# MiSAR Manual:

**Functionality:** The MiSAR parser generates the MiSAR Platform Specific Model (PSM) of a microservice software system. Currently the parser is able to analyse Java projects with Docker and SpringCloud frameworks. Therefore, the parser can receive Java, YAML and XML files such as docker-compose.yml and pom.xml. The Spring boot framework and technologies include Consul, Eureka, MongoDB, MySQL, Neo4j Graph database, OAuth2 and RabbitMQ.

THIS MANUAL IS BUILT AROUND USAGE ON A WINDOWS OPERATING SYSTEM.

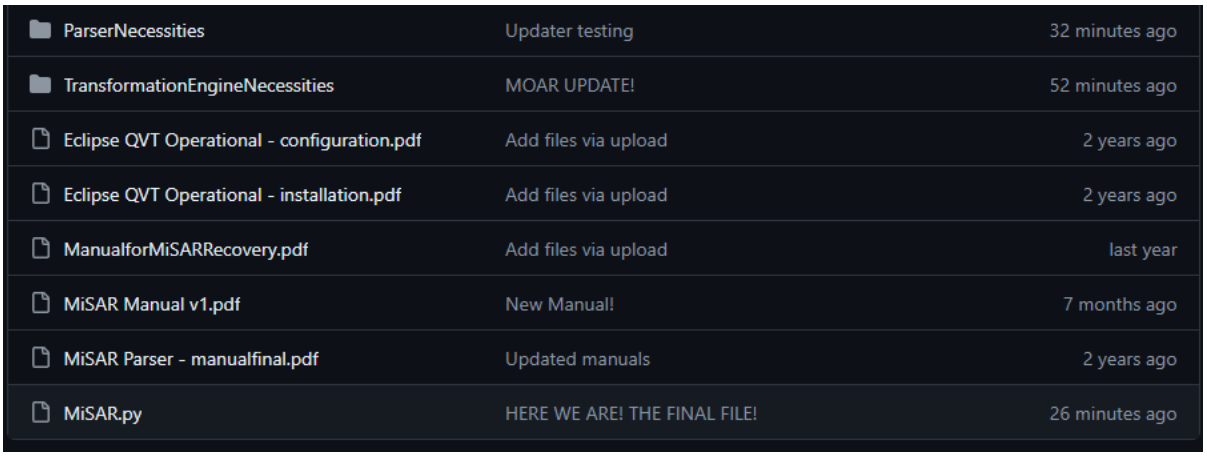
## Installation Guidelines:

The MiSAR parser is one of the many programs built into MisarIntegration.py.

1) Install Python 3.11: https://www.python.org/downloads/release/python-3110/

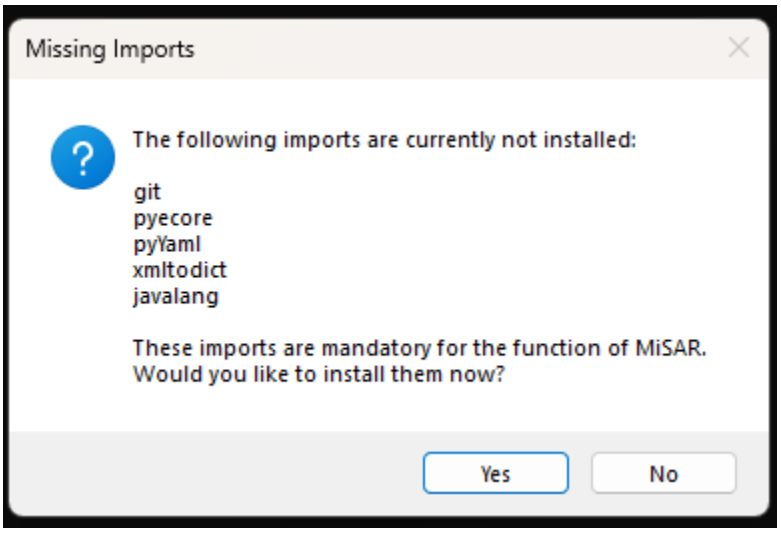
2) From the GitHub repository: https://github.com/MicroServiceArchitectureRecovery/MiSAR-Parser-and-Model-Transformation

   Download MiSAR.py. You can save this anywhere you like on your computer.




3) Run the newly downloaded MiSAR.py using Python 3.11

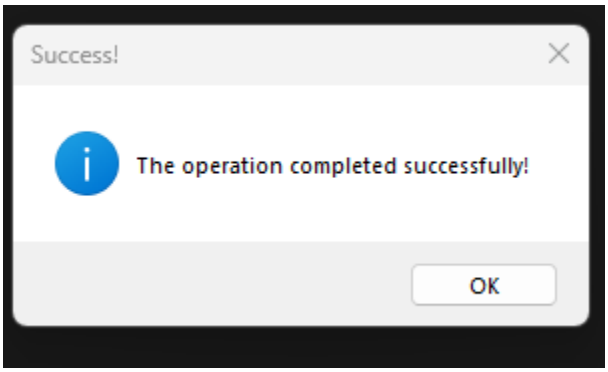
4) Upon running for the first time, you should see this popup appear:



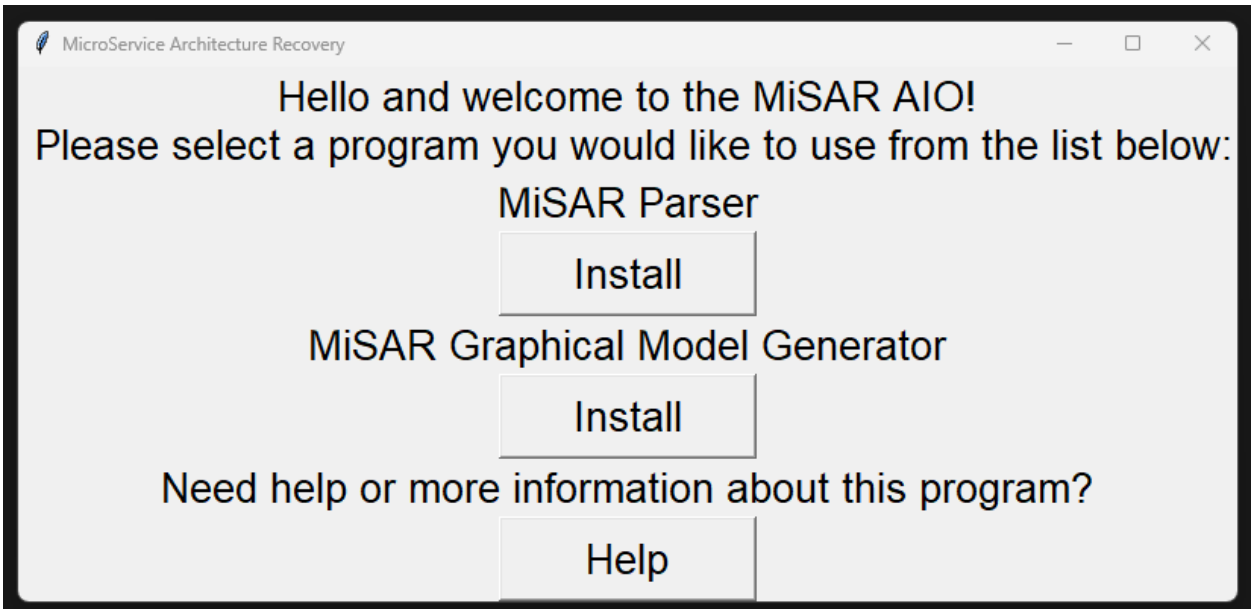

Select yes to install the required modules. If you select no, you will be prompted again to install the modules whenever you try to install the Parser, Transformation Engine, or Graphical Model Generator.

Installation of all modules related to MiSAR are entirely automatic, so you will not have to rely on using CMD to install related modules.

