

**NANYANG**  
**TECHNOLOGICAL**  
**UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**NANYANG TECHNOLOGICAL UNIVERSITY**

**CZ2002 Project Report, Lab Group SS9**

**LAB TA: NGUYEN XUAN PHI**

<b>Introduction</b>	<b>3</b>
<b>Design Considerations</b>	<b>4</b>
Entity-Control-Boundary	4
<b>Principles, OO Concepts</b>	<b>4</b>
Open-Closed Principle (OCP), Reusability and Extensibility	4
Single Responsibility Principle (SRP) & Maintainability	5
Loose Coupling	5
Liskov Substitution Principle (LSP)	6
Interface Segregation Principle (ISP)	6
Encapsulation	6
<b>Assumptions made</b>	<b>7</b>
<b>UML Class diagram</b>	<b>7</b>
<b>UML Sequence Diagram</b>	<b>8</b>
<b>Test Cases</b>	<b>8</b>
<b>Declaration of Original Work</b>	

## **Introduction**

The objective of this assignment is to model, design and develop an Object-Oriented (OO) console-based application whilst applying relevant OO concepts taught in the course and increase our familiarity with using Java as an object oriented programming language.

The My STudent Automated Registration System (MySTARS) is a university application for each School's academic staff and undergraduate students. This application facilitates the creation of courses, adding of students into the system and registration of courses and students.

## **Design Considerations**

### **Approach Taken**

#### Entity-Control-Boundary

The application adopts a entity-control-boundary stereotype classes framework which splits the application into three different logical components, entity, control and boundary. Users would only interact with the boundary. The boundary would communicate with the controller, and the controller would interact with the entities. The boundary handles inputs and actions from the user. The controller handles the logic and execution of different use-cases of the application. The entity represents system data, usually data that is persistent throughout the application.

The boundaries in the application include *MySTARSAApp*, *AdminUI*, *StudentUI*. The controllers in the application include *StudentController*, *CourseController*, *AccountController*, *NotificationController* and *FileManager*. The entities in the application include *Admin*, *Course*, *Index*, *Lesson*, *Student*, *StudentRegisteredCourses* and *User*.

### **Principles, OO Concepts**

#### Open-Closed Principle (OCP), Reusability and Extensibility

“A module should be open for extension but closed for modification”. This principle ties in closely with the OO concept inheritance and abstraction.

The User class contains common logic and attributes that are not only reused by Admin and Student Classes, but also leaves room for flexibility for Admin and Student classes to accommodate future extensions without modifying the parent User class. User class can also be reused when new user-types, such as ‘Professor’ or ‘IT Support’ are implemented in the future.

Another example of using OCP, would be the NotificationController. The EmailNotificationController inherits from NotificationController and implements the abstract method. The parent class, NotificationController allows easy extension, any new notification methods such as SMS could be easily implemented without modifying any code in it.

### Single Responsibility Principle (SRP) & Maintainability

The Single Responsibility Principle states that every class, function or module in an application should only have one responsibility.

In the MySTARS application, entities such as Student, Index, Course are only responsible for their relevant attributes. For example, the Student entity class is only responsible for storing each student's personal details and returning those details, while the Course class is only responsible for storing and returning Course details and the index within the course. This principle has been kept in mind while designing all entities, to ensure that a class "should have one, and only one, reason to change".

Another aspect of the application that follows the SRP closely are the controllers. Each controller is created strictly for one purpose and one purpose only. For example, the StudentController is responsible for managing and manipulating all Student entities and relevant data pertaining to the Student entity. This includes student verification and retrieval of students that are registered to a certain course or index. Similarly, CourseController only manages data and entity manipulation with regard to the Course and Index entities.

Since "There should never be more than ONE reason for a class to change", we are able to achieve maintainability while enforcing SRP. When there is an error to debug or a feature to improve, we are able to quickly identify the module in question and implement changes as every class/module only has one responsibility. In addition, keeping our code short and readable improves maintainability as well.

### Loose Coupling

An effect of SRP is also loose coupling. A system is said to be loosely coupled if the elements of the system have minimal knowledge of internal details of other elements, and are also able to carry out its intended function with minimal or no such knowledge. For example, in the Course class, the Course is not tagged to any specific student since there is no need for the Course to

know about students taking the course. Instead, the StudentRegisteredCourses class will record such information. This way, coupling is kept as low as possible.

