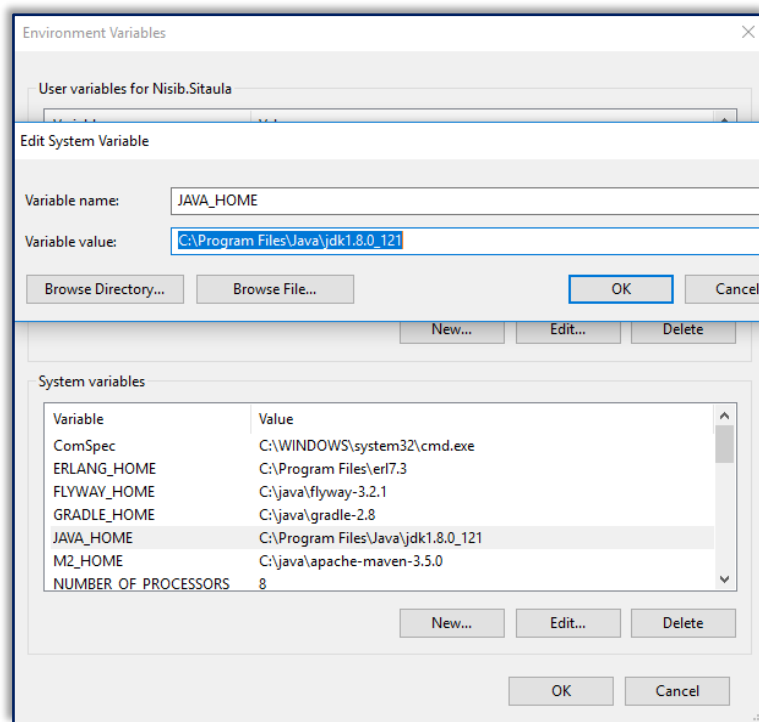
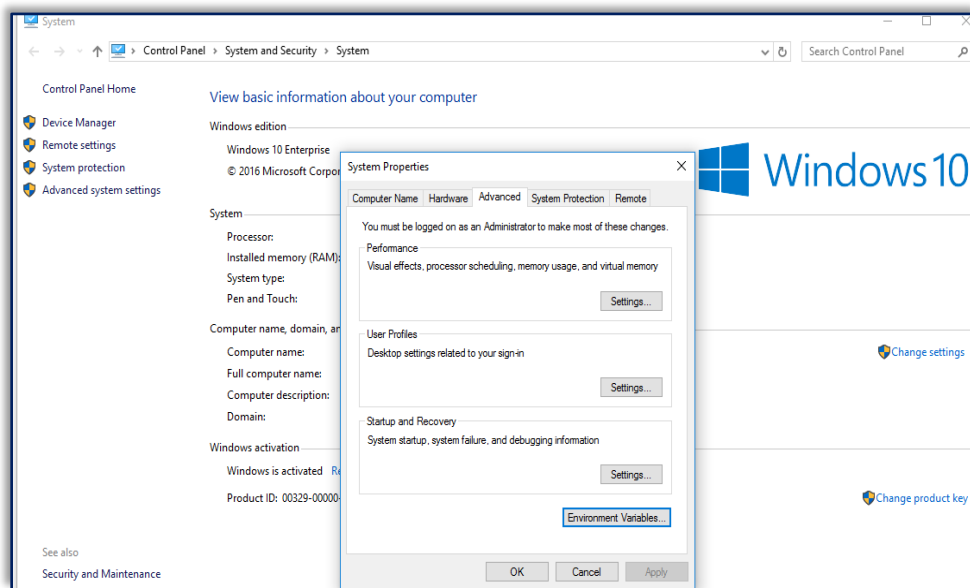
Set Environment Variables

Right click “This PC,” then select “Properties,” then “Advanced System settings,” and then “Environment variables”.

Put Environment Variable: JAVA_HOME

Value: C:\Program Files\Java\jdk1.8.0_xx (here xx is the update version)

Put %JAVA_HOME%\bin in the System Path (at the very beginning of the Path variable value).



At the Command Prompt:

Run “java -version” to check Java version.

Run “where java” to verify the java path and make sure that java is from the path you specified in JAVA_HOME (Java from JAVA_HOME should appeared as the first item.)

1.1.3 Install Java Cryptography Extension (JCE)

Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 8 Download is at <http://www.oracle.com/technetwork/java/javase/downloads/ice8-download-2133166.html>

You can download from this link.

- Uncompress and extract the downloaded file.

This will create a subdirectory called UnlimitedJCEPolicyJDK8. This directory contains the following files:

- ✓ README.txt
- ✓ local_policy.jar — Unlimited strength local policy file
- ✓ US_export_policy.jar — Unlimited strength US export policy file
- Install the unlimited strength policy JAR files.

In case you later decide to revert to the original "strong" but limited policy versions, first make a copy of the original JCE policy files (US_export_policy.jar and local_policy.jar). Then replace the strong policy files with the unlimited strength versions extracted in the previous step.

The standard place for JCE jurisdiction policy JAR files is:

➤ %JAVA_HOME%\jre\lib\security

1.2 Maven

1.2.1 Install Maven

- Maven 3 (Version 3.3.3) is used in Consent2Share development if not otherwise specified in individual project by Maven Wrapper.
- You can download it from <https://archive.apache.org/dist/maven/maven-3/> (the latest version from <https://maven.apache.org/download.cgi>).
- Extract maven from apache-maven-3.3.3-bin.zip to the c:\java folder.

1.2.2 Set M2_HOME System Environment Variable and System Path

- Set M2_HOME system environment variable:
 - ✓ Variable: M2_HOME
 - ✓ Value: C:\java\apache-maven-3.3.3
- Put %M2_HOME%\bin in the System Path.
- Run “mvn -version” to verify.

1.3 Tomcat 8

1.3.1 Install Tomcat

- You can download it from <https://tomcat.apache.org/download-80.cgi>
- Unzip to C:\java folder.

1.3.2 Configure JVM and Other Options for Tomcat

- You can configure JVM and other options for Tomcat.
- Go to the bin folder of the Tomcat installation, open catalina.bat/catalina.sh based on your operation system to see how you can configure.
- By default, Tomcat runs on JVM pointed by JAVA_HOME environment variable.

1.3.3 Configure User to Operate the "/manager/html" Web Application

Go to the conf folder of the Tomcat installation path, open tomcat-users.xml file, add the following line inside of tomcat-users element:

➤ `<user username="admin" password="admin" roles="manager-gui"/>`

1.4 MySQL Server and Workbench

1.4.1 Install

You can download it from <https://downloads.mysql.com/archives/installer/> (The latest version is located at <https://dev.mysql.com/downloads/windows/installer/>).

Set up username (root) and password (admin) as admin to access local MySQL Server.

1.4.2 Set up System Environment Variable and System Path

- Environment Variable: MYSQL_HOME
- Value: C:\Program Files\MySQL\MySQL Server 5.6
- Append path system variable with %MYSQL_HOME%\bin

1.5 Install Git for Windows

Download Git-2.14.1-64-bit.exe from <https://git-for-windows.github.io/>.

To handle line ending issue, open the gitbash and issue the following command:

➤ `git config --global core.autocrlf true`

Issue the following command to verify:

➤ `git config --get core.autocrlf`

1.6 Gradle

1.6.1 Install

- You can download it from <https://gradle.org/releases/>. e.g. gradle-2.8-all.zip is downloaded.
- Extract gradle from gradle-2.8-all.zip to the c:\java folder.

1.6.2 Set GRADLE_HOME System Environment Variable and System Path

- Set GRADLE_HOME system environment variable:
 - Variable: GRADLE_HOME

- Value: C:\java\gradle-2.8 (Based on your directory)
- Put %GRADLE_HOME%\bin in System Path.
- Run “gradle -version” to verify

1.6.3 The Gradle Wrapper

The Gradle Wrapper is the preferred way of starting a Gradle build. The wrapper is a batch script on Windows, and a shell script for other operating systems. When you start a Gradle build via the wrapper, Gradle will be automatically downloaded and used to run the build.

You should check the wrapper into version control. By distributing the wrapper with your project, anyone can work with it without needing to install Gradle beforehand. Even better, users of the build are guaranteed to use the version of Gradle that the build was designed to work with. Of course, this is also great for continuous integration servers (i.e. servers that regularly build your project) as it requires no Gradle installation and configuration on the CI server.

1.7 Install Flyway (Optional)

Refer to “Flyway Installation Instructions” in the Appendix for more information.

1.7.1 Prerequisites

- Flyway 3.2.1 and mysql-connector-java-5.1.26-bin.jar.
- Download from internet:
<http://flywaydb.org/getstarted/download.html>
<http://mvnrepository.com/artifact/mysql/mysql-connector-java/5.1.26>

1.7.2 Windows Installation:

- Copy and unzip ‘flyway-commandline-3.2.1-windows-x64.zip’ to your machine
- Set FLYWAY_HOME system environment variable and system path (Optional)
 - ✓ Variable: FLYWAY_HOME
 - ✓ Value: C:\java\flyway-3.2.1 (Based on your directory)
 - ✓ Put %FLYWAY_HOME% in System Path.
 - ✓ Run flyway --version to check Flyway version.
- Copy mysql-connector-java-5.1.26-bin.jar to flyway-3.2.1\jars
- To check flyway versioned database, you can go to flyway-3.2.1\conf directory and set up database configuration in flyway.conf file:
 - ✓ flyway.url=
 - ✓ flyway.user=
 - ✓ flyway.password=
- Then go to command prompt by typing “flyway info” command to verify the change history of the database

For more, please read: <http://flywaydb.org/documentation/commandline/>

2 Frontend Development Environment Setup

2.1 Install Node.js Platform

Go to <https://nodejs.org/>, download node.js for your operating system.

In addition to Node.js, npm is installed as well.

Note: Current C2S UI projects are using Node.js version **6.11.1** and npm version **3.10.10**

Verify the installation by issuing the following commands in command prompt:

- `node --version` (Node 6.11.1)
- `npm --version` (NPM 3.10.10)

2.2 Install Angular CLI

Angular CLI is a Command Line Interface (CLI) to automate your development workflow.

Note: Current C2S UI projects are using Angular CLI version **1.2.1**

Install:

- `npm install -g @angular/cli@1.2.1`

Then to verify the installation by issuing the following command:

- `ng --version`

3 Install and Configure IntelliJ IDEA for Consent2Share V3 Development

Consent2Share source code can be imported and developed in any IDE (e.g. [IntelliJ IDEA](#), [Eclipse](#), [Spring Tool Suite](#), and [NetBeans](#)). Since Consent2Share source code heavily uses Spring Projects, [IntelliJ IDEA](#) and [Spring Tool Suite](#) are the recommend IDEs.

The following set up is based on IntelliJ IDEA Ultimate version.

3.1 Install

Download the latest IntelliJ IDEA Ultimate from <https://www.jetbrains.com/idea/download>. Next, begin the installation. After installing, register your installation if you have the license.