After installation has been completed, a popup will display showing so:

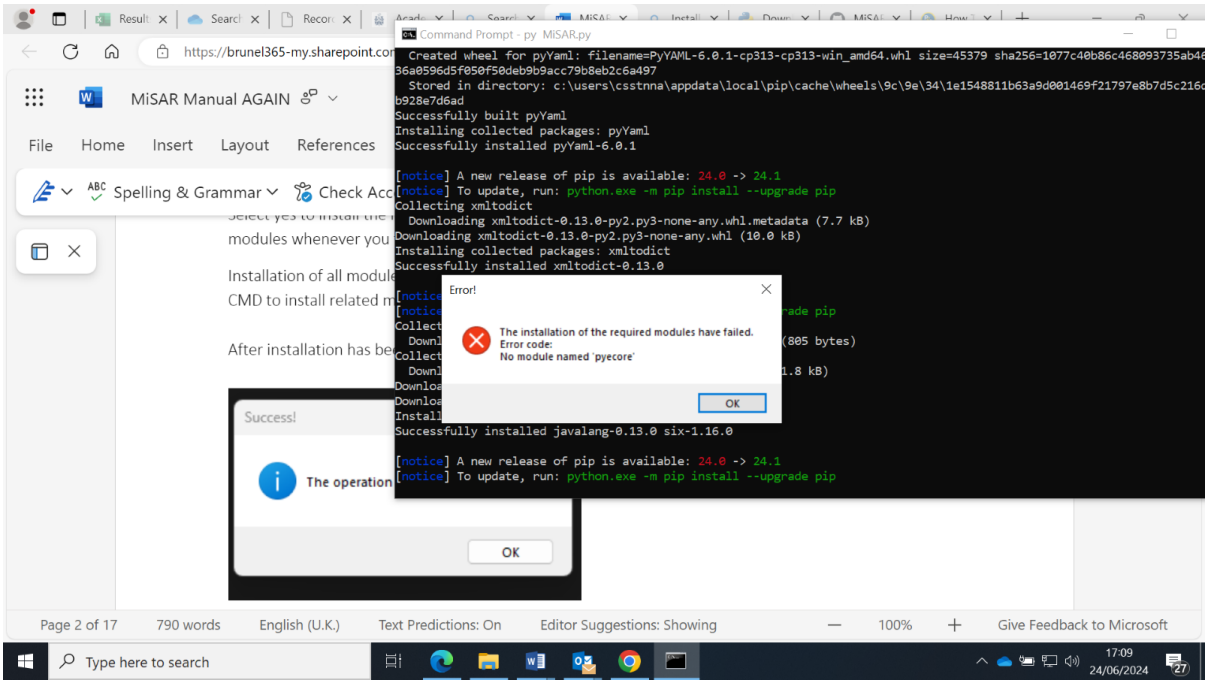



Followed by the main menu appearing:




(OPTIONAL)

If one or more modules fail to install for any reason:

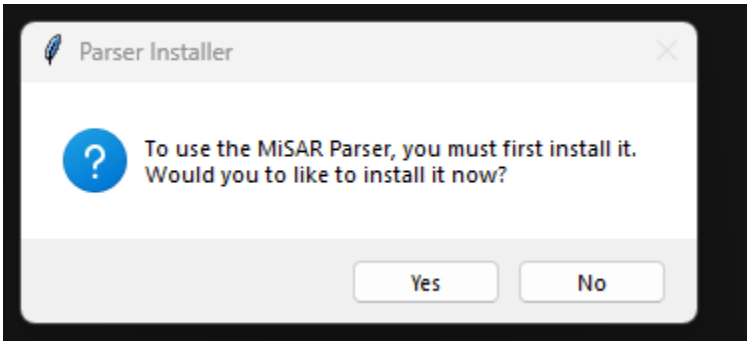



Using pyecore as the example module here. Open up your command prompt, and type in the following command:

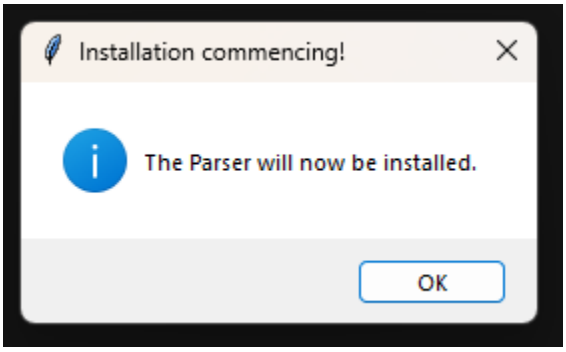
pip install pyecore

Repeat this process for any other modules if any that also failed to install.

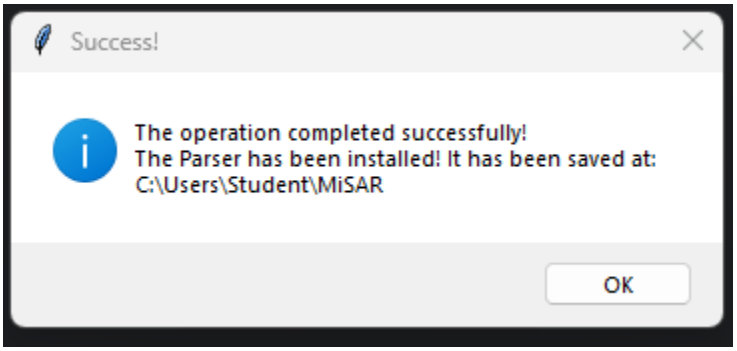
5) If it is your first time using the MiSAR Parser, click the “Install” button below the “MiSAR Parser” text label. The following popup will appear once that happens:




6) Click yes to install the Parser, the following popup will appear:



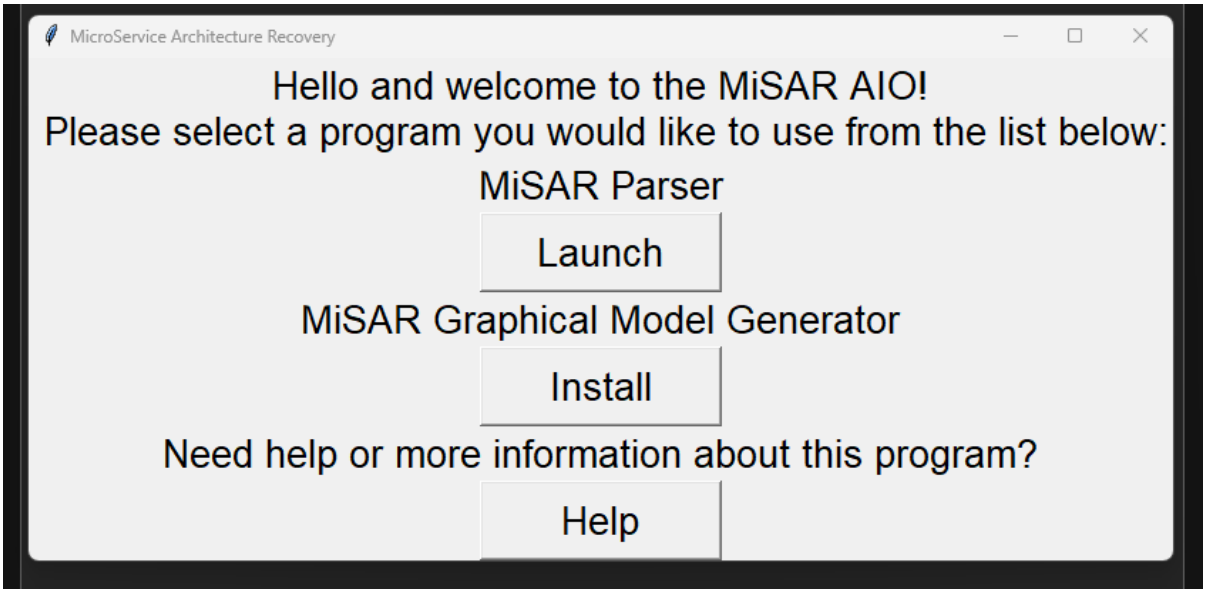

7) After the parser has been installed, you will be notified about its success alongside the location of where the parser has been installed.




The MiSAR folder is the master folder in which the primary components of

MiSAR, like the Parser and Graphical Model Generator are stored.

8) The parser will then be ready to use:

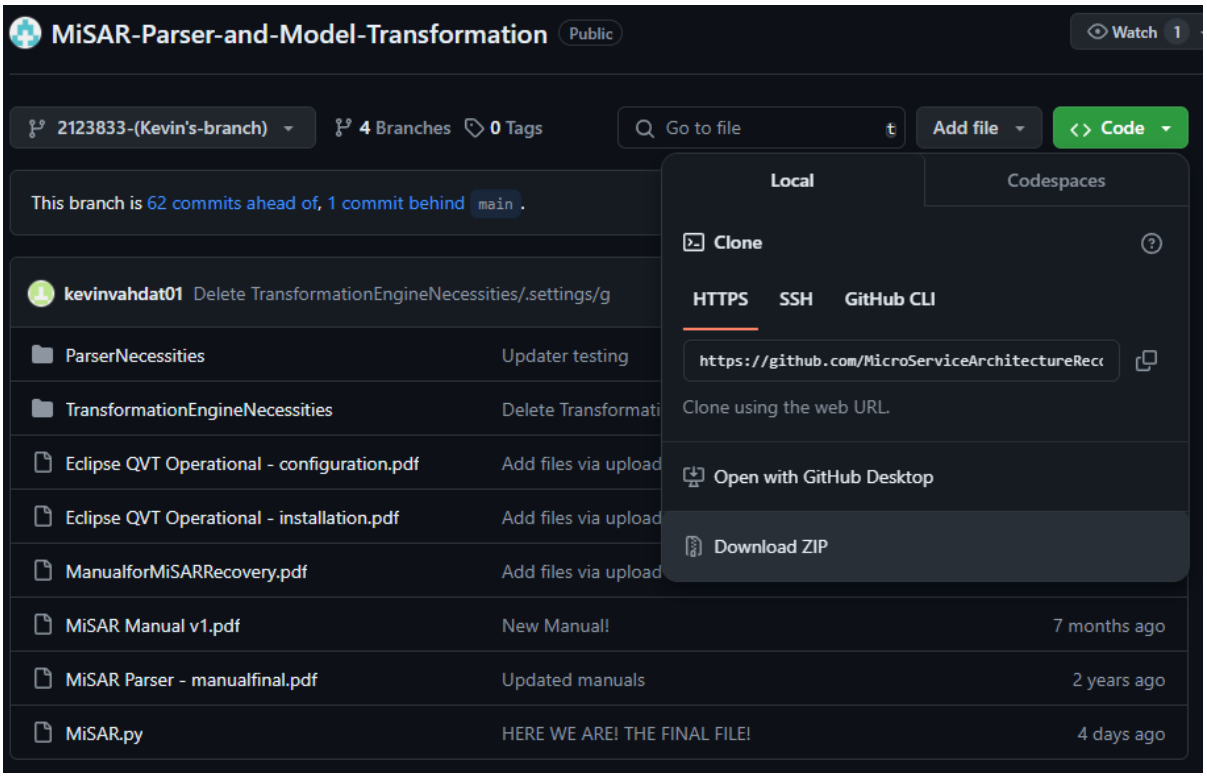



This method of installation is optional

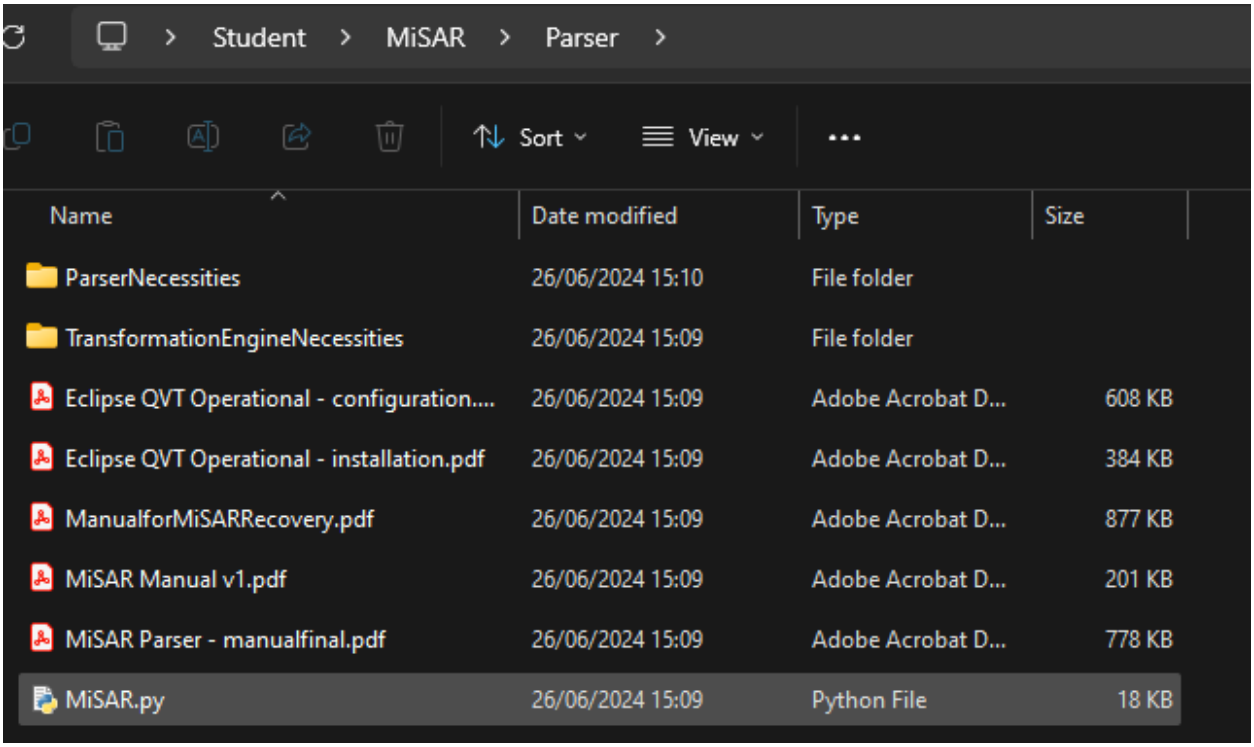
1. If you would like to install the parser manually, then navigate your way to the github shown here:

https://github.com/MicroServiceArchitectureRecovery/MiSAR-Parser-and-Model-Transformation

2. Next, go to “Code” and download the branch as a ZIP file.

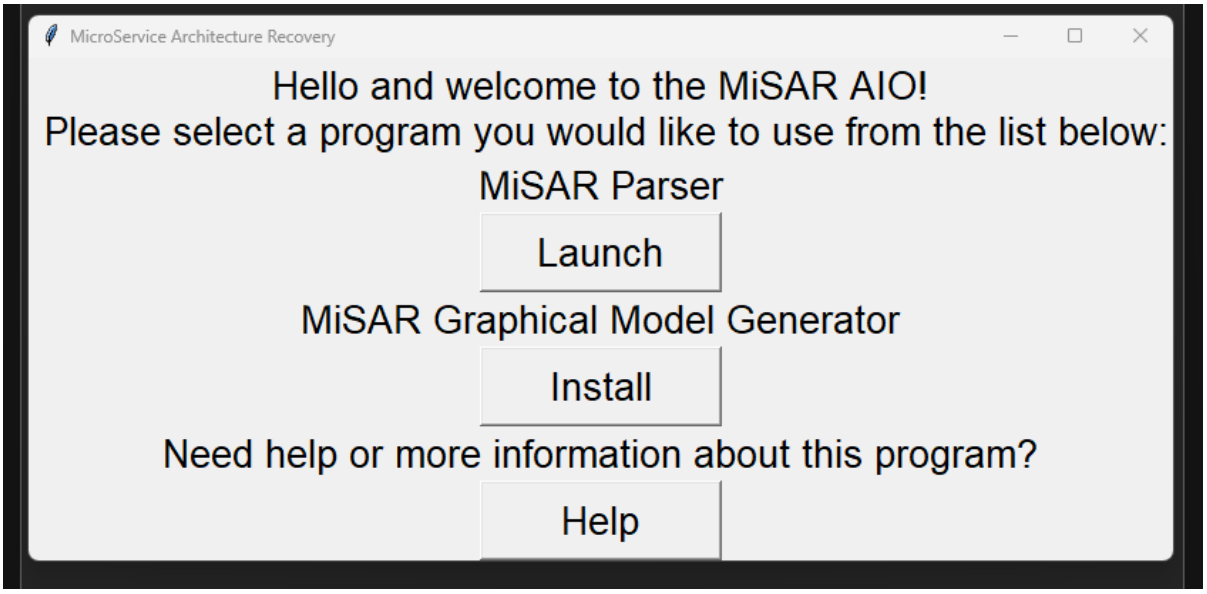



3. Create a folder named ”MiSAR” in C:\Users\(YourAccountName), and within that folder, create another folder called ”Parser”. Extract the contents of the ZIP file there, ensuring it looks like this:




Afterwards, run the MiSAR.py that you saved on step 2, NOT THE ONE IN THE FOLDER YOU CREATED.

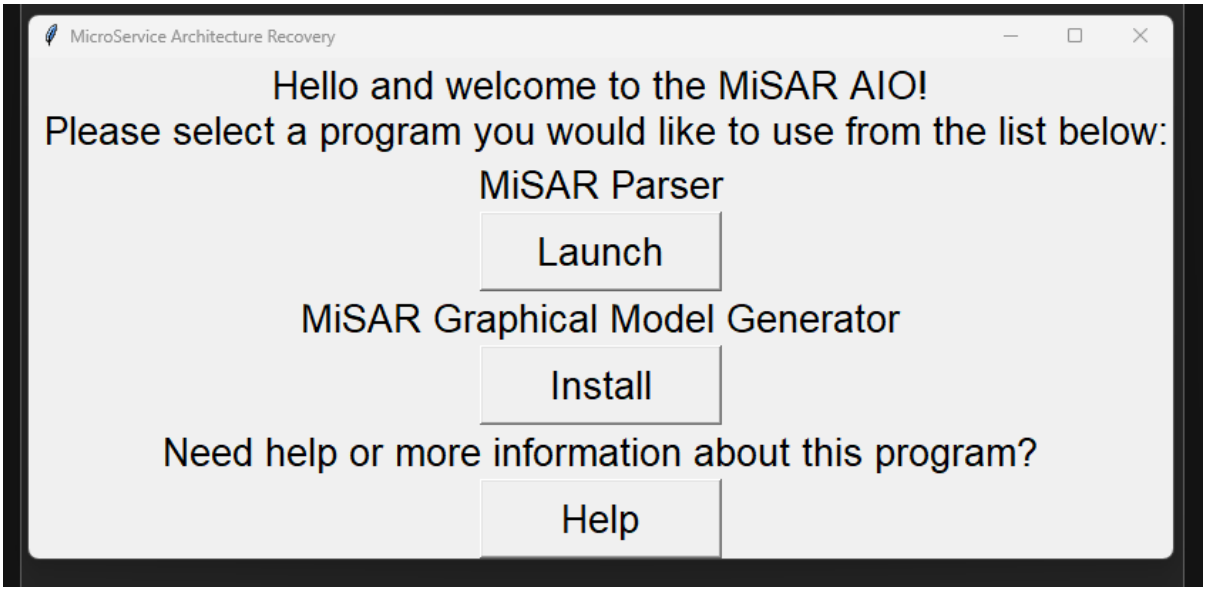
Once this is done, you are good to go:



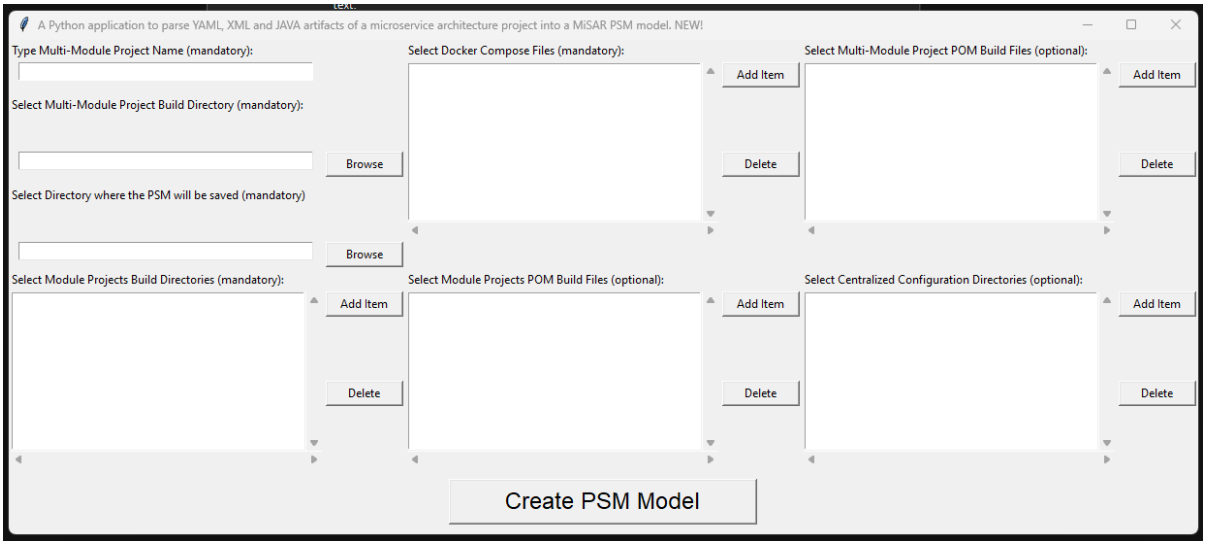

## User Guidelines:

1) Download all source files of a microservice-based project (along with central configuration files if any) to a local drive.

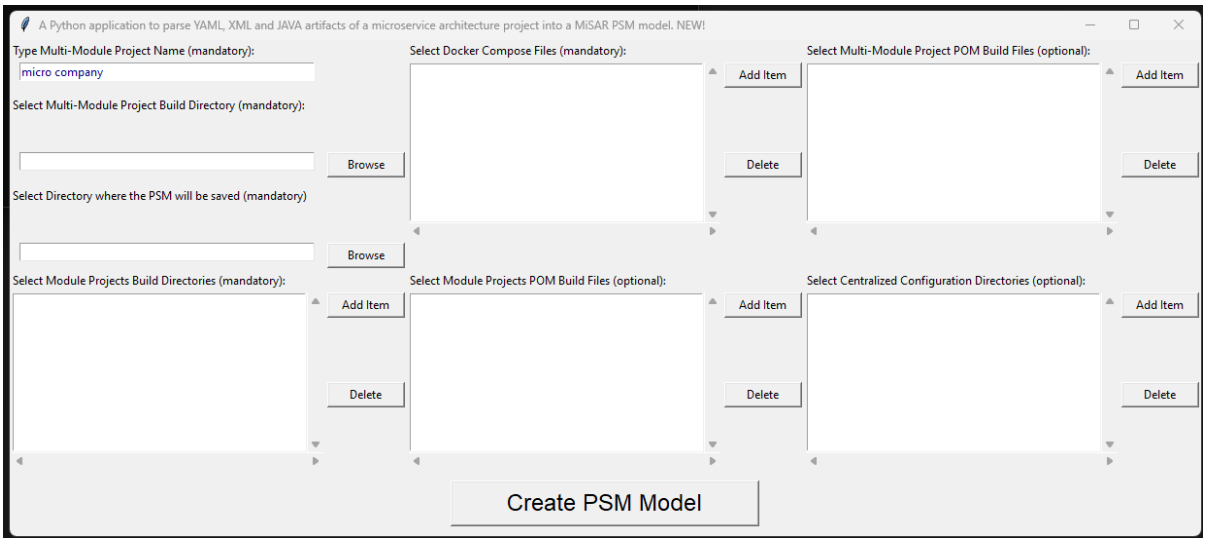
2) Click the “Launch” button underneath the MiSAR Parser text to launch the parser:



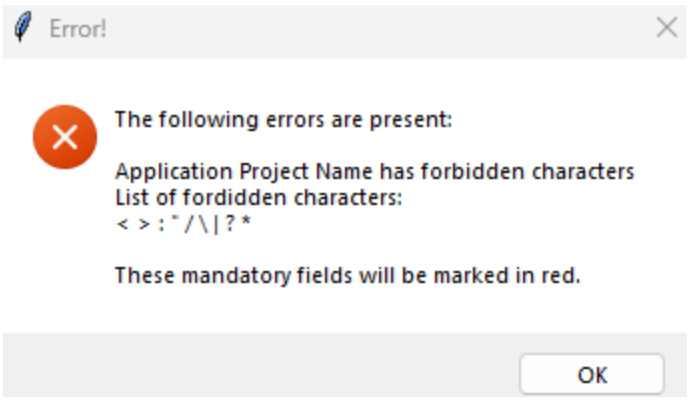
The MiSAR Parser GUI will load shortly after pressing the "Launch" button below the MiSAR Parser text:




3) Fill in the Multi-Module Project Name field with any name you like, as long as no forbidden or reserved characters used by the OS are present within the name.