### Liskov Substitution Principle (LSP)

The Liskov Substitution Principle is present in our Admin and Student classes. The LSP defines that objects of a base class shall be substituted with objects of its subclasses without noticeable difference. For example, hashPassword() is present in all User, Admin and Student classes where the pre-conditions and post-conditions are no stronger and no weaker than the base class method respectively. A user of the User class will be able to continue to function as intended if Admin or Student is passed to it as all three have the same log-in process. Hence they are substitutable for the User class and demonstrate LSP.

### Interface Segregation Principle (ISP)

“Clients should not be forced to depend upon interfaces that they do not use.” Likewise, the modules in MySTARS do not unnecessarily inherit classes that are not relevant to their responsibilities. This helps to keep the maintenance and updating of the application in the future simple and also prevent bloated classes that define methods for various responsibilities.

### Encapsulation

Encapsulation is used to keep values and inner workings of an object hidden within the class and prevent illegal direct access by other classes. In our application, entity attributes are private while the getter and setter methods are public. This allows any user of the class to be able to get or set an attribute, without knowing its internal workings. For example, for a Student entity, to obtain a student’s matric number, we would have to use getMatricNumber() to retrieve it.

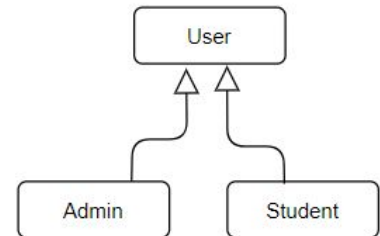
### Composition

In our application, there is a composition relationship between Course and Index as well as Index and Lesson. There is a whole-part relationship between Course and Index, Index and Lesson. An index cannot exist without a course and a lesson cannot exist without an index. Thus, if an index is updated, all the lessons under the index would be updated as well. If a course is updated, all the indexes under the course would be updated.



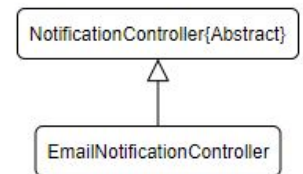
### Inheritance

In our application, Student and Admin classes are derived from the User class. Both children classes inherit and use the attributes username and password from the User class, and getter and setter methods as well as the HashPassword() method.



### Abstraction

In our application, EmailNotificationController inherits from NotificationController, which is an abstract class. It does not have implementation code, while EmailNotificationController will implement the abstract method, sendNotification(). This allows for any notification methods implementing NotificationController to implement their own code to send notification.