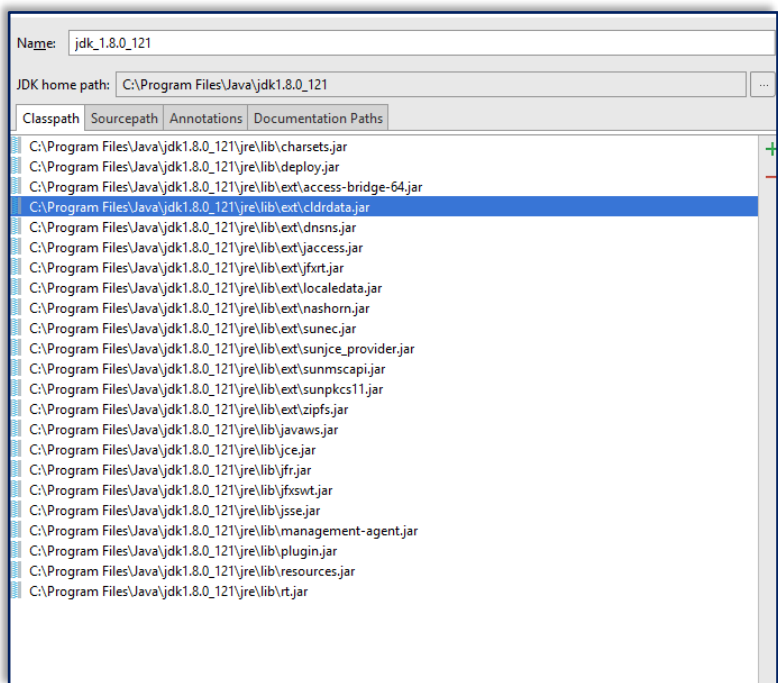
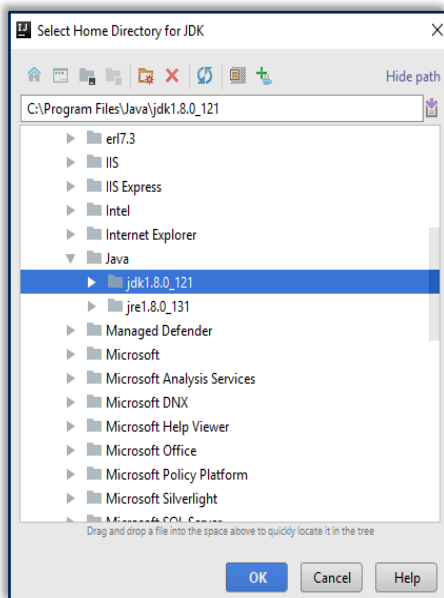
3.1.1 Configure/Update JDK

3.1.1.1 Configure SDKs at the Global (IDE) Level

- Open the Default Project Structure dialog (from Welcome Screen → Configure → Project Defaults → Project Structure, or from File → Other Settings → Default Project Structure).
- In the left-hand pane, under Platform Settings, click SDKs.
- To add a new SDK, click Add and select the desired SDK type.
- In the dialog box that opens, select the SDK home directory and click OK.
- As a result, a new SDK is added to IntelliJ IDEA, and its settings are shown on the SDK page in the right-hand part of the dialog box.
- Optionally, edit the SDK name and contents.
- If necessary, add more SDKs as described above.
- Click OK in the Project Structure dialog.

3.1.1.2 Configure a Default Project SDK

- Open the Default Project Structure dialog (from Welcome Screen → Configure → Project Defaults → Project Structure, or from File → Other Settings → Default Project Structure).
- In the left-hand pane, under Project Settings, click Project.
- On the page that opens in the right-hand part of the dialog, select the necessary SDK from the Project SDK list.
- If the desired SDK is not present in the list, click New and select the necessary SDK type.
- In the dialog that opens, select the SDK home directory and click OK. As a result, a new SDK is added to IntelliJ IDEA as Global Platform Settings and selected as the Default project SDK.
- To view or edit the SDK name and contents, click Edit. (The SDK page will open.)
- Click OK in the Project Structure dialog.



3.1.1.3 Configure a Project SDK

- Open the Project Structure dialog (e.g. Ctrl+Shift+Alt+S, or from File → Project Structure).
- In the left-hand pane, under Project Settings, click Project.
- On the page that opens in the right-hand part of the dialog, select the necessary SDK from the Project SDK list.
- If the desired SDK is not present in the list, click New and select the necessary SDK type. In the dialog that opens, select the SDK home directory and click OK. As a result, a new SDK is added to IntelliJ IDEA as Global Platform Settings and selected as the project SDK.
- To view or edit the SDK name and contents, click Edit. (The SDK page will open.)
- Click OK in the Project Structure dialog.

3.1.2 Configure Maven

3.1.2.1 Configure Maven at the Global (IDE) Level

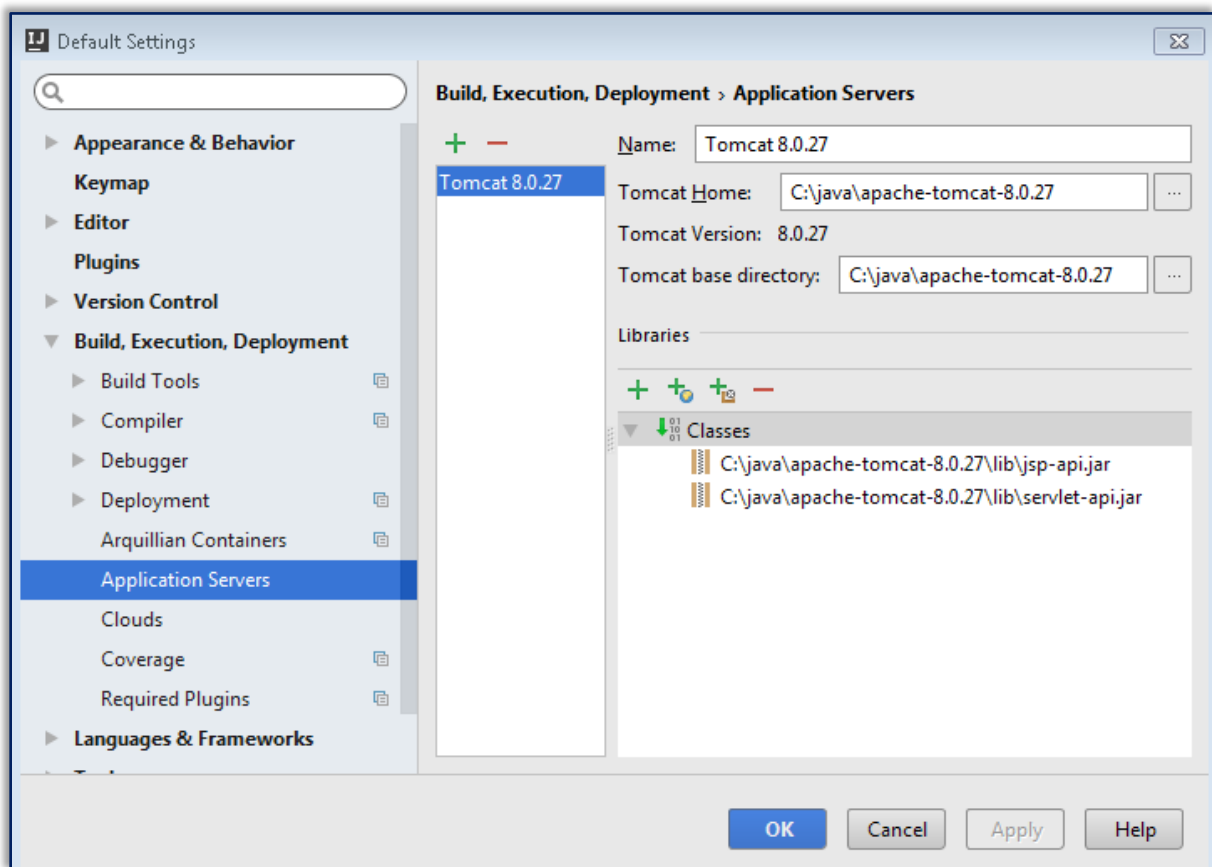
- Open the Default Settings dialog (from Welcome Screen → Configure → Project Defaults → Settings, or from File → Other Settings → Default Settings).
- In the left-hand pane, under Build, Execution, Deployment → Build Tools, click Maven.
- In the right-hand pane, set Maven home directory to the Maven directory which is the parent folder of bin folder.
- Click OK in the Default Settings dialog.

3.1.2.2 Configure Maven for a Project

- Open the Settings dialog (from File → Settings).
- In the left-hand pane, under Build, Execution, Deployment → Build Tools, click Maven.
- In the right-hand pane, set Maven home directory to the Maven directory, which is the parent folder of the bin folder.
- Click OK in the Settings dialog.

3.1.3 Set up Tomcat

- From the Welcome Screen, click Configure → Settings, open Settings dialog. Alternatively, after you open your project, open the Settings dialog from File → Settings. Both ways achieve some results here, but the first way is better.
- In the left-hand pane, under Build, Execution, Deployment, click Application Servers.
- Click the green “+” in the middle pane to set up Tomcat. Provide a meaningful name for the application server e.g. Tomcat 8.0.27.



3.1.4 Configure Github

From the Welcome screen, open the Default Settings dialog. Version Control → Github. Select and enter the followings in the right-hand pane.

- Host: github.com
- Auth Type: Password
- Login: your username
- Password: your-password

3.1.5 Configure for Frontend Development

3.1.5.1 Install the NodeJS Plugin

- In the Welcome Screen, click Configure → Settings → Plugins → Browse repositories, → type node.js to find the NodeJS plugin → Install plugin.
- Then you will be able to use Run → Edit Configuration, to open Run/Debug Configurations, and configure to run NodeJS.

3.1.5.2 Install the Karma Plugin

In the Welcome Screen, click Configure → Settings → Plugins → Install JetBrains plugin ... → type karma to find the Karma plugin → click the green Install Plugin button to install.

3.1.6 Configure Default Settings for File Template

We do not want the “Created by” header in our source code files when creating new files. Use the following steps to configure:

- Select File → Other Settings → Default Settings.
- Expand the Editor option in the sidebar, then select File and Code Templates.
- Select the Includes tab, and then select File Header from the list. You will then see three lines of code in the box to the right of the list.
- Select all three lines of code in the box and delete them. Make sure the box is empty.
- Click the OK button.

3.1.7 Multirun Plugin

The Multirun Plugin enables multiple run configurations at once, group multiple run configurations, and the ability to start them in a single click. Not only can application and test run configurations be grouped, but other Multirun configurations can be organized into a single run configuration.

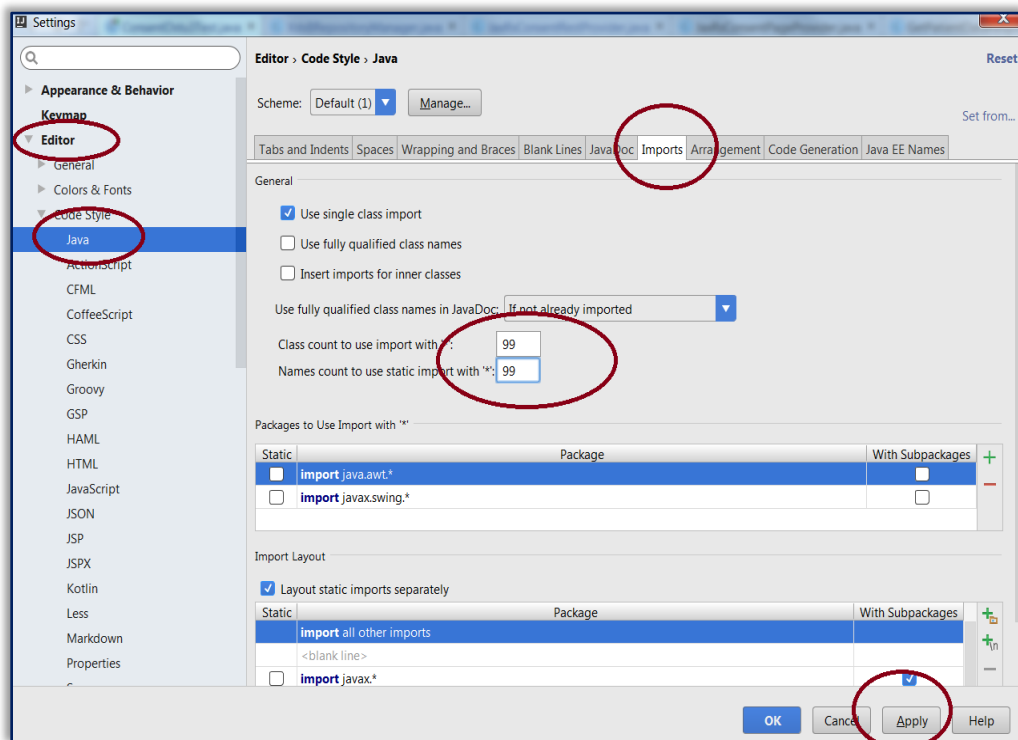
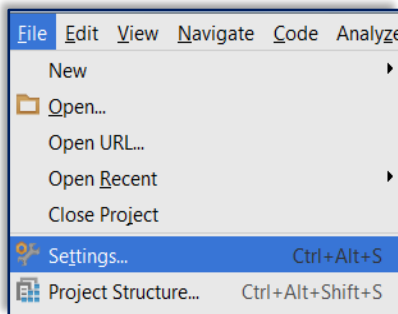
Use the following method to install Multirun Plugin:

From Welcome Screen → Configure → Settings to Open Default Settings, Plugins then search for Multirun and install.