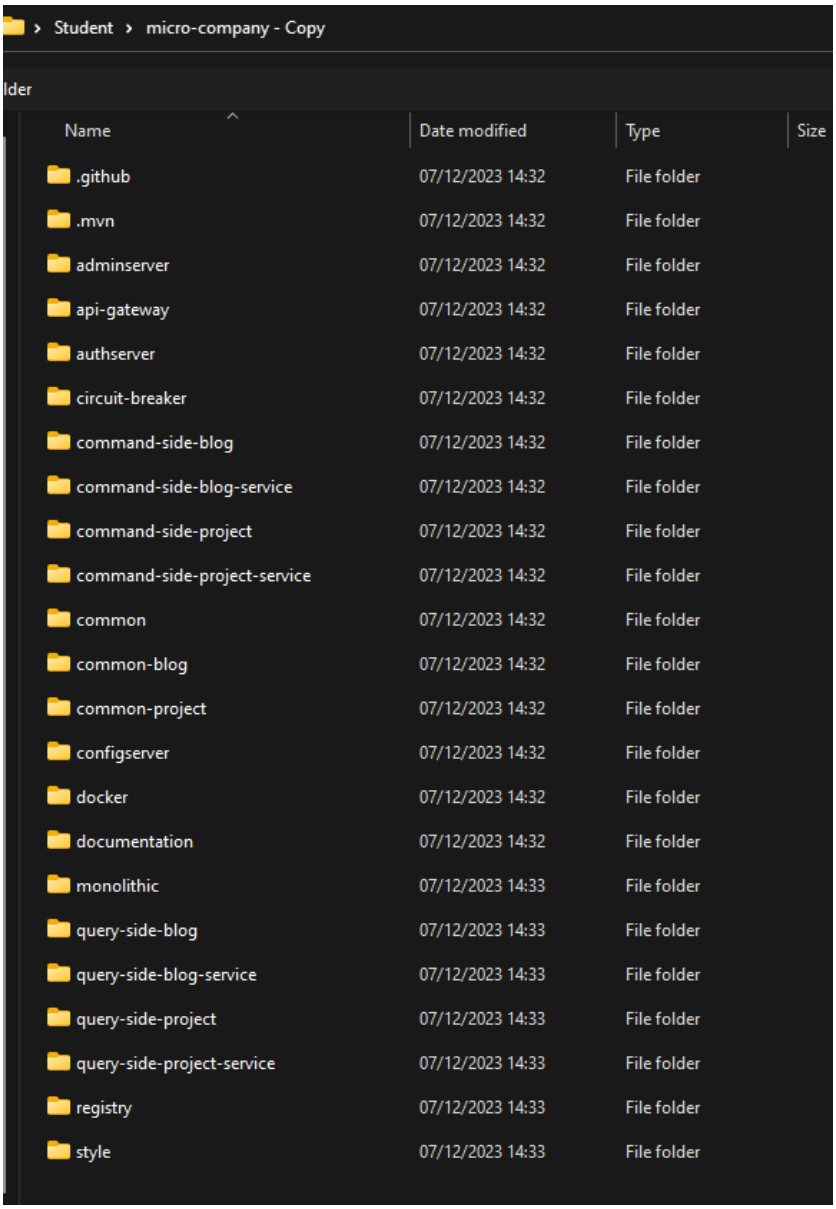


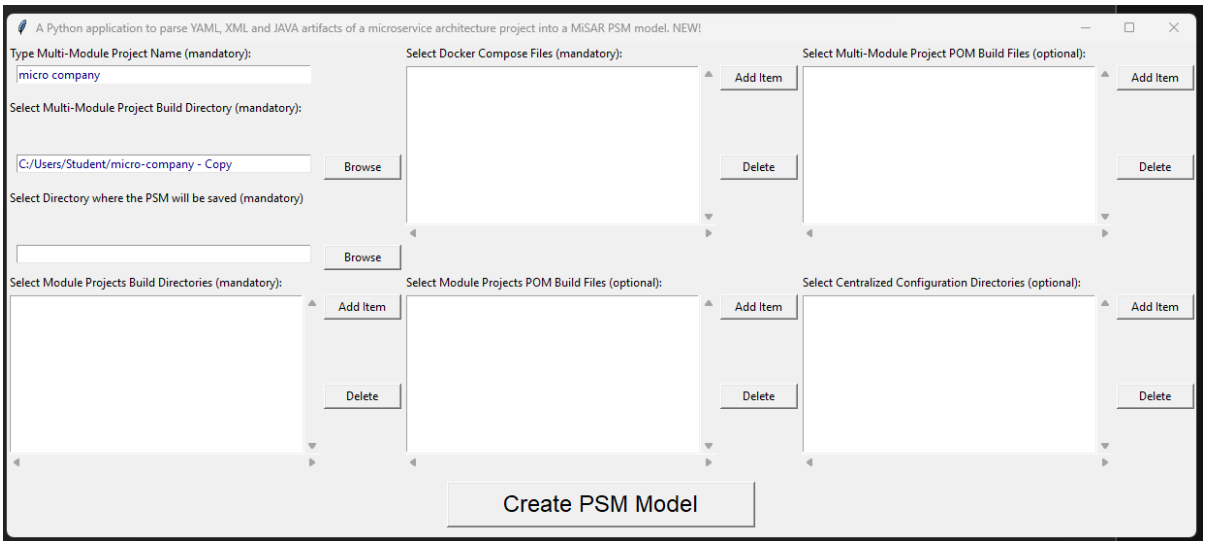

If forbidden characters are detected as the user tries to commence the parsing, then you will get the following error:



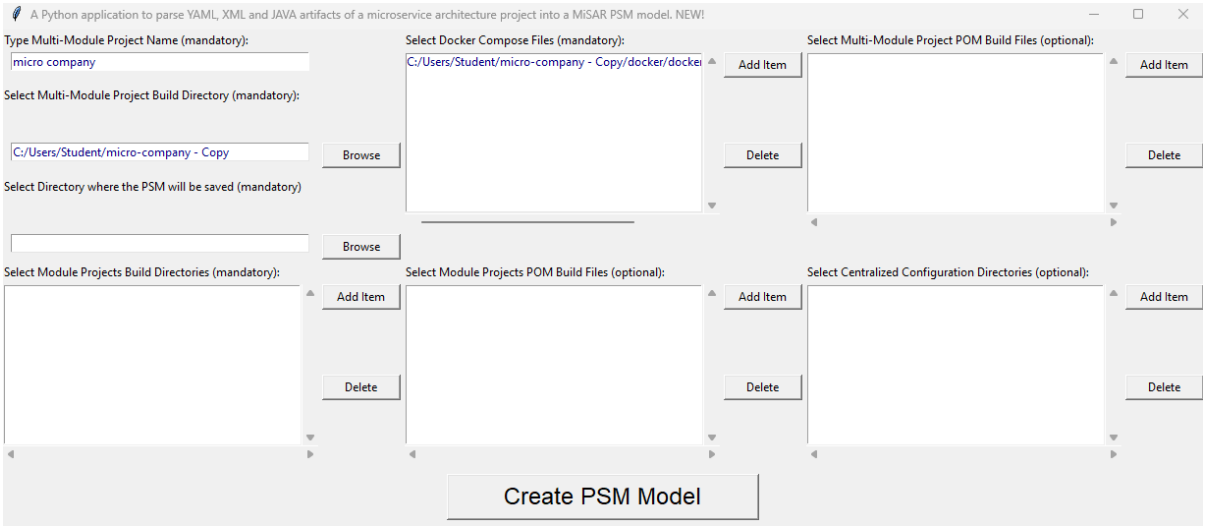

These characters in bold: **< > : ” / \ | ? *** ARE NOT ALLOWED WITHIN NAMES.

4) Select the directory of the microservice-based project you would like to parse and create its PSM (See Step 1) and insert it into the Multi-Module Project Build Directory field. Ensure it’s the master folder and not a folder of the master folder, because the Parser operates on the basis that the microservices are present as shown:

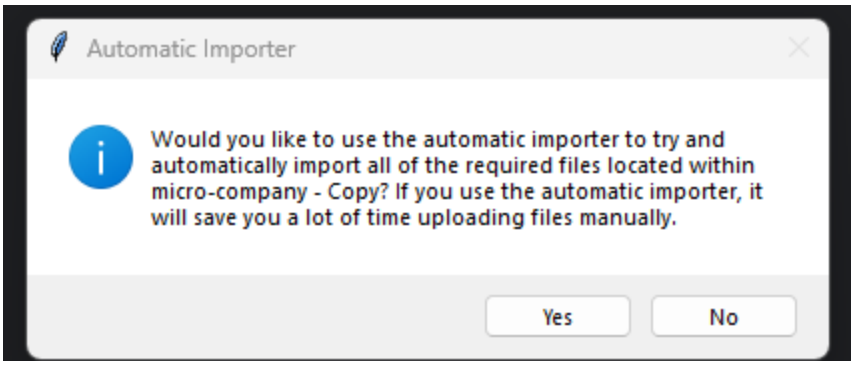

Student › micro-company - Copy
lder
Name
Date modified
Type
Size
.github 07/12/2023 14:32 File folder
.mvn 07/12/2023 14:32 File folder
adminserver 07/12/2023 14:32 File folder
api-gateway 07/12/2023 14:32 File folder
authserver 07/12/2023 14:32 File folder
circuit-breaker 07/12/2023 14:32 File folder
command-side-blog 07/12/2023 14:32 File folder
command-side-blog-service 07/12/2023 14:32 File folder
command-side-project 07/12/2023 14:32 File folder
command-side-project-service 07/12/2023 14:32 File folder
common 07/12/2023 14:32 File folder
common-blog 07/12/2023 14:32 File folder
common-project 07/12/2023 14:32 File folder
configserver 07/12/2023 14:32 File folder
docker 07/12/2023 14:32 File folder
documentation 07/12/2023 14:32 File folder
monolithic 07/12/2023 14:33 File folder
query-side-blog 07/12/2023 14:33 File folder
query-side-blog-service 07/12/2023 14:33 File folder
query-side-project 07/12/2023 14:33 File folder
query-side-project-service 07/12/2023 14:33 File folder
registry 07/12/2023 14:33 File folder
style 07/12/2023 14:33 File folder



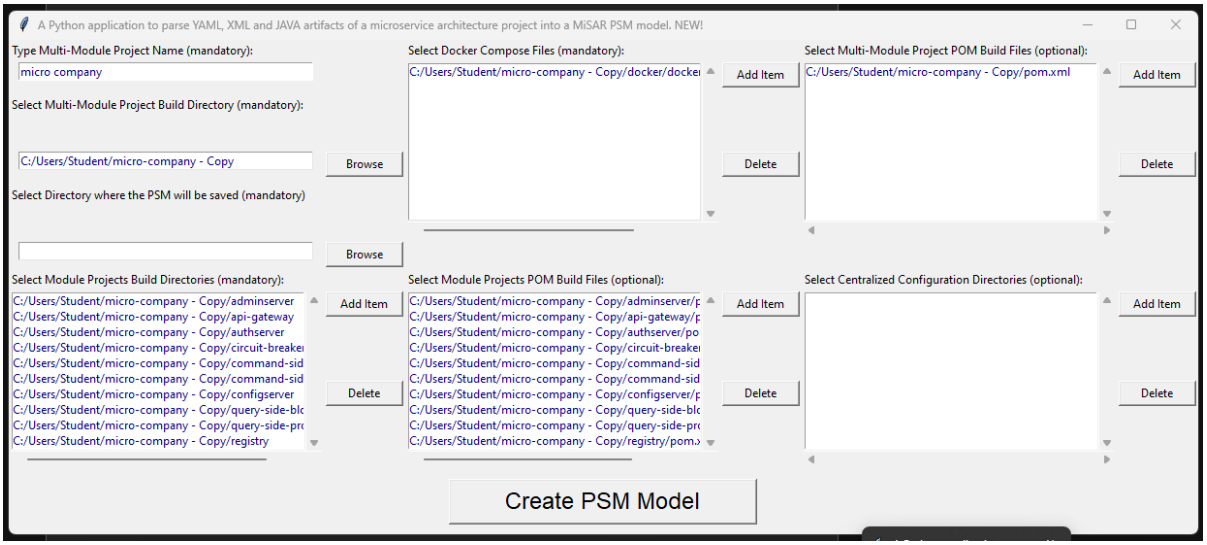

5) Afterwards, select the Docker compose files you would like to import into the parser:




When that is done, the automatic importer will have the requirements fulfilled. The automatic importer attempts to save a lot of time in the next steps by importing the required microservice files in their corresponding segments. This is an optional feature:

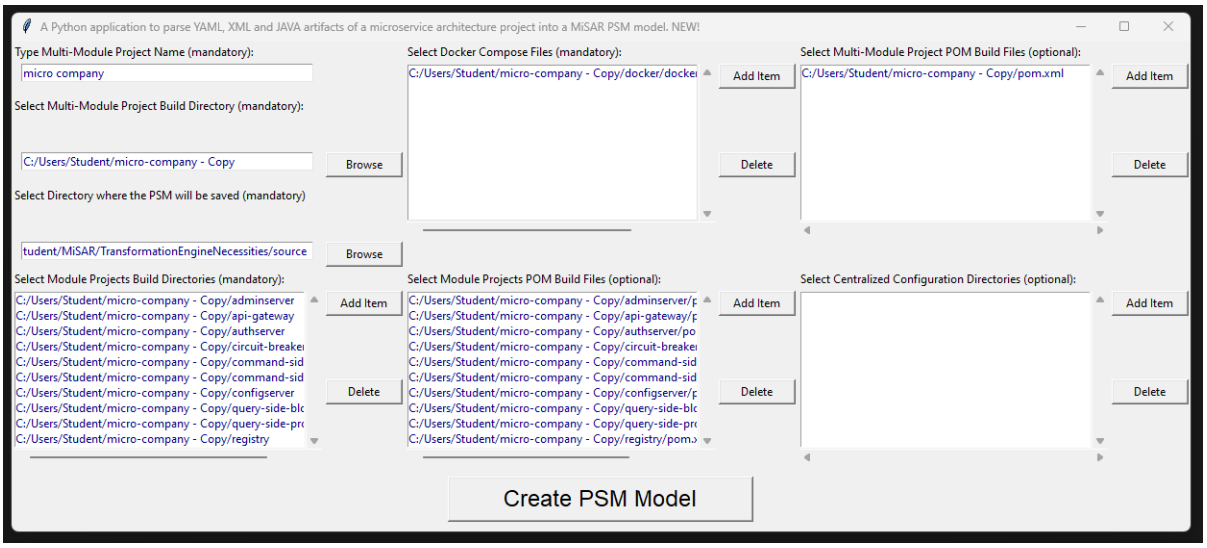



Selecting yes will have all the required files imported into the parser:




If you select No, then you will have to use the Add Item buttons for every single segment.

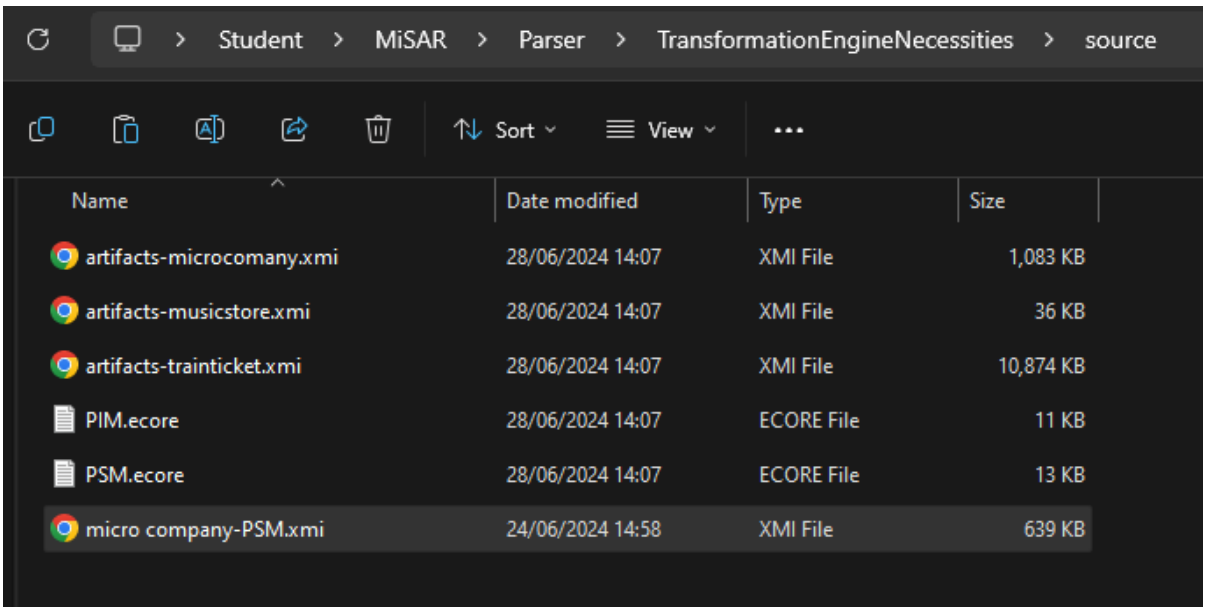
6) Afterwards, select the output directory in which you would like to save the .xmi file to be produced. You can change this directory at any step of importing the files. We recommend for further uses of the PSM file, in the model transformation engine you save it in MiSAR\Parser\TransformationEngineNecessities\source



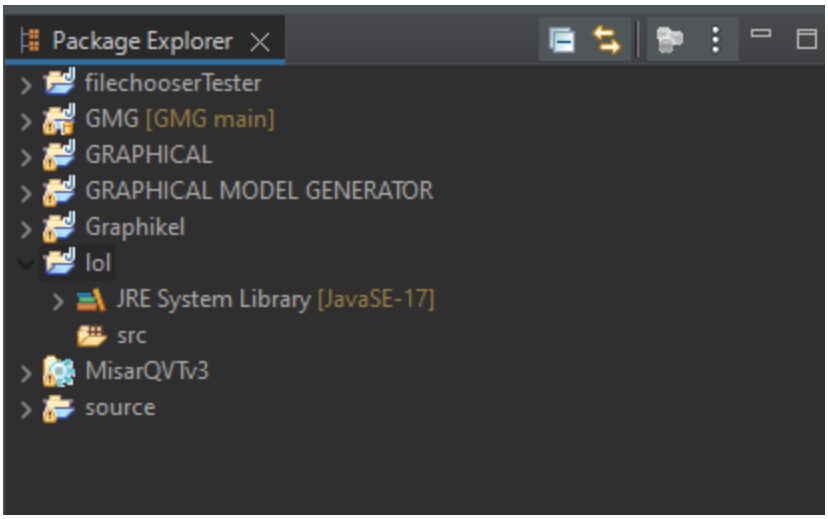

7) When that is done, you can click “Create PSM Model” to parse the result:

The output produced is the PSM file with the name of Multi-ModuleProject-PSM.xmi. You will find it in the directory you selected in step 7

In this case, it’s C:\Users\Student\MiSAR\Parser\TransformationEngineNecessities\source:

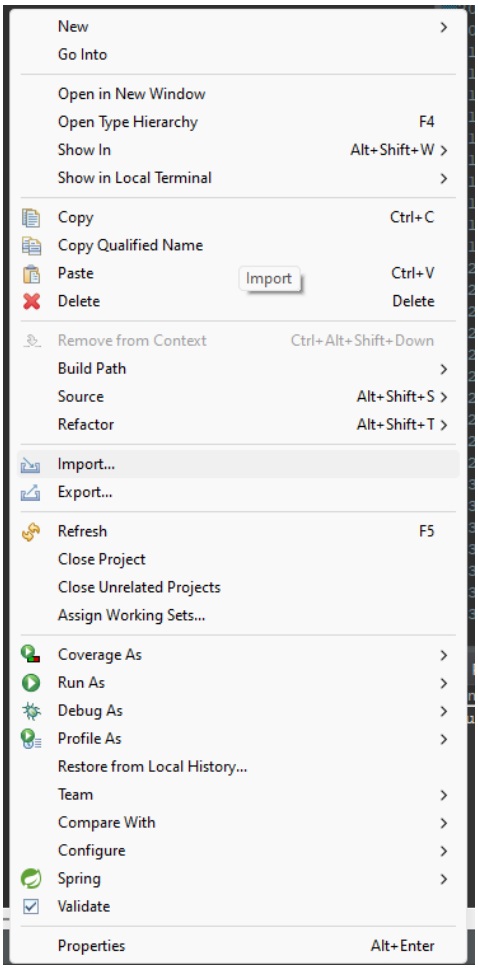



8) After parsing succeeds, you can now view the tree results! To do this open the Eclipse IDE (version 2022-09), create a new **Java** project, naming it whatever you like in the process:

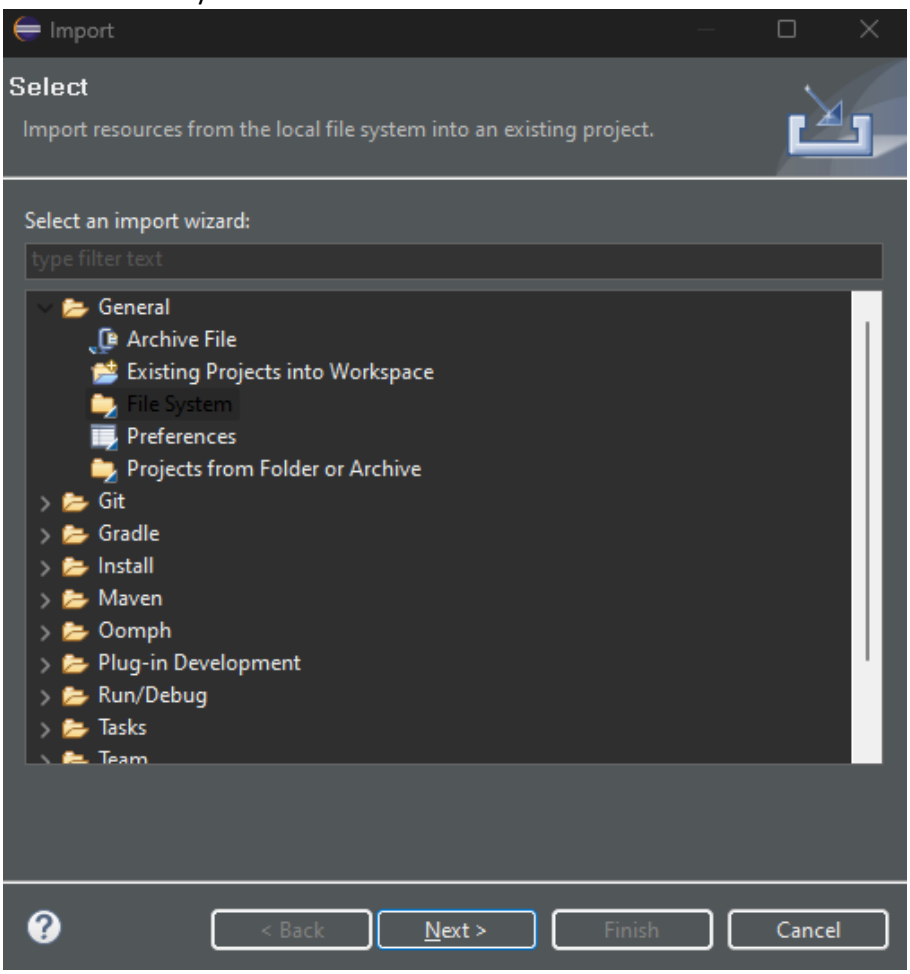



For this example, I named the project ”lol”, using Eclipse IDE for Java developers, version 2022-09 available at: https://www.eclipse.org/downloads/packages/release/2022-09/r/eclipse-ide-java-developers

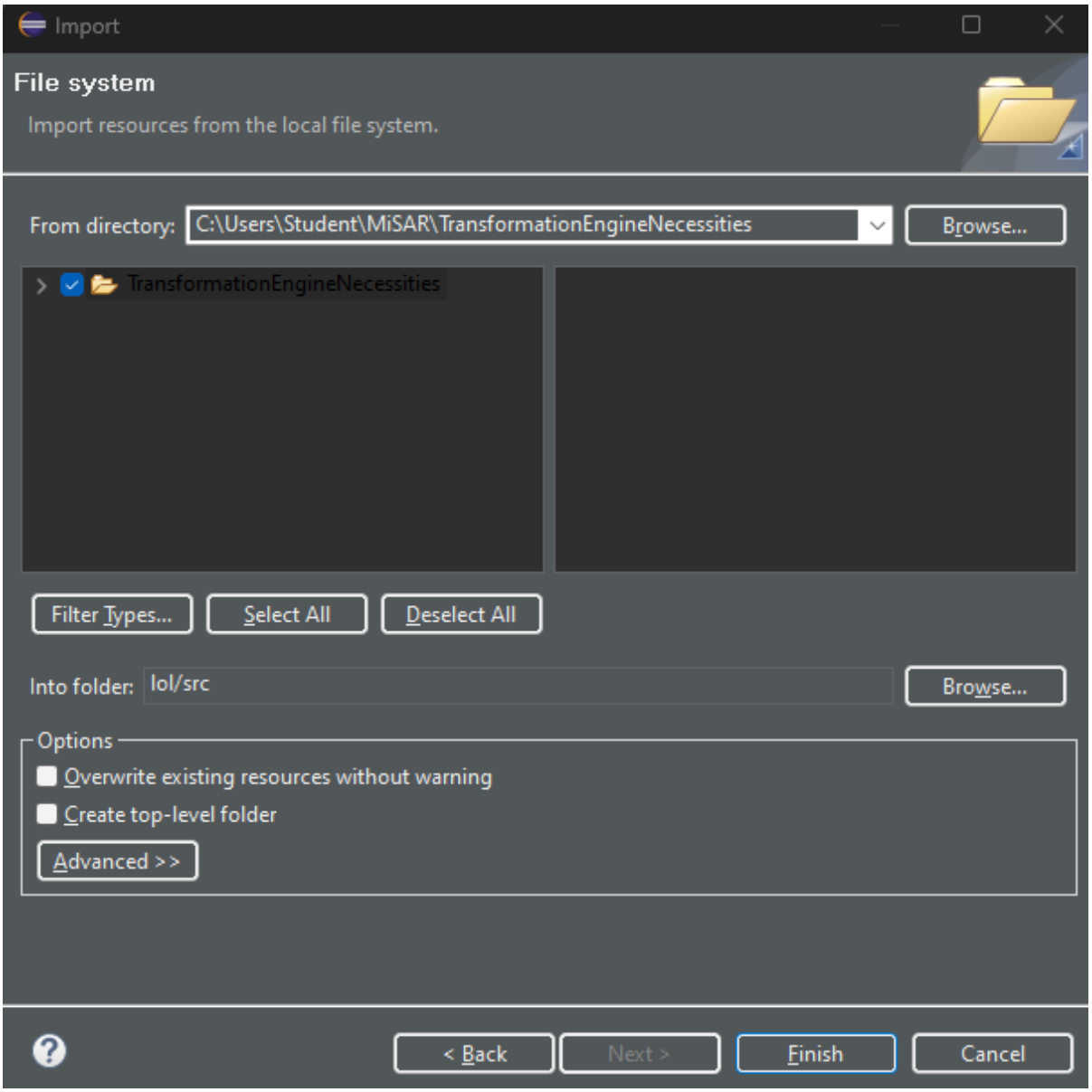
9) Right click on the Package Explorer area and click “Import”


New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Import
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Assign Working Sets...
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Team
Compare With
Configure
Spring
Validate
Properties Alt+Enter

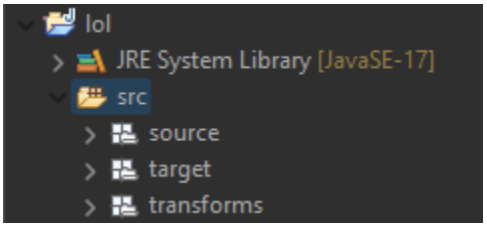
10) Select the “file system” wizard and then click Next:



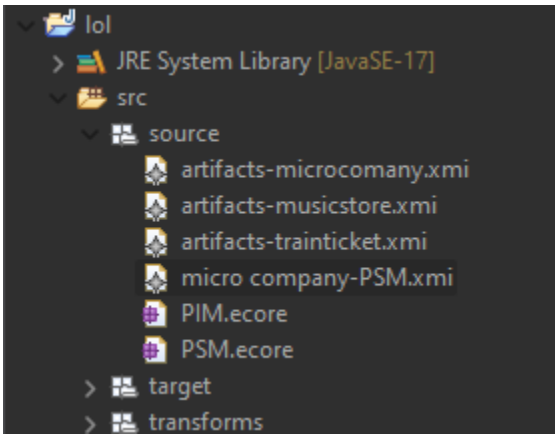

11) Browse the directory “C:\Users\Student\MiSAR\TransformationEngineNecessities”, and check the entire folder, afterwards, click “Finish”.



