

CZ2002 Object Oriented Design & Programming

Group Project

Lab Group XXX Group XXX

Prepared by:

Wu Wenxuan UXXXXXXXX

Tang Hoong Jing U1721417E

Jordon Tan UXXXXXXXX

Yeo Lai Xiang, Leon UXXXXXXXX

Lance Gabriel UXXXXXXXX

Contents

[1. Declaration of Original Work for CZ2002 Assignment 3](#_Toc529992967)

[2. Executive Summary 4](#_Toc529992968)

[3. Design Considerations 5](#_Toc529992969)

[4. UML Class Diagram 6](#_Toc529992970)

[5. UML Sequence Diagram of “Print Student Transcript” 7](#_Toc529992971)

[6. Required Functionalities 8](#_Toc529992972)

[7. Test Cases and Results 9](#_Toc529992973)

## 1. Declaration of Original Work for CZ2002 Assignment

We hereby declare that the attached group assignment has been researched, undertaken, completed and  
 submitted as a collective effort by the group members listed below.

We have honored the principles of academic integrity and have upheld Student Code of Academic  
 Conduct in the completion of this work.

We understand that if plagiarism is found in the assignment, then lower marks or no marks will be  
 awarded for the assessed work. In addition, disciplinary actions may be taken.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Course | Lab Group | Signature/ Date |
| Wu Wenxuan | BCG | SS4 |  |
| Tang Hoong Jing | BCG | SS4 |  |
| Jordon | BCG | SS4 |  |
| Yeo Lai Xiang, Leon | BCG | SS4 |  |
| Lance Gabriel | CS | SS4 |  |

## 2. Executive Summary

This report details our team’s implementation of the *STUDENT COURSE REGISTRATION AND MARK ENTRY (SCRAME)* application, using various Object-Oriented (OO) concepts taught in CZ2002. The objective of this console-based application is to facilitate the course registration process for both professors and undergraduate students.

## 3. Design Considerations

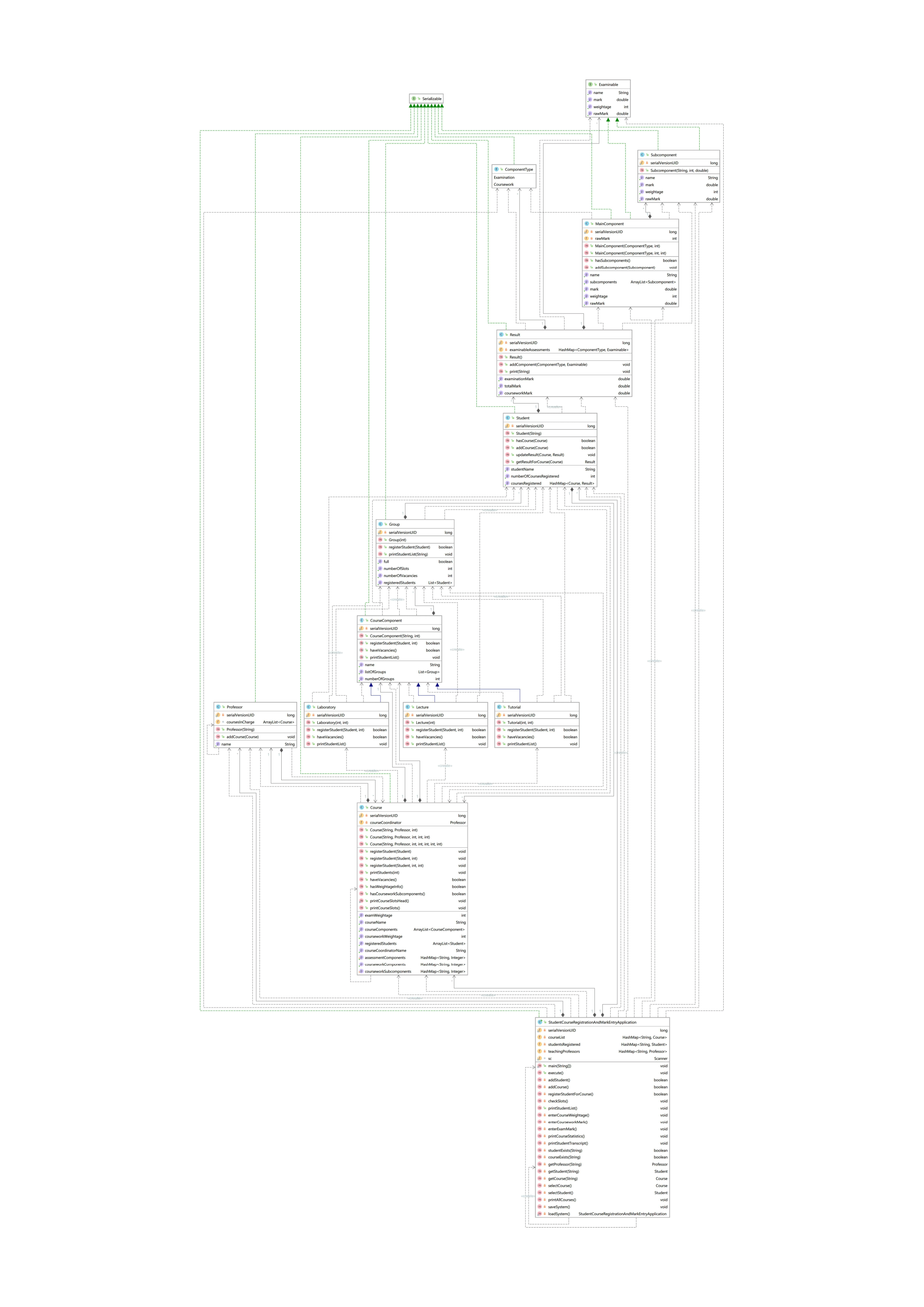
With the Single Responsibility Principle (SRP) in mind, our group has defined our classes to include attributes and methods that are directly related to the classes themselves. In doing so, we ensure that each class only assumes a single responsibility and there will not be more than one reason for any class to change. Furthermore, we noticed that some of our classes contribute to realizing the same underlying requirement. For example, both lecture and tutorial classes deal with course components. Consequently, we have opted to group such classes under the same package for organization and communication purposes.