3.1.8 Configure Settings for Importing Java Classes

To force IntelliJ include each and every import individually instead of using (*), do the following:

- Go to File → Settings → Editor → Code Style → Java → Import tab
- Set Class count to use import with '*' to 99 (a number which is bigger enough)
- Set Names count to use static import with '*' to 99 (a number that is sufficiently large).



3.1.9 Install Lombok Plugin

Lombok is a framework that generates boilerplate code in an annotation-driven fashion and it has been utilized in several Consent2Share services. The projects will successfully build if the Lombok is on classpath (as a maven dependency). However it is required to also install a plugin to IDE to prevent errors. For IntelliJ Plugin installation follow these steps:

- Go to: File → Settings → Plugins
- Search for Lombok to see whether or not it is installed.
- If it is already installed, no additional steps are required. If it is not installed, click the Search in Repositories link.
- Select Lombok Plugin and click the Install button. Potentially, you might need to restart the IDE after installation.
- See <https://projectlombok.org/> for details including documentation and support for other IDEs.

4 Import and Develop Consent2Share V3 Projects in IntelliJ IDEA

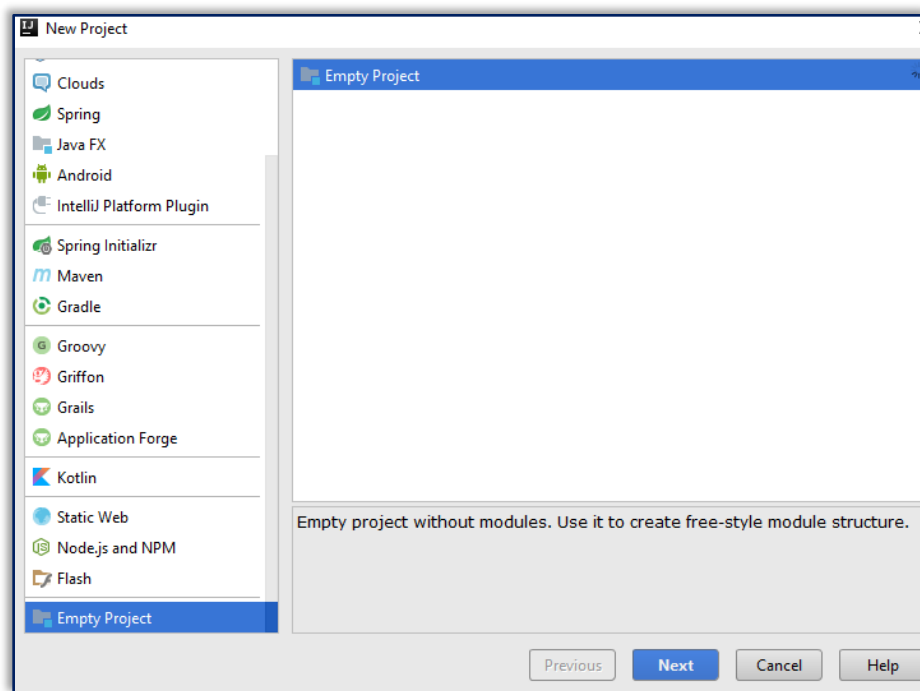
The first three chapters involved installing tools required to develop the Consent2Share application. This chapter focuses on how to set up Consent2Share V3 projects in IntelliJ IDEA for development.

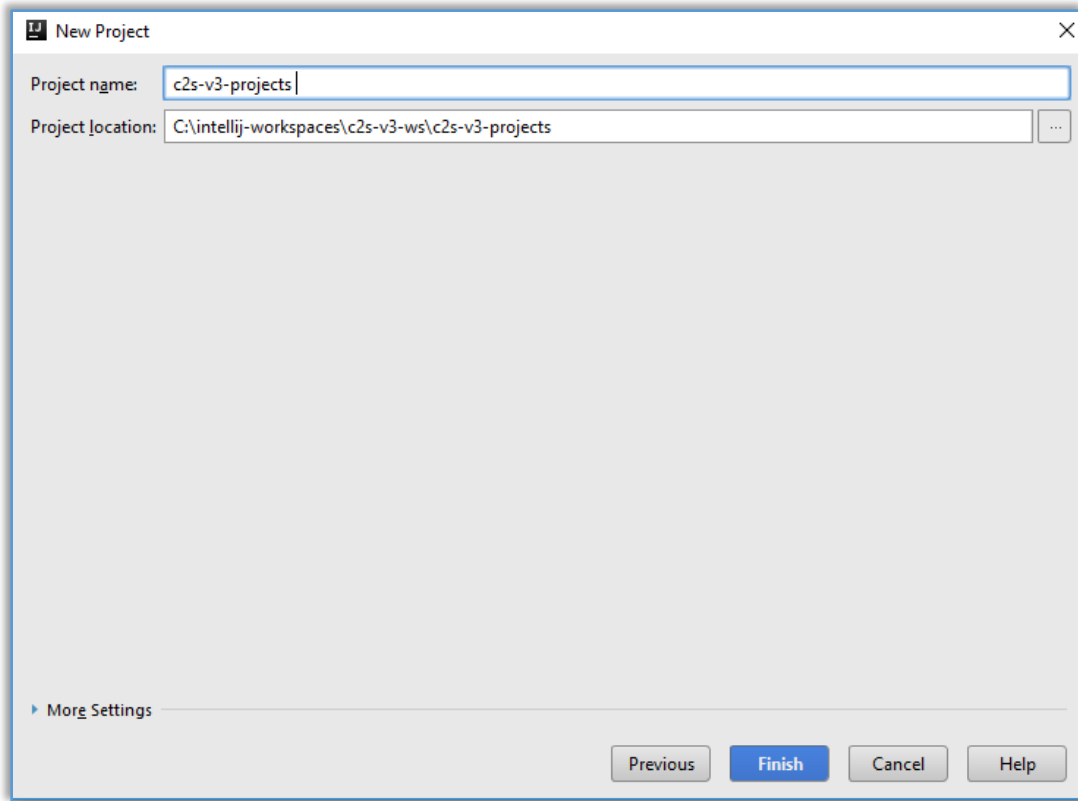
4.1 Open Multiple Projects in the Same Window in IntelliJ IDEA

If you are working with multiple projects, it is very convenient to open these projects in the same window for easy development. We can do this by creating an empty project and housing other projects as modules:

4.1.1 Create an Empty Project

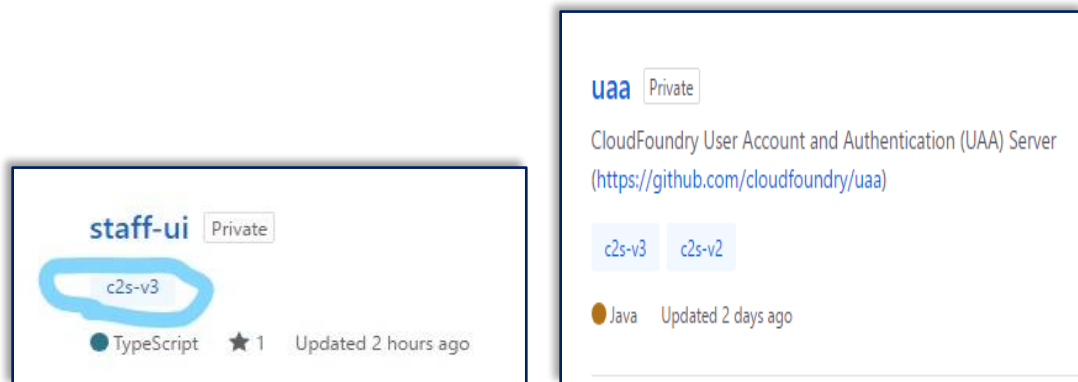
- In C:\ drive, create a new folder: intellij-workspaces
- Inside C:\intellij-workspaces, create another empty folder: c2s-v3-ws.
- Open IntelliJ
- From the Welcome Screen, click Create New Project or File → New → Project → Empty Project
- Provide the name of the project (eg: c2s-v3-projects) and set the project location in c2s-v3-ws\c2s-v3-projects then click finish button.





4.1.2 Clone Project Git Repositories

In your `intellij-workspaces\c2s-v3-ws` folder, we can now clone all the Git repositories for Consent2Share labeled by `c2s-v3` topic. Examples of such Git repositories are given below:



Clone all of the Git repositories labeled with `c2s-v3` topic.

4.1.2.1 Place `c2s-config-data` Repository

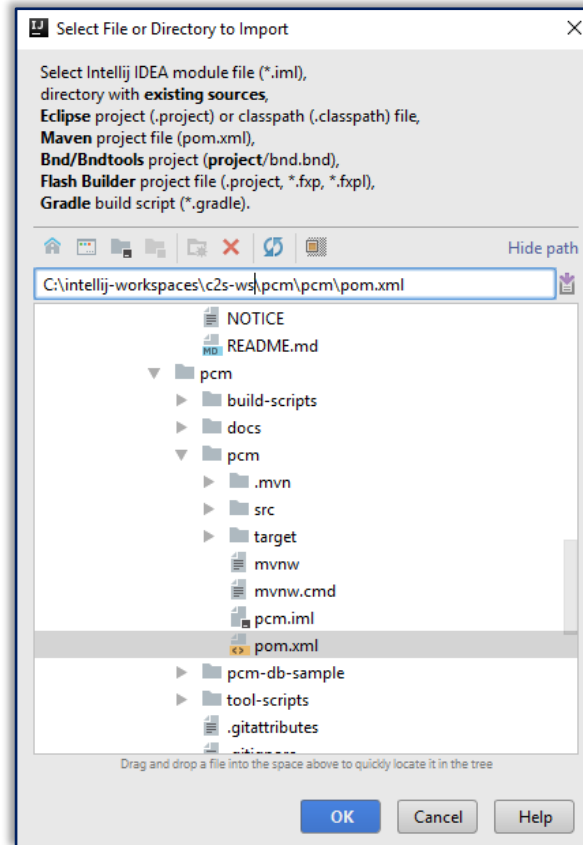
Based on the configuration (`config-server/src/main/resources/application.yml`) in `config-server` project, the configuration data Git repository (`c2s-config-data`) need be placed in a specific location which is `C:\java`.

Move the cloned c2s-config-data Git repository to C:\java folder or directly go to C:\Java and clone c2s-config-data Git repository.

4.1.3 Import C2S Projects as Modules in IntelliJ IDEA

4.1.3.1 Import all the Maven Projects

In the opened empty project that we created in section 4.1.1, go to File → New → Module from Existing Sources. Go to the appropriate cloned C2S project git repository work directory and select its pom.xml (or you can just select the folder which contains the pom.xml file), and click OK button to open Import Module modal dialog.



In the opened Import Module modal dialog, select Maven for Import module from external model, then click Next button to go to next step in Import Module modal dialog.

Select the *Import Maven projects automatically* option.