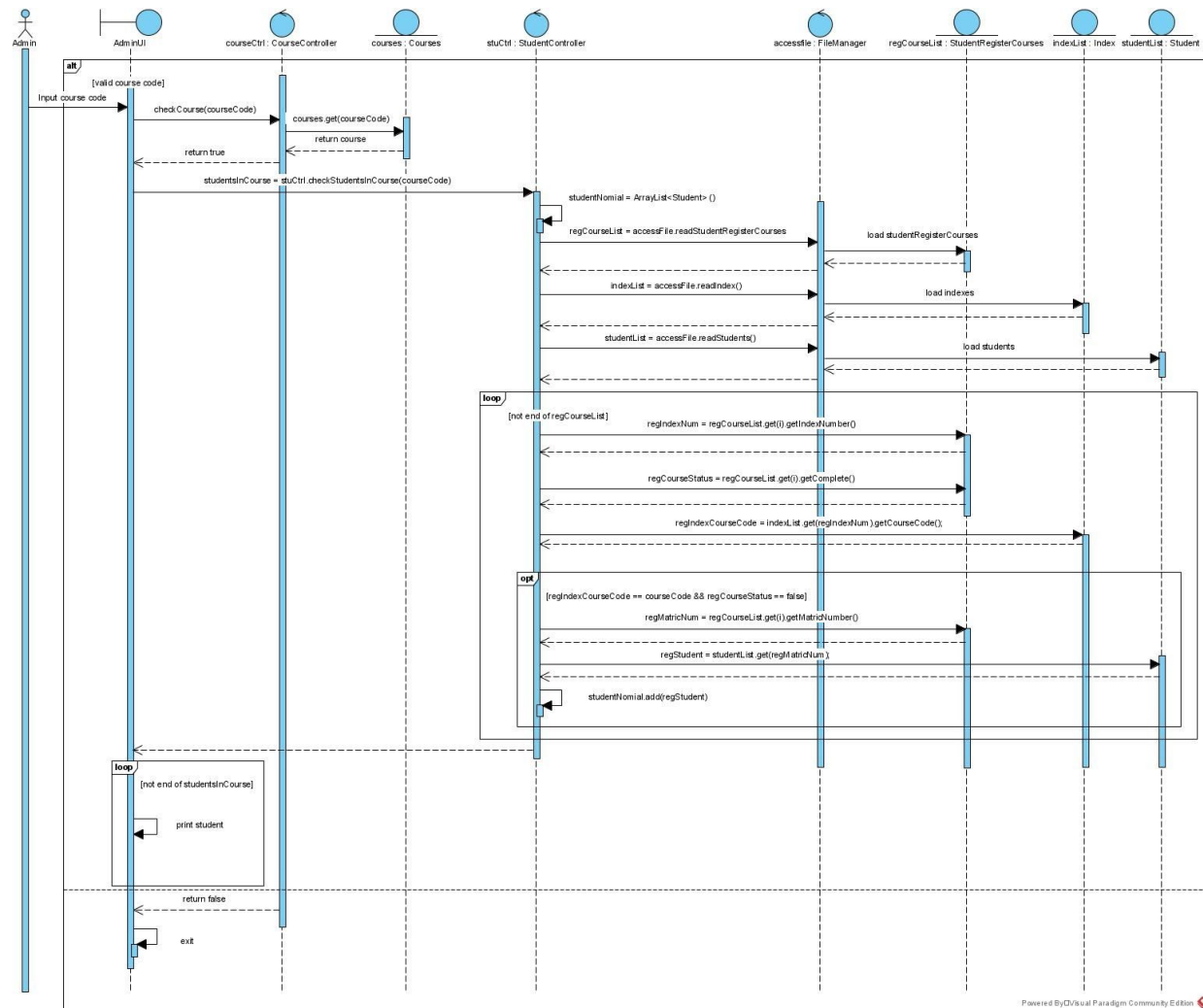


### Assumptions made

1. We assume that all lessons (lectures, seminars, labs etc) will be held weekly, i.e. every week.
2. We do not consider pre-requisite conditions when registering for a course.

### UML Class diagram

## UML Sequence Diagram



## Test Cases

### 1. Student Login

Login before allowed period	<pre> Enter Domain (Admin or Student): student Enter username: alch01 Enter password: Please log in during your registration access period Your registration access period: 20-Nov-2021 10:30 to 30-Nov-2021 10:30                 </pre>
-----------------------------	---



Login after allowed period	<pre> Enter Domain (Admin or Student): student Enter username: jonlee1 Enter password: -----Student Menu----- 1. Add Course 2. Drop Course 3. Check/Print Courses Registered 4. Check Vacancies Available 5. Change Index Number of Course 6. Swap Index Number With Another Student 7. Logout </pre>
Wrong password	<pre> Enter Domain (Admin or Student): student Enter username: alch01 Enter password: Login Failed - Wrong password </pre>

## 2. Add a student

Add a new student	<pre> Enter student's matric number: U1921999Y Enter student's name: David Lee Enter student's username: davlee1 Enter student's password: davlee1 Enter student's email: davlee1@gmail.com Enter student's gender: (Male/Female) Male Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-25 10:30 Student successfully added!  List of all Students: Matric Number   Username   Name   Email   Gender   Nationality ===== U1922331U   albch01   Albert Ching   albch01@gmail.com   Male   Singaporean U19219297B   bgyu   Bagel Yu   grp9stud@gmail.com   Male   Singaporean U1988771E   slam01   Susan Lam   grp9stud@gmail.com   Female   Singaporean U1823454C   jonlee1   Jonas Lee   jonlee1@gmail.com   Male   Singaporean U1899229E   ivtan2   Ivan Tan   ivtan2@gmail.com   Male   Singaporean U1920117A   alee1   Alex Lee   grp9stud@gmail.com   Male   Singaporean U1878330R   haich01   Hailey Chua   haich01@gmail.com   Female   Singaporean U1856315S   justch1   Justin Chong   justch1@gmail.com   Male   Singaporean U1988118T   kelch4   Kelvin Chan   kelch4@gmail.com   Male   Singaporean U1921999Y   davlee1   David Lee   davlee1@gmail.com   Male   Singaporean U1803447K   kygli   Kygo Lim   kygli@gmail.com   Male   Singaporean U1989829M   rox01   Roxanna Jing   grp9stud@gmail.com   Female   Singaporean U1900152G   selng1   Selena Ng   selng1@gmail.com   Female   Singaporean U1919191A   aimann01   Aimee Ann   grp9stud@gmail.com   Female   Singaporean U19223370   jtan01   John Tan   grp9stud@gmail.com   Male   Singaporean U1944728P   mikch4   Mike Chan   mikch4@gmail.com   Male   Singaporean </pre>
-------------------	---