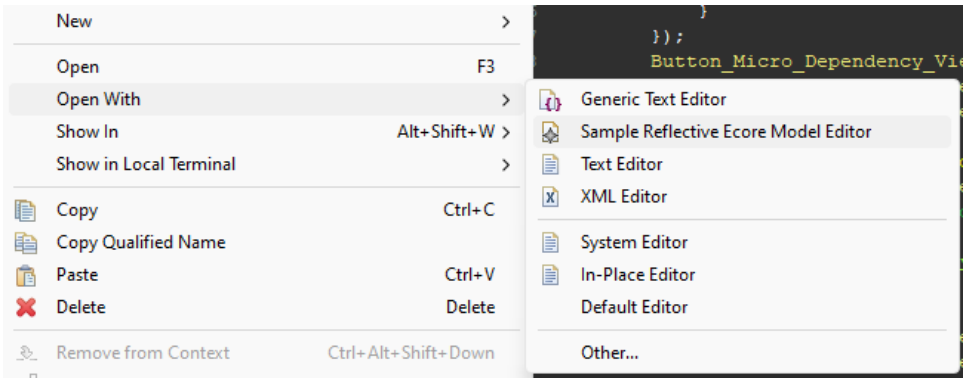

The end result should look like this:



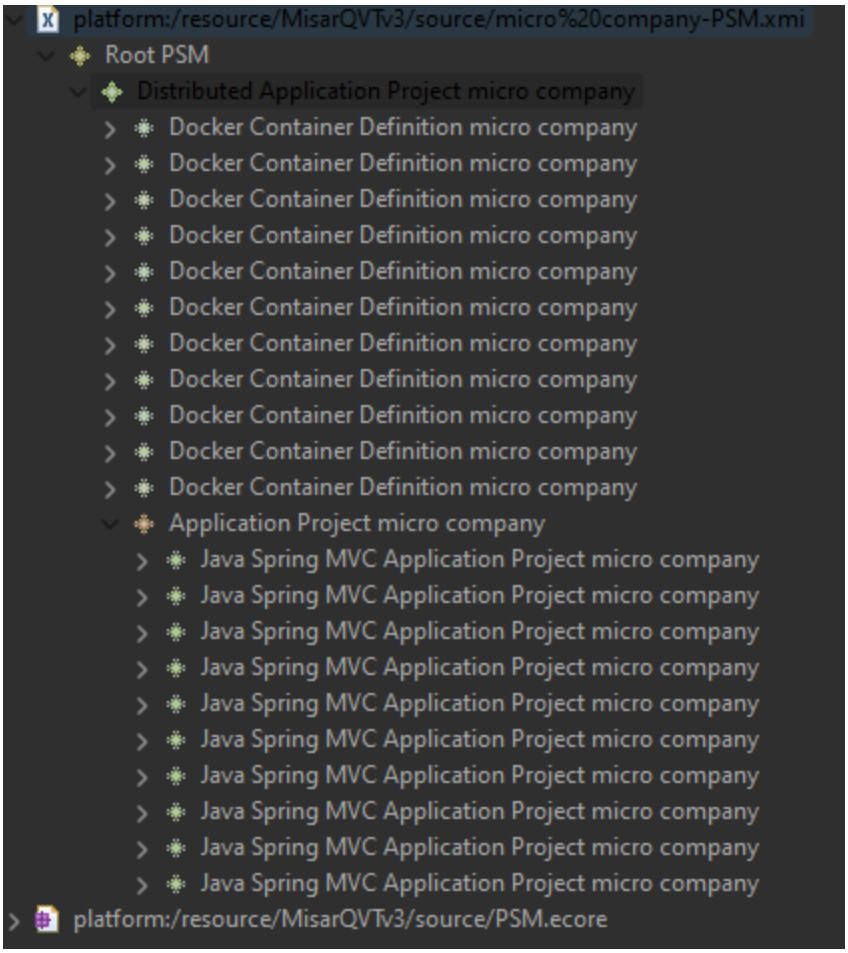
12) After the folder has been imported, you can now open the newly generated xmi file in tree-view. To do this, find the xmi file located in “source”



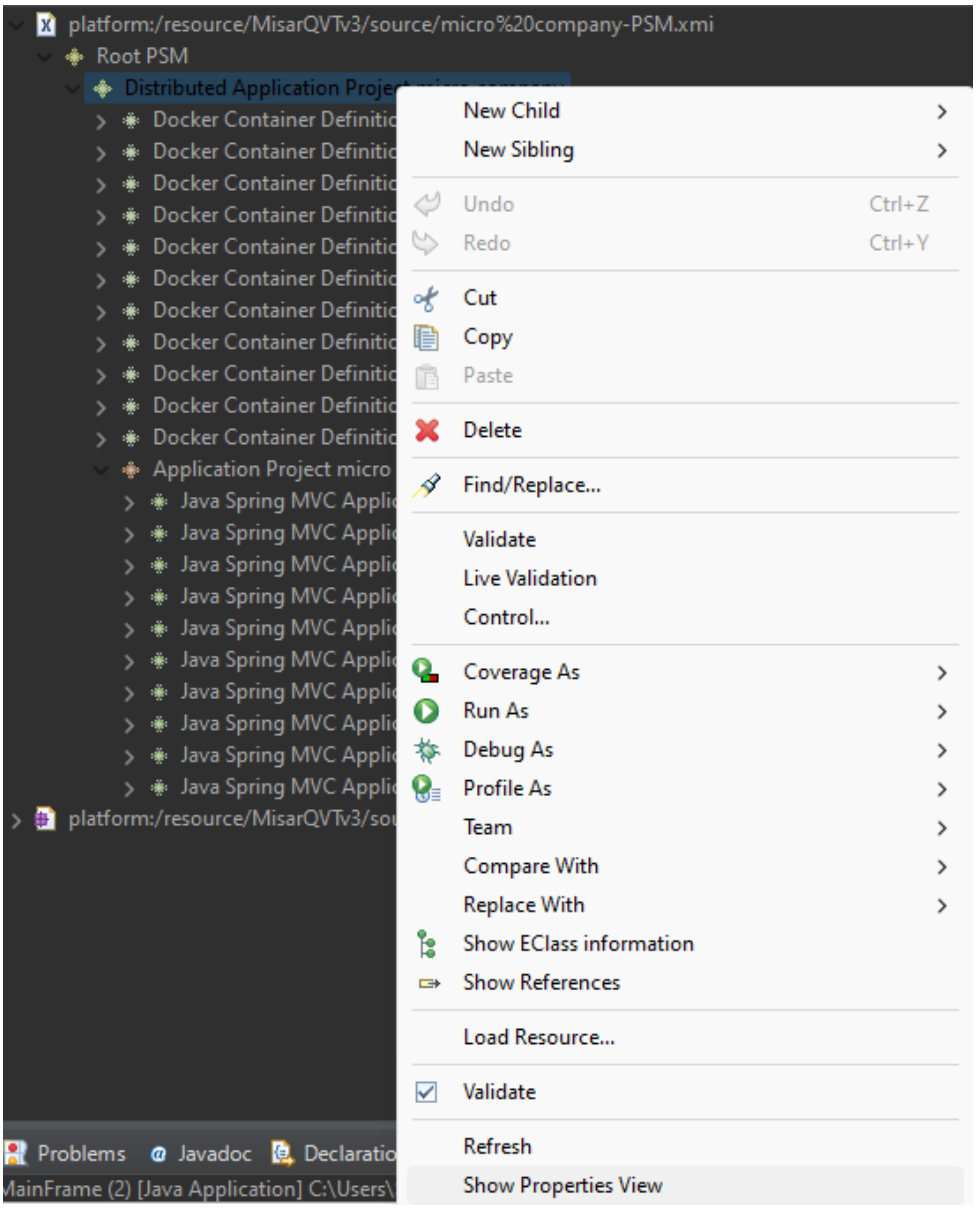

And then Right click the newly generated file, **Open With -> Sample Reflective Ecore Model Editor**

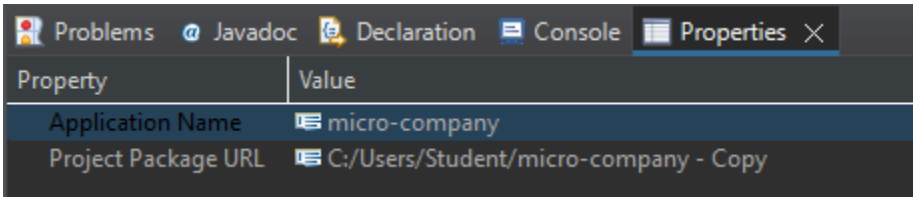



**And afterwards you should see the tree view of the PSM:**




13) To view one of the attributes of a PSM instance element, right click the element, then click on **Show Properties View**


platform:/resource/MisarQVTv3/source/micro%20company-PSM.xmi
Root PSM
Distributed Application Project
Docker Container Definitio
Application Project micro
Java Spring MVC Applic
platform:/resource/MisarQVTv3/sou
New Child
New Sibling
Undo Ctrl+Z
Redo Ctrl+Y
Cut
Copy
Paste
Delete
Find/Replace...
Validate
Live Validation
Control...
Coverage As
Run As
Debug As
Profile As
Team
Compare With
Replace With
Show EClass information
Show References
Load Resource...
Validate
Refresh
Show Properties View
Problems
Javadoc
Declaratio
MainFrame (2) [Java Application] C:\Users\


Problems
Javadoc
Declaration
Console
Properties
Property
Value
Application Name
micro-company
Project Package URL
C:/Users/Student/micro-company - Copy