☒ Search for projects recursively

☐ Keep project files in:

☐ Import Maven projects automatically

☒ Create IntelliJ IDEA modules for aggregator projects (with 'pom' packaging)

☐ Create module groups for multi-module Maven projects

☒ Keep source and test folders on reimport

☒ Exclude build directory (%PROJECT_ROOT%/target)

☒ Use Maven output directories

Generated sources folders:

Phase to be used for folders update:

IDEA needs to execute one of the listed phases in order to discover all source folders that are contained in the project.
Note that all test-* phases firstly generate and compile production sources.

Automatically download: ☐ Sources ☐ Documentation

Dependency types:

Comma separated list of dependency types that should be imported

Click the Next button, then click Finish button to close Import Module modal dialog.

In the project Tool Window, the project is listed as a module.

Follow the above steps to import all the C2S Maven projects as modules for the project named c2s-v3-projects.

Don't forget to import the server Maven projects in the UI projects (e.g. c2s-ui, master-ui etc.)

Alternatively, you can import all Maven projects in Project Structure modal dialog: go to File → Project Structure. To open the Project Structure modal dialog, click Modules under Project Settings to open two panes on the right, in the middle pane, click + button and then select Import Module to import existing Maven projects.

4.1.3.2 Import Client Projects

For Consent2Share UI projects, such as c2s-ui, the Maven pom.xml is located in the server folder. Using the pom.xml file only loads the server side project. We have to import the client site projects for all UI projects in IntelliJ IDEA.

To import c2s-ui client project, go to File > New > Module from Existing Sources. In the opened modal dialog, choose c2s-ui\client directory and click OK button to open Import Module modal

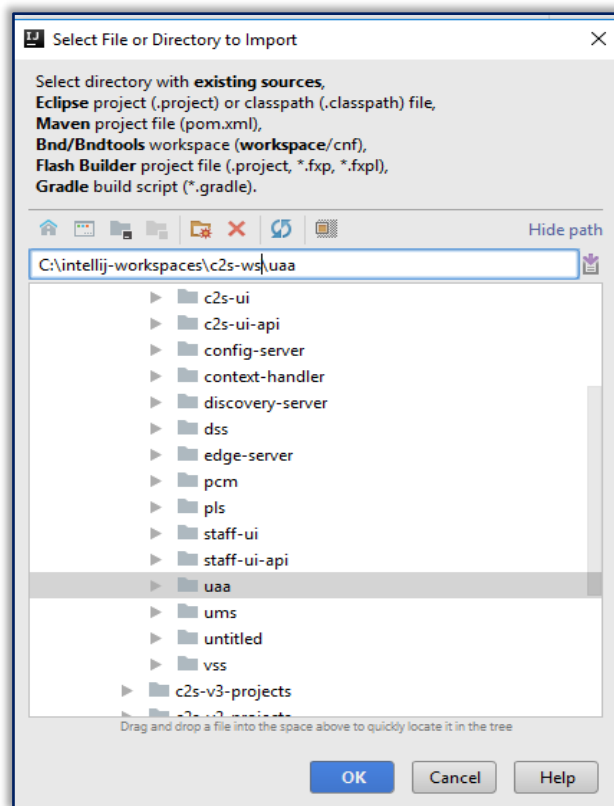
dialog. Choose “Create module from existing source” in the modal dialog. Continue to follow along to finish the dialog.

Once the client project as a module is imported, it is named as a client in the Project Tool Window. Rename it to distinguish it from the other client projects. Right click the client module and select “Open Module Settings” and provide a name. For example: c2s-ui-client.

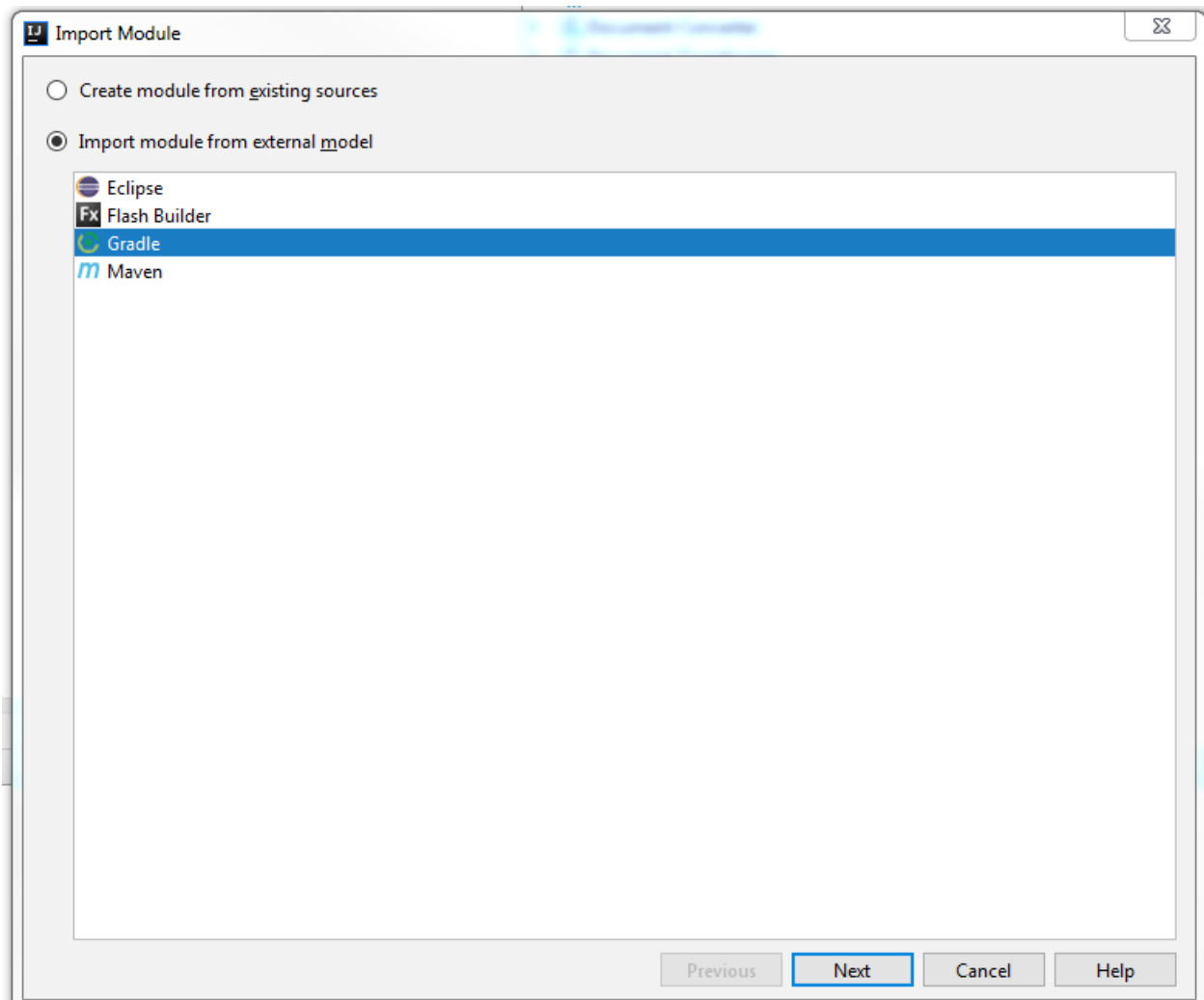
Follow the same steps to import other client projects.

4.1.3.3 Import UAA

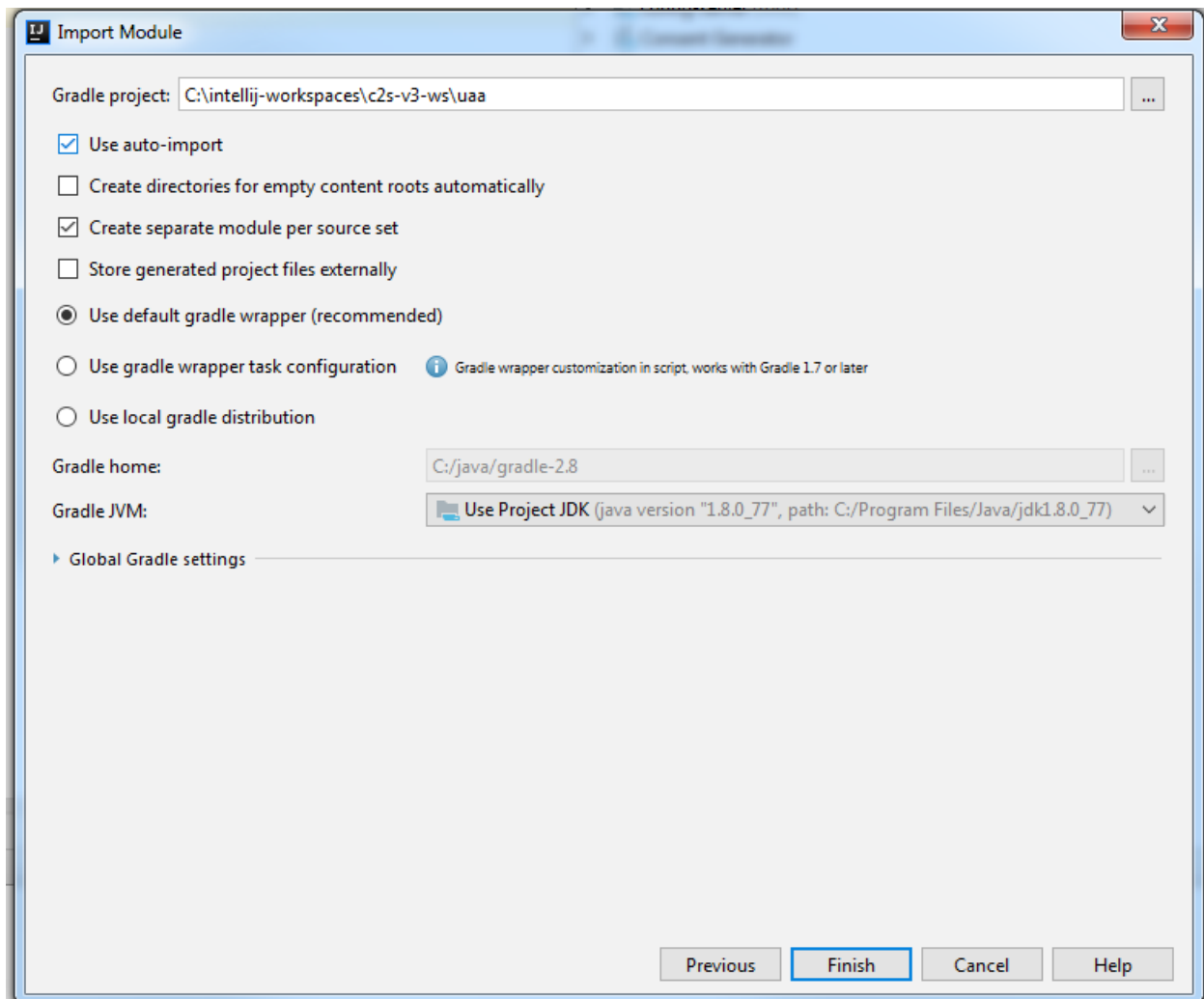
UAA is a Gradle project. To import the UAA project, go to File → New → Module from existing sources and then select the uaa directory which contains build.gradle. Click OK button to open Import Module modal dialog.



In the dialog, choose Import module from external model, and then choose Gradle. Then, click the Next button.



Check Use auto-import and click the Finish button.



4.2 Create Schemas in MySQL Workbench

Create empty schemas called: pcm, phr, uaa, ums, vss and pls.

Ex: CREATE SCHEMA 'uaa';

Tables and data will be created and inserted when corresponding applications are running for the first time.

