SAI Final Assignment:

Graduation Approval

Table of Contents

[1. Introduction 2](#_Toc8737810)

[2. Start-up Code 4](#_Toc8737811)

[3. Screen Shots 4](#_Toc8737812)

[4. Assignment 5](#_Toc8737813)

[4. Grading Criteria 6](#_Toc8737814)

[5. Submission and Deadlines 7](#_Toc8737815)

# 1. Introduction

In this assignment you will integrate a system of several applications for approving start of graduation projects at Fontys ICT school. The whole scenario for the Graduation Approval system is shown in Figure 1.

**Graduation Client application** is a Java desktop application. It is used by students for requesting the approval of their graduation project (*GraduationClientRequest*). The student must provide the student number, company name and project title.

The system first retrieves *StudentInfo* from the **Student Administration** (RESTfull service) for the student with student number specified in the *GraduationClientRequest*. *StudentInfo* contains the following information about the student: (1) total achieved ECs in the semester 7, and (2) name of the school mentor of this student.

Next, a *GraduationApprovalRequest* is created and forwarded to several **Graduation Approval** applications. The *GraduationApprovalRequest* contains the same student number, company name and project title from the original *GraduationClientRequest* and ECs and mentor name from *StudentInfo*. A *GraduationApprovalRequest* is forwarded to the four Graduation Approval applications according to following rules:

1. All requests are forwarded to Graduation Coordinator to approve the project content.
2. If student has ECs in range [24..29], then the request should also be fowraded to student’s mentor. Mentor approves that the student may start graduation.
3. If student has less than 24 ECs, the request is not forwarded to any Graduation Approval application because the student may not start graduation. This request is automatically rejected.

Table below shows these rules per Graduation Approval application:

|  |  |
| --- | --- |
| graduation approval application | processes *GraduationApprovalRequest* |
| **Graduation Coordinator**  *(approves the project content)* | All requests. |
| **Bert (mentor)**  *(approves that the student is ready for graduation)* | ECs are in range [24..29] and  student’s mentor is Bert |
| **Rafayel (mentor)**  *(approves that the student is ready for graduation)* | ECs are in range [24..29] and  student’s mentor is Rafayel |
| **Chung (mentor)**  *(approves that the student is ready for graduation)* | ECs are in range [24..29] and  student’s mentor is Chung |

Each Graduation Approval application sends back *GraduationApprovalReply* containing the approval decision (approved or rejected) and name of the person who rejected (rejectedBy). If the decision is approved, then field rejectedBy should be left empty.

After replies of all Graduation Approval applications are received, a *GraduationClientReply* is sent back to the **Graduation Client** application. This *GraduationClientReply* contains fields whether the request was approved (if all Graduation Approval applications have approved) or rejected (if at least one Graduation Approval application has rejected). If *GraduationClientReply* is rejected, then field rejectedBy contains names of all Graduation Approval applications which have rejected it.



Figure 1. The Graduation Approval integration system

# 2. Start-up Code

In **SAI-final-startup-code.zip** you can find:

1. **“graduation-project-plan-gradle”**: IntelliJ IDEA GRADLE project with **client** and **approval** sub-projects modules. Note that the whole project is a GRADLE project, and **client** and **approval** are GRADLE sub-projects (see settings.gradle in the project root)[[1]](#footnote-1). Note that the GUI is creaed in JavaFX, but , it you want, you may re-create the GUI in Java Swing (see the next section which shows the GUI).

**Execution** of the javaFX clients is done by going to the run/debug configurations, and change:

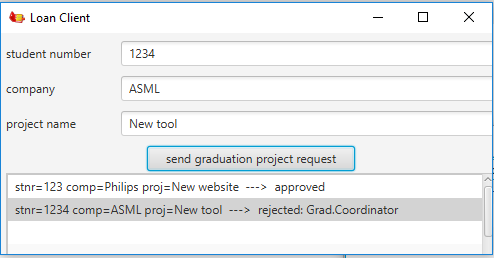
* the settings for ‘VM options’ for each of the approval applications. Have the ‘—module-path’ option point to your local javaFX library directory.
* The ‘working directory’ should point to your local module folder

After these changes, the run configuration can be used.