Add an existing student	Enter student's matric number: U1922331U Enter student's name: Albert Ching Enter student's username: alch01 Enter student's password: alch01 Enter student's email: alch01@gmail.com Enter student's gender: (Male/Female) Male Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-30 10:30 Student already exists! Please enter fields again!
Edit student access period	Enter student's matric number: U1922331U Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-21 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-25 11:30
<u>Invalid data entries:</u>	
Matric number already exists	Enter student's matric number: U1922331U Enter student's name: Michelle Cheng Enter student's username: micheng1 Enter student's password: micheng1 Enter student's email: micheng1@gmail.com Enter student's gender: (Male/Female) Female Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-30 10:30 Matric number already exists! Please enter fields again!
Username already exists	Enter student's matric number: U1912367A Enter student's name: Michelle Cheng Enter student's username: alch01 Enter student's password: alch01 Enter student's email: alch01@gmail.com Enter student's gender: (Male/Female) Female Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Username already exists! Please enter fields again!

Email already exists	Enter student's matric number: U1912367A Enter student's name: Michelle Cheng Enter student's username: micheng1 Enter student's password: micheng1 Enter student's email: alch01@gmail.com Enter student's gender: (Male/Female) Female Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-30 10:30 Email already exists! Please enter fields again!
Gender not entered correctly	Enter student's matric number: U1912367A Enter student's name: Michelle Cheng Enter student's username: micheng1 Enter student's password: micheng1 Enter student's email: micheng1@gmail.com Enter student's gender: (Male/Female) f Enter student's nationality: Singaporean Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-11-30 10:30 Please enter correct format for gender(Male/Female)! Please enter fields again!
Access date time not entered correctly	Enter start date time for access(YYYY-MM-DD HH:MM): 2020-20-11 10:#0 Please enter the valid date time format! Enter start date time for access(YYYY-MM-DD HH:MM): 2020-11-20 10:30 Enter end date time for access(YYYY-MM-DD HH:MM): 2020-30-11 10:30 Please enter the valid date time format! Enter end date time for access(YYYY-MM-DD HH:MM):

### 3. Add a course

Add a new course	Enter new Course Code: CZ3001 Enter the Course name: Advance Computer Architecture Enter Course School: SCSE Enter number of AUs: 3 New Course successfully added! List of all Courses: Course Code   Name   School   AU): CZ3001   Advance Computer Architecture   SCSE   3 CZ2001   Algorithm   SCSE   3 CZ2002   Object Oriented   SCSE   3 CZ1002   Basic Programming   SCSE   3 CZ2004   Human Computer Interaction   SCSE   3 CZ3005   Computer Visualization   SCSE   3 CZ2006   Software Engineering   SCSE   3 CZ2007   Database Management   SCSE   3
Add a new index to course	Enter the index you wish to add: 12320 Enter the Vacancy: 25 New Index successfully added!

Add a new lesson to index	Enter index number: 12320 Enter lesson type: Lecture Enter lesson day: Thursday Enter lesson venue: LT2A Enter lesson start time(HH:MM): 11:00 Enter lesson end time(HH:MM): 12:00 Lesson successfully added!
<u>Invalid data entries:</u>	
Add an existing course	Enter new Course Code: CZ2004 This Course Code already exists
Add lesson to non-existing index	Enter index number: 11111 The index doesn't belong to the course. Please re-enter!

#### 4. Register student for a course

Add a student to a course index with available vacancies	Enter the course CZ2007 Enter index 19802 Successfully registered!
Add a student to a course index with 0 vacancies in Tut/Lab	Enter the course CZ2004 Enter index 13002 You are added to waitlist!
Register the same course again	Enter the course CZ2007 Enter index 19801 You have already registered for this course!
Change index of registered course	Enter the course CZ2002 Enter the original index 18812 List of indexes and vacancies: Index                      Vacancy ----- 18811                      0 18812                      20 18813                      21  Enter new index 18813 Successfully changed index!

Swop index with peer	Enter peer's username ivtan2 Enter peer's password ivtan2 Enter your index 18813 Enter peer's index 18812
Print registered courses	Susan Lam registered courses. ===== Course: CZ2007    Title: Database Management    AU: 3    Index: 19802    School: SCSE Lecture Day: Tuesday Time: 12:30-14:30 Venue: LT2A Registered Tutorial Day: Thursday Time: 14:30-16:30 Venue: TR+16 Registered Lecture Day: Tuesday Time: 12:30-14:30 Venue: LT2A Registered ===== Course: CZ2002    Title: Object Oriented    AU: 3    Index: 18812    School: SCSE Tutorial Day: Tuesday Time: 10:30-12:30 Venue: TR+14 Registered Lecture Day: Tuesday Time: 08:30-10:30 Venue: LT2A Registered =====
<u>Invalid data entries:</u>	
Index does not belong to course	Enter the course CZ2002 Enter index 19801 Index does not belong under the course. Please re-enter
Course code does not exist	Enter the course CZ2237 Enter index 18811 Invalid course code: Please re-enter!
Course code and index does not exist	Enter the course CZ2237 Enter index 11111 Invalid index and course code: Please re-enter!