4.3 Run/Debug Source Code

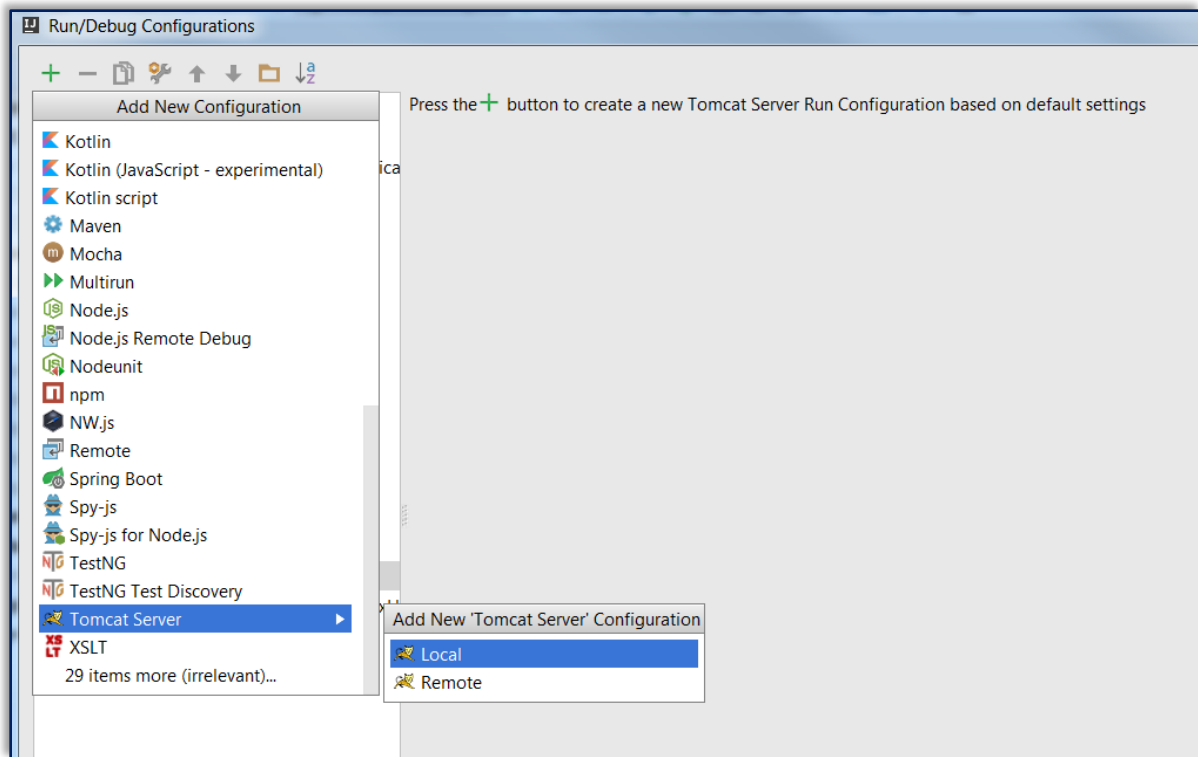
When you import a Spring Boot project, a Run/Debug configuration is automatically generated.

For UAA you need to configure Run/Debug configuration to use Tomcat server.

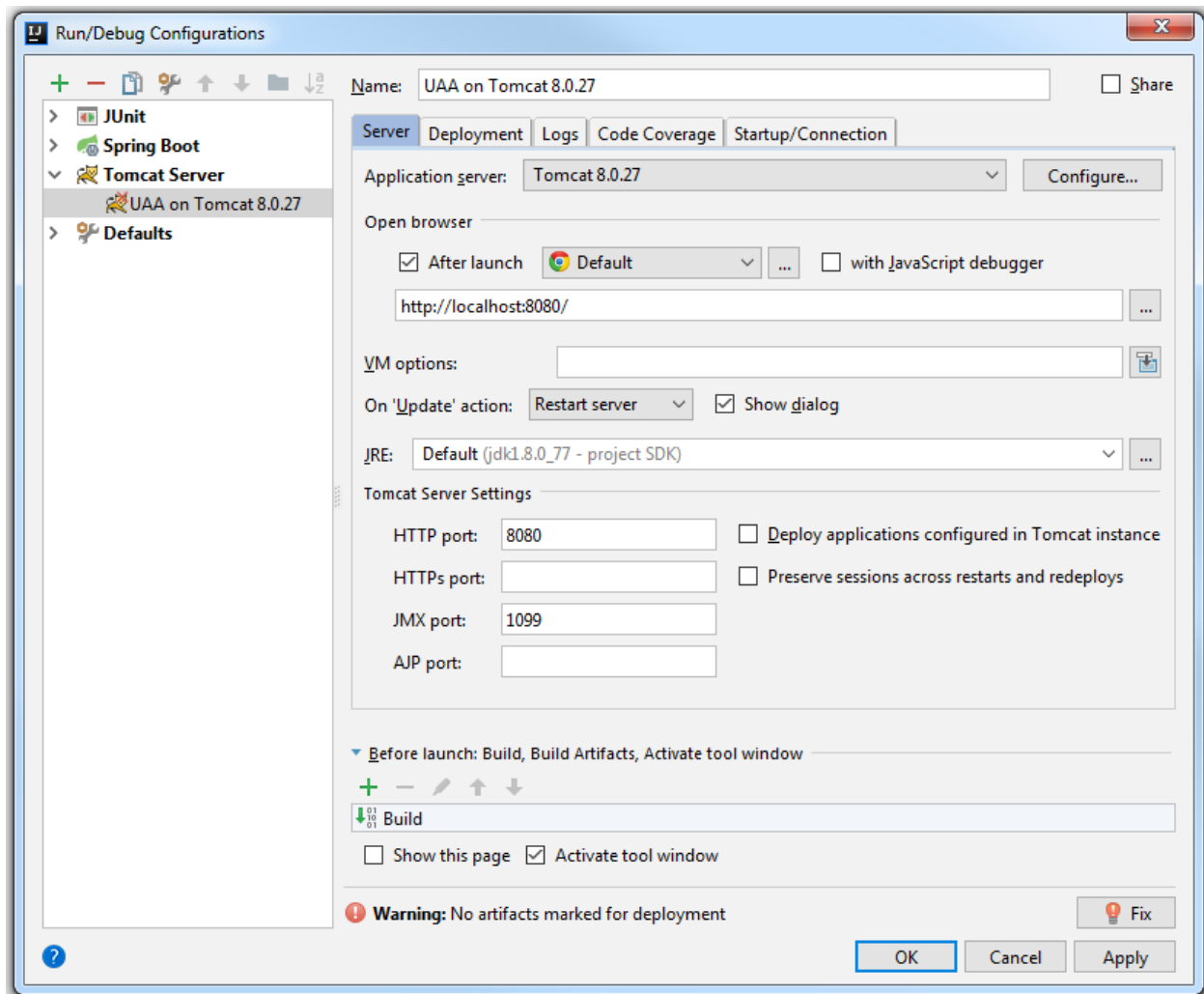
4.3.1 Run/Debug Configuration with Tomcat for UAA

Click Run → Edit Configurations to open "Run/Debug Configurations" dialog.

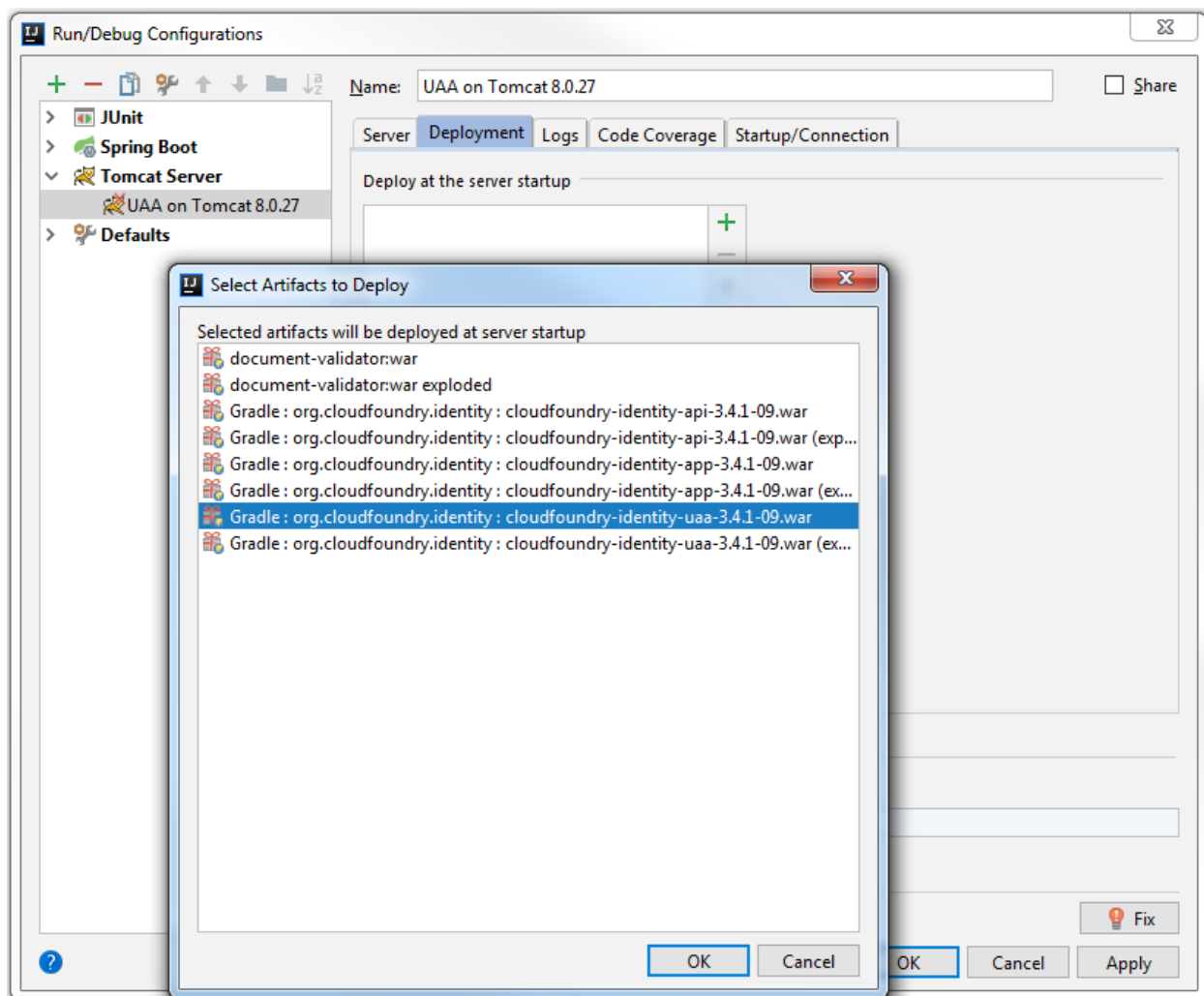
Click "+" in the upper left, and Select and press Tomcat Server → local to open new Tomcat Server Run Configuration.