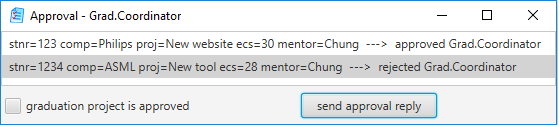
1. **“student-administration-service”**: has war file of a RESTfull service which you should deploy on a web Server (e.g., Apache Tomcat). There are two WAR files: one is built with Java 8 and the other with Java 11. ). The service can be accessed at <http://localhost:8080/administration/>, where you can read a description about how this service works.

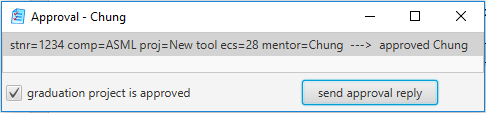
# 3. Screen Shots

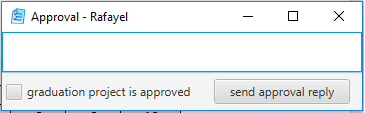
**Graduation Client application:**

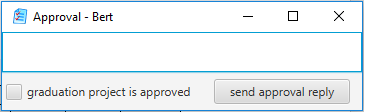


**Graduation Approval applications:**

****

****

****

****

# 4. Assignment

Implement Health Insurance integration system as described in this document. You should make use of the following integration patterns:

* Message Broker
* Correlation Identifier (for asynchronous request-reply communication with JMS),
* Return Address (for asynchronous request-reply communication with JMS)
* Messaging Gateway
* Chained Gateways
* Content-Based Router,
* Content Enricher,
* Recipient List,
* Aggregator, and
* Scatter-Gather.

# 4. Grading Criteria

This assignment is INDIVIDUAL, i.e., it is not allowed to work in groups with other students. The grade you get for this assignment is between 1 and 10, and this will be your grade for the Software Applications Integration (SAI) course.

IntelliJ project(s) including full source code and all necessary libraries (gradle, maven or .jar) must be submitted. All submitted projects must compile and run correctly on the computer of the teacher. If the teacher does not have your full source code, or cannot run your project(s) due to compiling errors, missing libraries or exceptions, then your SAI grade will be 1. Otherwise, SAI grades will be determined based on implemented Application Integration Patterns in the following way:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | SAI grade | | | | |
|  | 6 | 7 | 8 | 9 | 10 |
| The system works correctly with one hospital application | x | x | x | x | x |
| Message Broker | x | x | x | x | x |
| Correlation Identifier | x | x | x | x | x |
| Return Address | x | x | x | x | x |
| Messaging Gateway | x | x | x | x | x |
| Chained Gateways | x | x | x | x | x |
| Content-Based Router |  | x | x | x | x |
| Content Enricher |  | x | x | x | x |
| The system works correctly with three hospital applications |  |  | x | x | x |
| Recipient List |  |  | x | x | x |
| Aggregator |  |  | x | x | x |
| Scatter-Gather |  |  | x | x | x |
| Use of Jeval (or similar) instead of  hard-coded approval routing rules. |  |  |  | x | x |
| Well organized code, with comments, proper variable and method names, no redundant code. |  |  |  |  | x |

# 

# 5. Submission and Deadlines

**Submission of the source code**

The IntelliJ project(s) with full source code and all necessary libraries (via gradle, maven or \*.jar) must be submitted via Canvas. The deadline for submission is set in this Canvas assignment. It is not possible to submit after this deadline. If you do not submit your source code before the deadline, you will not receive a SAI grade in this block (i.e., you will not pass the SAI course in this block).

**Defense of your assignment**

In week 8 or 9 SAI exam is scheduled (see class schedules). During this exam you will speak in person to the teacher about your assignment: you will be asked to explain your code, suggest ideas for improvement, etc. If you are not present during this exam, then you will not get a grade for SAI. It is not possible to mote your exam at another time.

Only students who submitted their source code via Canvas before the deadline specified in Canvas will be invited for this exam. You will receive this invitation with your specific time slot from you teacher several days before the exam. In this invitation it will be specified at which time you should be present for this exam. Each student will have his/her own time slot, and you should and can be present only during your own time slot.

1. If you do not want to work with GRADLE, create a new IntelliJ project (without gradle) and copy-paste the source code of the client and approval sub-projects. [↑](#footnote-ref-1)