## 5. Check available slot in a class(vacancy in a class)


Check for vacancy in course index	Enter the course code CZ2007 Course Code: CZ2007 Index Vacancy ----- 19801 29/32 19802 19/20
<u>Invalid data entries:</u>	
Course code does not exist	Enter the course code CZ2003 Invalid course code: Please re-enter.
Course code has no indexes	Enter the course code CZ2006 Currently, this course code have no indexes.

## 6. Day/Time clash with other course



Add a student to a course index with available vacancies, but clash with timetable	Enter the course CZ2002 Enter index 18811 Chosen index clashes with current timetable. Please choose another index								
Change index of course, but clash with own timetable (Additional cases)	Enter the course CZ2002 Enter the original index 18811 List of indexes and vacancies: <table> <thead> <tr> <th>Index</th><th>Vacancy</th></tr> </thead> <tbody> <tr><td>18811</td><td>0</td></tr> <tr><td>18812</td><td>21</td></tr> <tr><td>18813</td><td>21</td></tr> </tbody> </table> Enter new index 18813 Chosen index clashes with current timetable, unable to change index!	Index	Vacancy	18811	0	18812	21	18813	21
Index	Vacancy								
18811	0								
18812	21								
18813	21								
Swop index with peer, but clash with own timetable (Additional cases)	Enter peer's username mikch4 Enter peer's password mikch4 Enter your index 18812 Enter peer's index 18813 The new index clashes with current timetable, unable to swap index!								
Swop index with peer, but clash with peer's timetable (Additional cases)	Enter peer's username bgyu Enter peer's password bgyu Enter your index 18813 Enter peer's index 18812 The new index clashes with peer's timetable, unable to swap index!								

## 7. Waitlist Notification

Add studentA to a course with 0 vacancies	Enter the course CZ2004 Enter index 13002 You are added to waitlist!
Drop studentB from the same course index	<b>Student B (jonlee1)</b> Enter the course CZ2004 Enter index 13002 You have successfully de-register from this course! <b>Student A(ivtan2)'s email</b>  ss9cz2002grp9@gmail.com to me ▾ Dear Ivan Tan  You have successfully registered CZ2004
Display studentA's timetable	Ivan Tan registered courses. ===== Course: CZ2004    Title: Human Computer Interaction    AU: 3    Index: 13002    School: SCSE Lecture Day: Wednesday Time: 08:30-10:30 Venue: LT2A Registered Tutorial Day: Monday Time: 08:30-10:30 Venue: TR+18 Registered =====

## 8. Print student list by index number, course

Print student list by Course	Enter Course code: <b>CZ2002</b> Student List for Course Code CZ2002 (Name, Gender, Nationality) 1) Bagel Yu, Male, Singaporean 2) Susan Lam, Female, Singaporean 3) Alex Lee, Male, Singaporean 4) Hailey Chua, Female, Singaporean 5) Kygo Lim, Male, Singaporean
Print student list by Index	Enter Index number: <b>18811</b> Student List for Index 18811 (Name, Gender, Nationality) 1) Bagel Yu, Male, Singaporean 2) Alex Lee, Male, Singaporean 3) Hailey Chua, Female, Singaporean 4) Kygo Lim, Male, Singaporean
<u>Invalid data entries:</u>	
Course code does not exist	Enter Course code: <b>CZ2003</b> The course does not exist.
Index does not exist	Enter Index number: <b>18814</b> The Index does not exist.
Course has no student	Enter Course code: <b>CZ3001</b> Student List for Course Code CZ3001 (Name, Gender, Nationality) There no student under this course.
Index has no student	Enter Index number: <b>12320</b> Student List for Index 12320 (Name, Gender, Nationality) There no student under this index.

### Declaration of Original Work for CE/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 (CE2002 or CZ2002)	Lab Group	Signature /Date
Tay Zhixuan	CZ2002	SS9	

Ray Myat Theingar Cho	CZ2002	SS9	
Leow Guan Wei	CZ2002	SS9	
Benedict Leong Wei Xin	CZ2002	SS9	