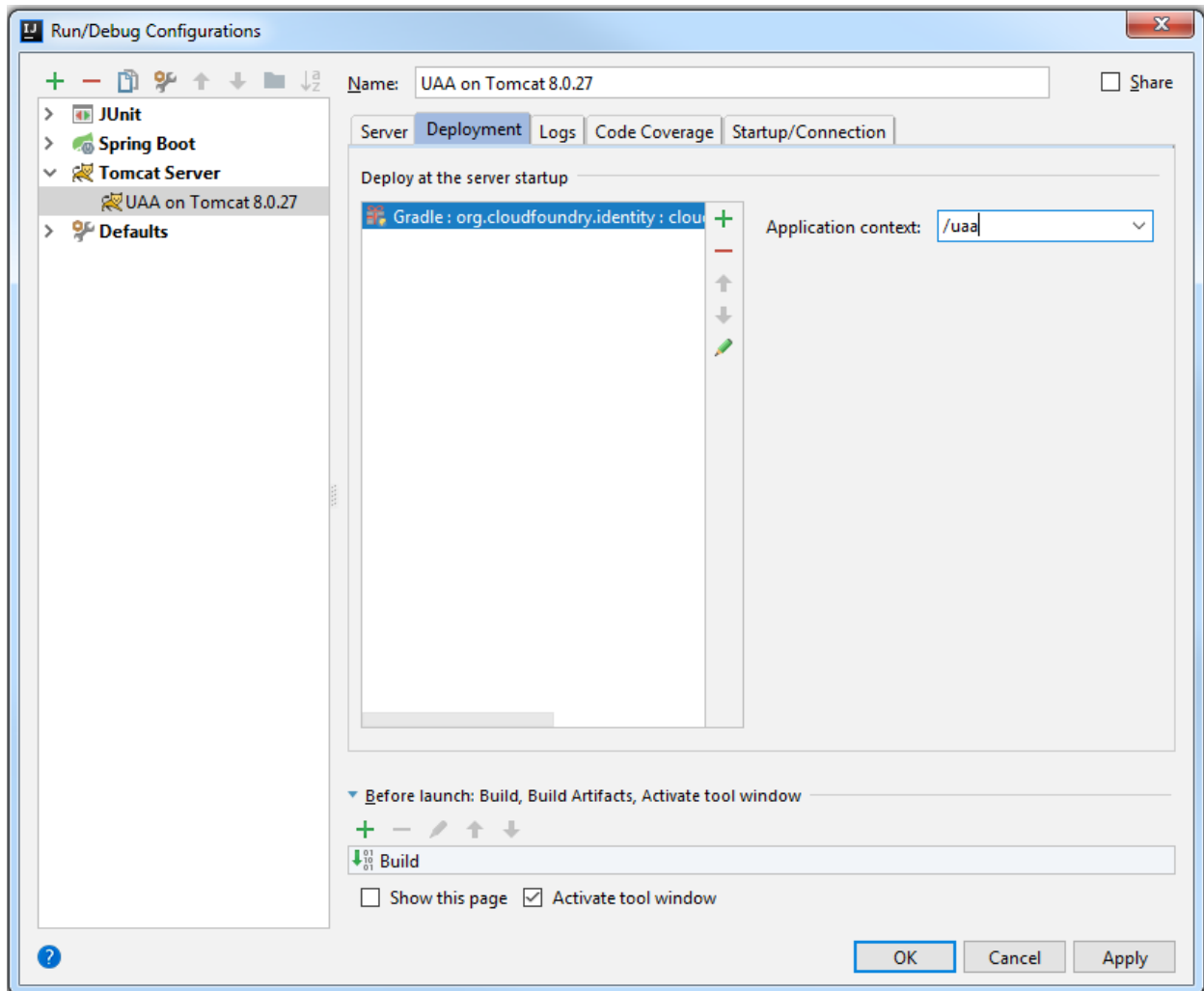
Next, give the Configuration a relevant name such as: “UAA on Tomcat 8.0.27”.



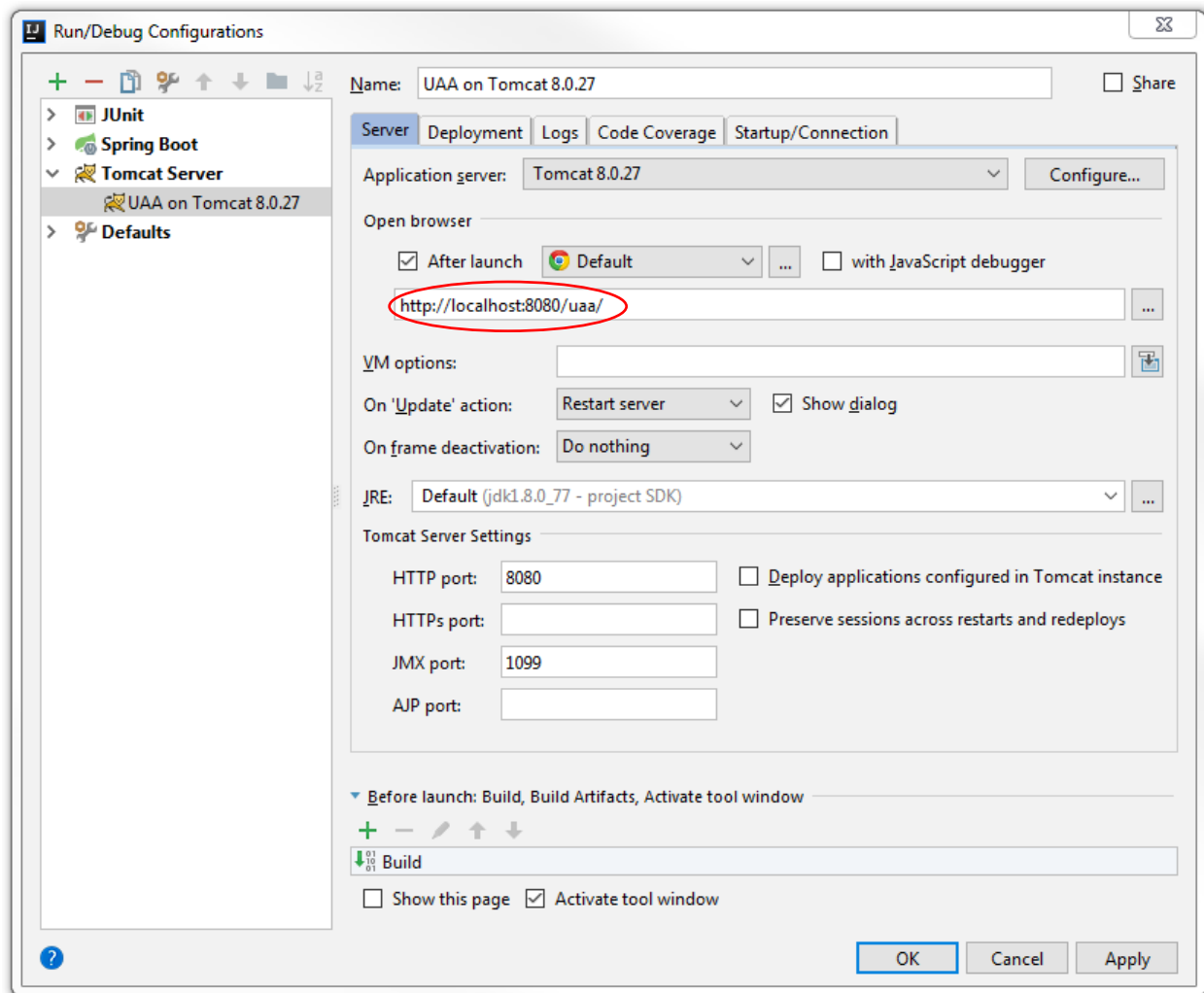
Next, click the Deployment tab and click the “+” button under “Deploy at the server startup”. Then click “Artifact...” to open the “Select Artifacts to Deploy” modal dialog. Choose uaa war.



Click the OK button to close “Select Artifacts to Deploy” dialog and return to “Run/Debug Configurations” dialog. And make sure to change “Application Context” to /uaa.



Now switch back to the Server tab. You should notice that the URL under the Open browser has changed to: <http://localhost:8080/uaa/>



Click the OK button to close the “Run/Debug Configurations” dialog box.

Add the UAA Configuration Path Environment Variable:

You can add UAA_CONFIG_PATH as an OS environment variable or a JVM environment variable or application server environment variable. Here we set up the environment variable at the application server level.

Append the following configuration line to the file of “catalina.properties” under the Tomcat directory. (C:\java\apache-tomcat-8.0.27\conf):

UAA_CONFIG_PATH=C:\\intellij-workspaces\\c2s-v3-ws\\uaa\\config-template

You can run this configuration to get a running UAA in the url <http://localhost:8080/uaa/>. The first time running of UAA creates tables and inserts data in the empty uaa schema.

4.3.2 Run/Debug Configuration with NPM for UI Projects

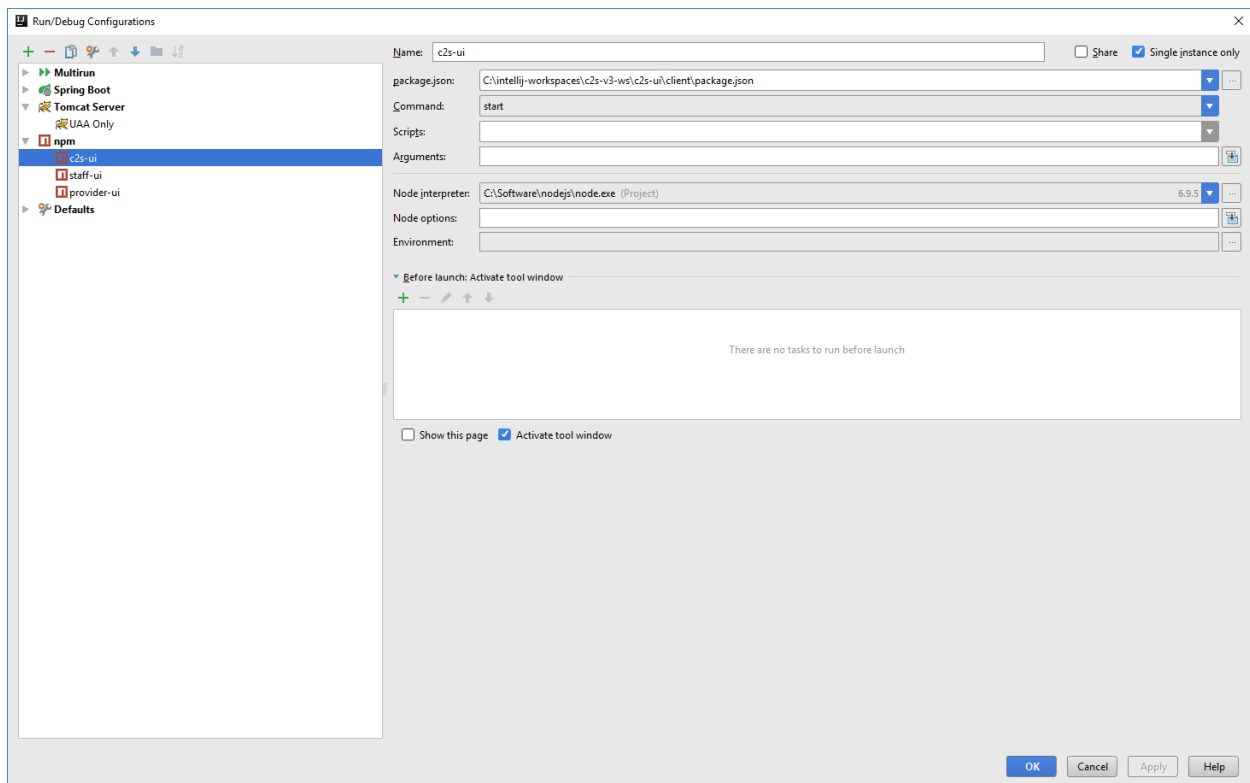
You can run Consent2Share UI projects by using the Spring Boot run configurations generated by IntelliJ IDEA.

The alternative way to run UI projects (Angular SPA projects) is to set up npm Run Configurations for these UI projects.

Click Run → Edit Configurations to open “Run/Debug Configurations” dialog.

Click “+” in the upper left, and select and press npm to open new npm Run Configuration.

Choose npm and ‘+’ icon. Create a c2s-ui run configuration as shown below with package.json pointing to c2s-ui project package.json in client folder:



Create the same kinds of run configurations for master-ui, staff-ui and provider-ui.