Except for the *SCRAME* class, every class deals with changes pertaining to their own class and can therefore be represented as ‘entities’. The *SCRAME* class however, acts as a ‘control’ as it is responsible for calling the methods of other classes to coordinate and realize use cases. It also acts as the ‘boundary’ since it is through this class that the user interacts with the system.

At the highest level of abstraction, all classes implement the ‘serializable’ interface. After serialization, the entire state of the application is saved into a binary file and can be deserialized during the next execution of the program. The purpose of this design is to avoid having to enter past data into the application every time it is restarted. By implementing ‘serializable’ interface, all the classes then become serializable can therefore have their internal states/attributes stored as a binary file.

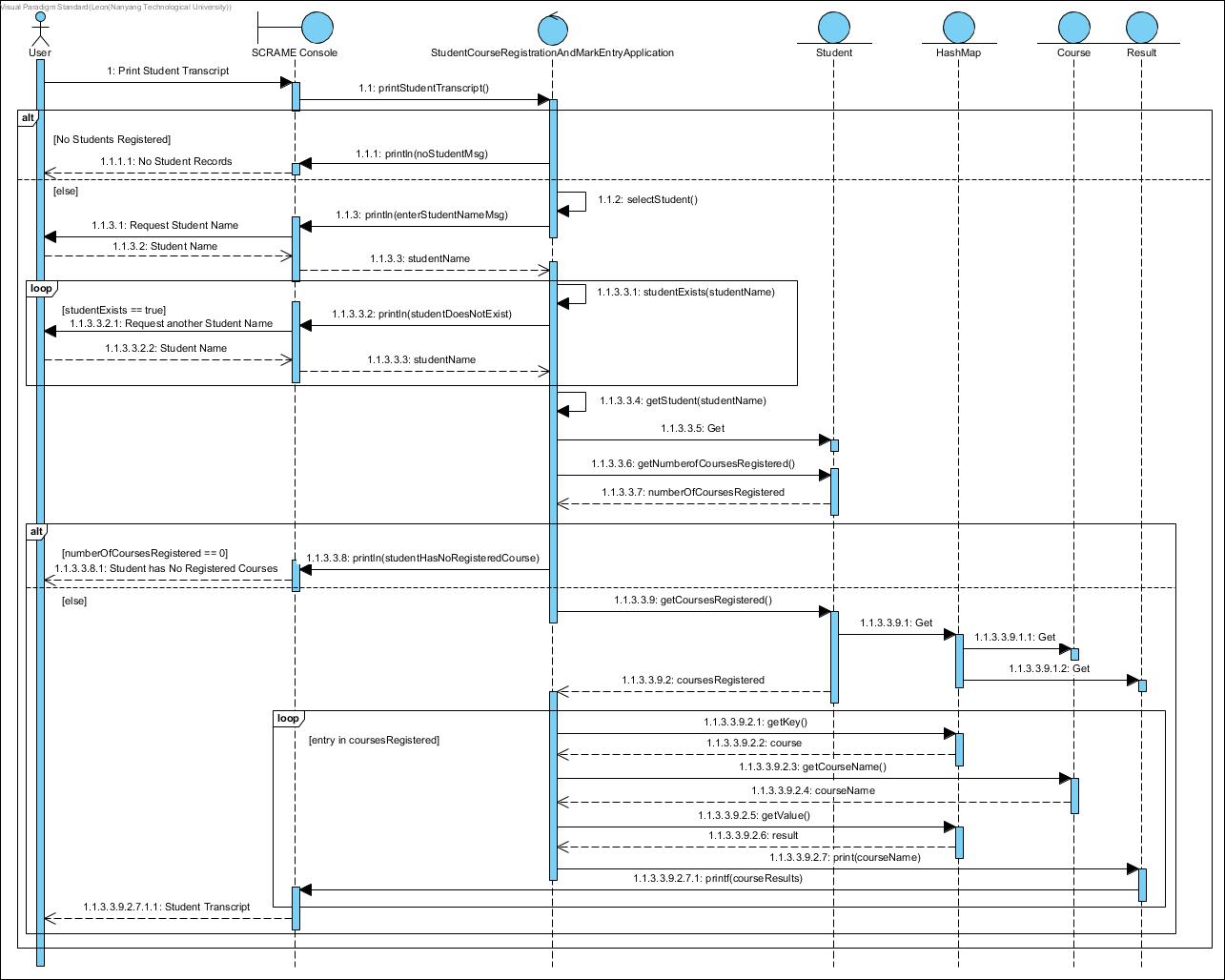
## 

## 4. UML Class Diagram



## 

## 5. UML Sequence Diagram of “Print Student Transcript



## 6. Required Functionalities

## 7. Test Cases and Results