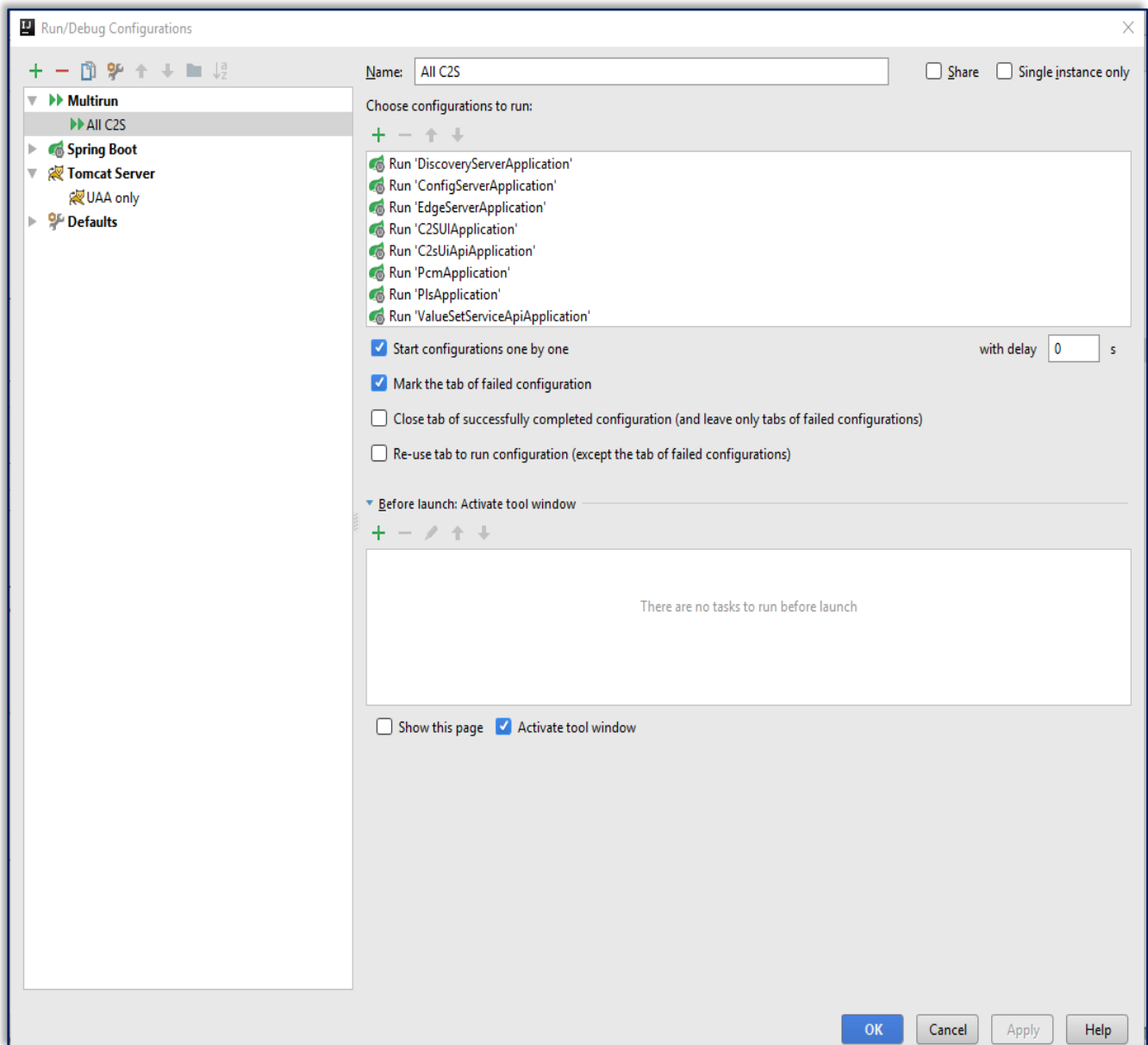
You cannot get these projects to work by only using these npm run configurations without running the required backend services.

4.3.3 Use Multirun Plugin to Run Consent2Share

Go to Run > Edit Configurations to open “Run/Debug Configurations” dialog. Click the “+” button at the left top side of the dialog to open “Add New Configuration” list. Click “Multirun” in the list to add a new Multirun configuration.

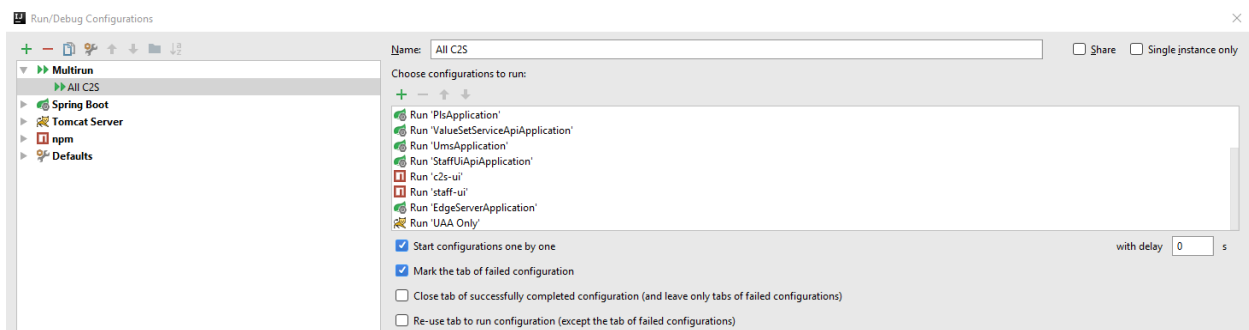
Give a meaningful name to the new Multirun configuration. Under “Choose configuration to run:”, and click “+” to add Consent2Share Run configurations for this multirun configuration.

Always add “Run ‘DiscoveryServerApplication’” and “Run ‘ConfigServerApplicaition’” first. Add all other required Consent2Share Spring Boot Run configurations to the Run Multirun configuration.



If you don't want to Spring Boot Run configurations for Consent2Share UI projects, add npm Run Configurations related to UI projects to this Multirun Configuration. But do not add both kinds of the Run Configurations for the same UI project at the same time.

Also add “Run ‘UAA on Tomcat 8.0.27’” configuration if you want to run UAA together with ‘All C2S’ configuration otherwise it needs to be run separately.



To speed up development, you do not need to run all Consent2Share projects in this Multirun Run Configuration. You have the flexibility to choose which projects to be included in this Multirun configuration. Or you may want to create several Multirun Run Configurations each of which has different projects included.

4.3.4 Add Environment Variables

To run Consent2Share in your IDE, set up the following environment variables with corresponding values (variable name: variable value):

```
spring.mail.protocol: smtp
spring.mail.host: ${your-smtp-mail-host}
spring.mail.port: 25
spring.mail.username: ${ask-team-member}
spring.mail.password: ${your-smtp-mail-password}
spring.mail.properties.mail.smtp.auth: true
spring.mail.properties.mail.smtp.ssl.trust: ${your-smtp-mail-host}
spring.mail.properties.mail.smtp.starttls.enable: true
```

```
UAA_SMTP_HOST: %spring.mail.host%
UAA_SMTP_PORT: %spring.mail.port%
UAA_SMTP_USER: %spring.mail.username%
UAA_SMTP_PASSWORD: %spring.mail.password%
```

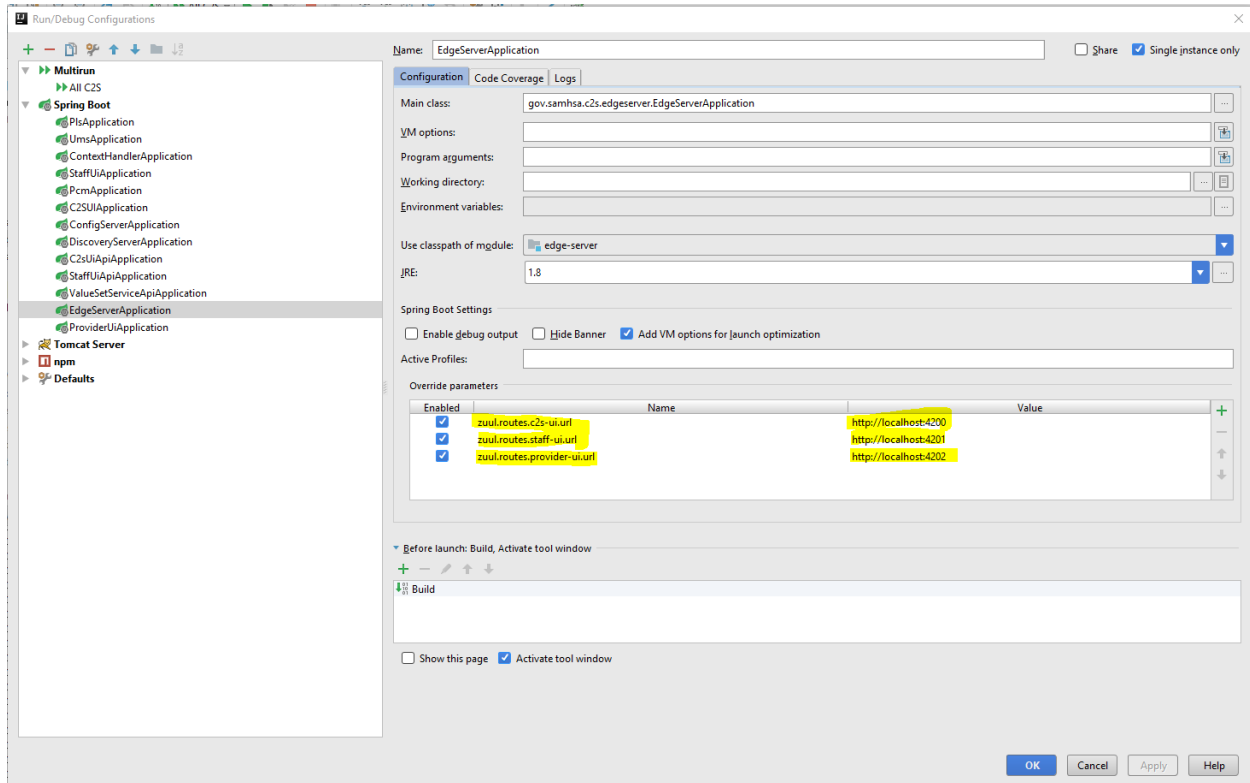
These environment variables can be set up as OS environment variables, JVM environment variable, or application server environment variables. For Spring Boot applications, parameters can be set up at command line as well. For convenience, we set up them as OS environment variables.

The reason that we set up these environment variables is that we do not want to hard-code sensitive information in configuration files that are source controlled and open-sourced. We could encrypt these values and put the encrypted values in configuration files for local development in IDE.

4.3.5 Update Configuration for EdgeServerApplication

To quickly run UI projects in debug mode, do the following setup for EdgeServer run configuration.

Go to Run → “Edit Configurations...” to open “Run/Debug Configurations” modal dialog. Click “Spring Boot” → EdgeServerApplication. Choose the Configuration tab and add the values as shown below in the “Override parameters” section:

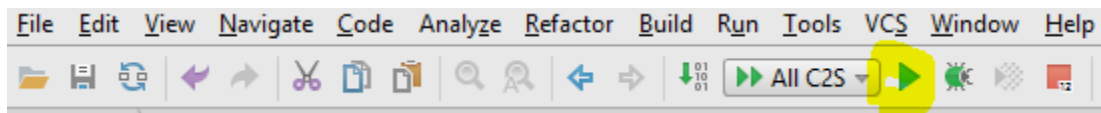


4.3.6 common-libraries Artifacts

Consent2Share projects depend on common-libraries Artifacts. If common-libraries artifacts are not available in the artifacts repository (specified in Maven settings.xml) that you are using, you will need to first build these artifacts to put them in your artifacts repository or your local maven repository.

4.3.7 Run Consent2Share Applications

Now, you can run the Consent2Share applications by clicking the button shown below.



The first time you run these applications, tables are generated for the empty schemas created in section 4.2.

You can use the following URLs to check Consent2Share:

- <http://localhost:8761/>
Eureka Dashboard
- <http://localhost:8080/uaa/>
UAA login page
- localhost/c2s-ui
- localhost/staff-ui
- localhost/provider-ui

Check the Eureka server to check the instances currently registered.

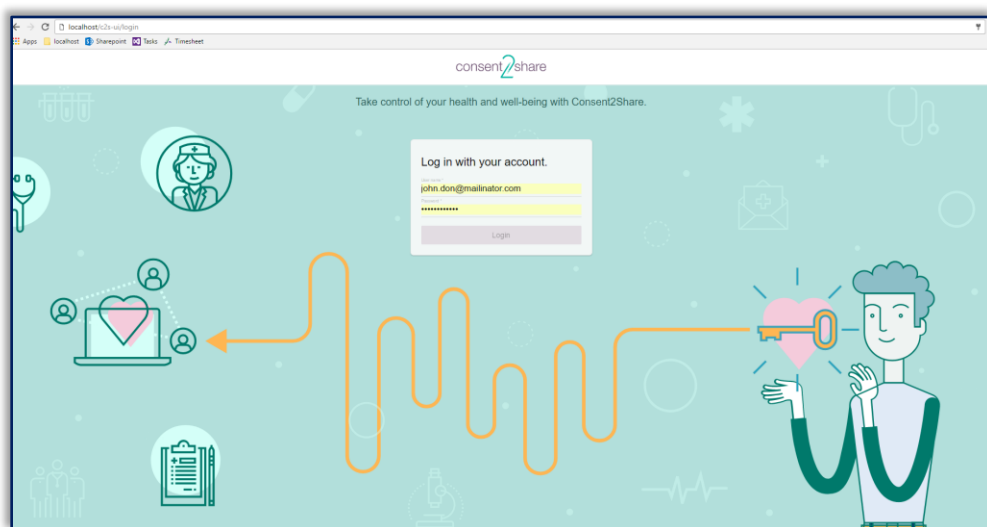
The screenshot shows the Spring Eureka dashboard at localhost:8761. The top navigation bar includes 'HOME' and 'LAST 1000 SINCE STARTUP'. The 'System Status' section displays the following information:

Environment	test	Current time	2017-06-15T13:55:59 -0400
Data center	default	Uptime	02:30
		Lease expiration enabled	true
		Renews threshold	0
		Renews (last min)	16

Below the status section, a red warning message states: 'THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.'

The 'DS Replicas' section is followed by 'Instances currently registered with Eureka', which lists the following applications:

Application	AMIs	Availability Zones	Status
C2S-UI-API	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:c2s-ui-api:0227ae7e0a11e895fa876e17ed59096a
CONFIG-SERVER	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:config-server:f7cc2f48274771b232c734f11bd25bf
EDGE-SERVER	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:edge-server:f2f616dc38f323f06b0161395252e894
PCM	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:pcm:71373b26cd61b740f046cea7b322ee5d
PLS	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:pls:32a13f8bb178485291a8d5fbc03b2e0
STAFF-UI-API	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:staff-ui-api:8b43adc84933757e037d93767db25b8c
UMS	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:ums:ca08156d7fbadb22b9b6d882f2e6f962
VSS	n/a (1)	(1)	UP (1) - URajkamikarLT.fei.local:vss:fb03f6915c0876e38f8e2f02b5626905





4.3.8 Run SQL Scripts to Insert Data

Run the scripts available at the following locations to insert lookup data and sample data:

C:\intellij-workspaces\c2s-v3-ws\pcm\pcm-db-sample

C:\intellij-workspaces\c2s-v3-ws\ums\ums-db-sample

C:\intellij-workspaces\c2s-v3-ws\vss\vss-db-sample

C:\intellij-workspaces\c2s-v3-ws\phr\phr-db-sample

C:\intellij-workspaces\c2s-v3-ws\phr\pls-db-sample



References

Github References

<https://git-scm.com/book/en/v2>

<https://app.pluralsight.com/library/courses/git-fundamentals/table-of-contents>

Maven References

<http://maven.apache.org/guides/>

<https://app.pluralsight.com/library/courses/maven-fundamentals/table-of-contents>

Gradle References

<https://gradle.org/docs>

<https://app.pluralsight.com/library/courses/gradle-fundamentals/table-of-contents>

Git Credential Storage

<https://github.com/Microsoft/Git-Credential-Manager-for-Windows>

Flyway Installation Instructions

1. Prerequisites

Flyway 3.2.1 and mysql-connector-java-5.1.26-bin.jar are needed.

Download them from the Internet:

<http://flywaydb.org/getstarted/download.html>

<http://mvnrepository.com/artifact/mysql/mysql-connector-java/5.1.26>

2. Windows Installation

1. Copy and unzip 'flyway-commandline-3.2.1-windows-x64.zip' to your machine

2. Set FLYWAY_HOME system environment variable and system path (Optional)

Variable: FLYWAY_HOME

Value: C:\java\flyway-3.2.1 (Based on your directory)

Put '%FLYWAY_HOME%' in System Path.

Run "flyway -version" to check Flyway version.

3. Copy 'mysql-connector-java-5.1.26-bin.jar' to flyway-3.2.1\jars

4. Go to directory flyway-3.2.1\conf directory and set up configuration

a. flyway.url=jdbc:mysql://localhost:3306/consent2share-bl

b. flyway.user=username

c. flyway.password=password

5. Go to command prompt by typing "flyway info" command to verify if the flyway works.

```
C:\Users\jiahao.li>flyway info
Flyway 3.2.1 by Boxfuse
Database: jdbc:mysql://localhost:3306/consent2share-bl (MySQL 5.6)
```

Version	Description	Installed on	State
1.0.0	Database schema	2015-03-24 13:55:02	Future
1.0.1	Address use code lookup data	2015-03-24 13:55:02	Future
1.0.2	Administrative gender code lookup data	2015-03-24 13:55:02	Future
1.0.3	Body site code lookup data	2015-03-24 13:55:02	Future
1.0.4	Clinical document section type code lookup data	2015-03-24 13:55:02	Future
1.0.5	Clinical document type code lookup data	2015-03-24 13:55:02	Future
1.0.6	Confidentiality code lookup data	2015-03-24 13:55:02	Future
1.0.7	Consent directive type code lookup data	2015-03-24 13:55:02	Future
1.0.8	Country code lookup data	2015-03-24 13:55:02	Future
1.0.9	Ethnic group code lookup data	2015-03-24 13:55:02	Future
1.0.10	Facility type code lookup data	2015-03-24 13:55:02	Future
1.0.11	Language ability code lookup data	2015-03-24 13:55:02	Future
1.0.12	Language code lookup data	2015-03-24 13:55:02	Future
1.0.13	Language proficiency code lookup data	2015-03-24 13:55:02	Future
1.0.14	Legal representative type code lookup data	2015-03-24 13:55:02	Future
1.0.15	Marital status code lookup data	2015-03-24 13:55:02	Future
1.0.16	Medication status code lookup data	2015-03-24 13:55:02	Future
1.0.17	Obligation policy code lookup data	2015-03-24 13:55:02	Future
1.0.18	Privacy law policy code lookup data	2015-03-24 13:55:02	Future
1.0.19	Problem status code lookup data	2015-03-24 13:55:02	Future
1.0.20	Procedure status code lookup data	2015-03-24 13:55:02	Future
1.0.21	Product form code lookup data	2015-03-24 13:55:02	Future
1.0.22	Provider taxonomy code lookup data	2015-03-24 13:55:02	Future
1.0.23	Purpose of use code lookup data	2015-03-24 13:55:02	Future
1.0.24	Race code lookup data	2015-03-24 13:55:02	Future
1.0.25	Refrain policy code lookup data	2015-03-24 13:55:02	Future
1.0.26	Religious affiliation code lookup data	2015-03-24 13:55:02	Future
1.0.27	Result interpretation code lookup data	2015-03-24 13:55:02	Future
1.0.28	Result status code lookup data	2015-03-24 13:55:02	Future
1.0.29	Route code lookup data	2015-03-24 13:55:02	Future
1.0.30	Sensitivity policy code lookup data	2015-03-24 13:55:02	Future
1.0.31	Social history status code lookup data	2015-03-24 13:55:02	Future
1.0.32	Social history type code lookup data	2015-03-24 13:55:02	Future
1.0.33	State code lookup data	2015-03-24 13:55:02	Future
1.0.34	Target site code lookup data	2015-03-24 13:55:02	Future
1.0.35	Telecom use code lookup data	2015-03-24 13:55:02	Future
1.0.36	Unit of measure code lookup data	2015-03-24 13:55:02	Future
1.0.37	Valueset data	2015-03-24 13:55:03	Future
1.1.0	Add unique constraints to valueset schema	2015-03-24 13:55:03	Future
1.2.0	Add unique constraints to consent and patient	2015-03-24 13:55:03	Future
1.3.0	Tinyint to Bit 1 changes	2015-03-24 13:55:05	Future

For more, please read <http://flywaydb.org/documentation/commandline/>

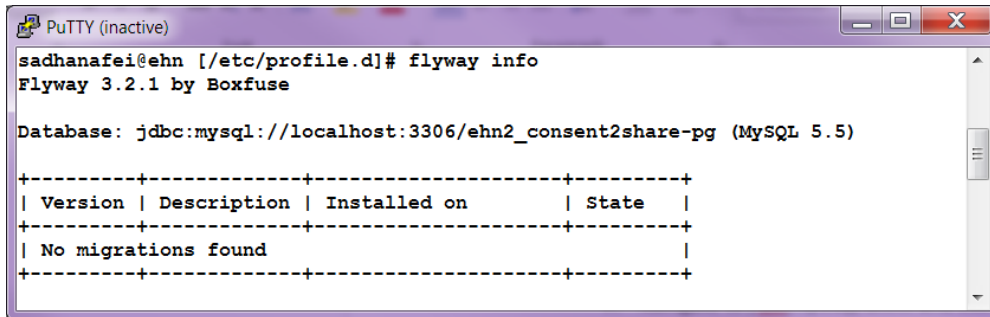
3. Linux Installation

1. Create a directory flyway under user.
`mkdir flyway`
2. Get the latest flyway binaries (refer to <http://flywaydb.org/getstarted/download.html>).
`sudo wget https://bintray.com/artifact/download/business/maven/flyway-commandline-3.2.1-linux-x64.tar.gz`
3. Untar the gz under flyway directory. This will create a flyway-3.2.1 directory under /usr/flyway
`sudo tar -zxvf flyway-commandline-3.2.1-linux-x64.tar.gz`
4. Remove the tar.gz
`rm -rf flyway-commandline-3.2.1-linux-x64.tar.gz`
5. Create a softlink to the path /usr/flyway/flyway-3.2.1 as /usr/flyway/latest
`ln -s /usr/flyway/flyway-3.2.1/ /usr/flyway/latest`
6. create a file called flyway.sh with following contents under /etc/profile.d

```
#!/bin/sh
#!/ Set FLYWAY_HOME environment variable to point to latest FLYWAY path

export FLYWAY_HOME=/usr/flyway/latest
export FLYWAY_HOME

PATH=$PATH:$FLYWAY_HOME
export PATH
```
7. Copy 'mysql-connector-java-5.1.26-bin.jar' to flyway-3.2.1\drivers
8. Go to directory flyway-3.2.1\conf directory and set up configuration
Note: following is using local information for sample
 - a. flyway.url=jdbc:mysql://localhost:3306/ehn2_consent2share-bl
 - b. flyway.user=username
 - c. flyway.password=password
9. Navigate to /usr/flyway/latest and execute the following
`sudo chmod +x flyway`
10. Verify flyway installation by executing info command(flyway info)



```
sadhanafei@ehn [/etc/profile.d]# flyway info
Flyway 3.2.1 by Boxfuse

Database: jdbc:mysql://localhost:3306/ehn2_consent2share-pg (MySQL 5.5)

+-----+-----+-----+-----+
| Version | Description | Installed on | State |
+-----+-----+-----+-----+
| No migrations found |
+-----+-----+-----+-----+
```