Team Information

Team ID	Group 15
Team Repo on GitHub	https://github.com/a1347539/comp3111Project

Name (Member 1)	Li Chun Tak
GitHub ID	a1347539
Email ID	1347539@gmail.com
Dev Branch	dev/input
Task Assignment	INPUT

Name (Member 2)	HE Qihao
GitHub ID	phyqh
Email ID	qheag@connect.ust.hk
Dev Branch	dev/process
Task Assignment	PROCESS

Name (Member 3)	SZE Wing Kwan
GitHub ID	szeu
Email ID	wkszeaa@connect.ust.hk
Dev Branch ID	dev/output
Task Assignment	OUTPUT

Submitted by: _____Li Chun Tak_____

Date of Submission: _____16/10/2022_____

Class Diagram (for overall system)

Student

Student_Name : String
Student_ID : String
Email : String
K1_energy : int
K2_energy : int
K3_Tick1 : Boolean
K3_Tick2 : Boolean
My_Preference : Boolean
Concerns: String

3..4   AssignTo   1

Team

Team_ID : int
Member_Count : int
RecommendedLeader : Student

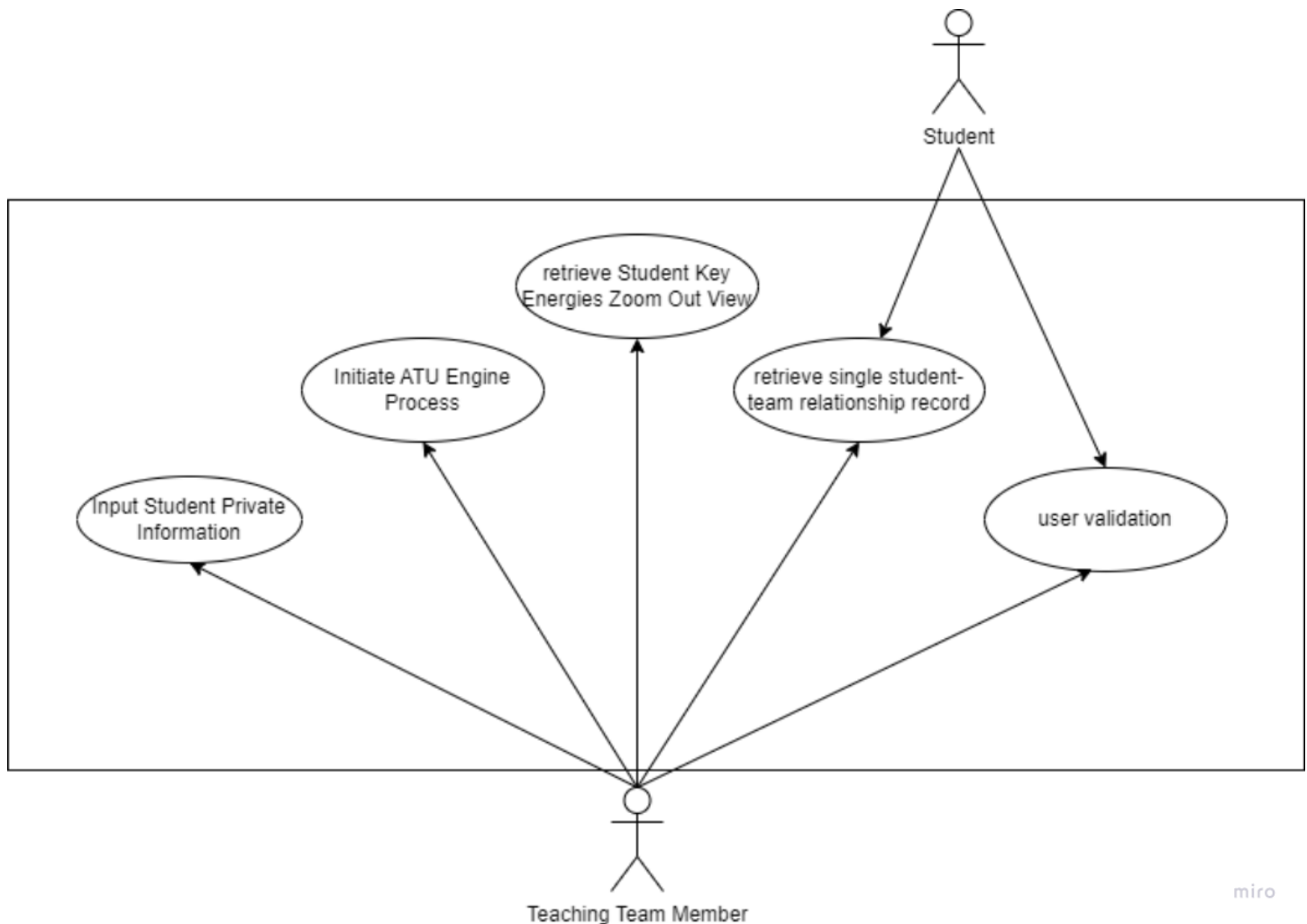miro
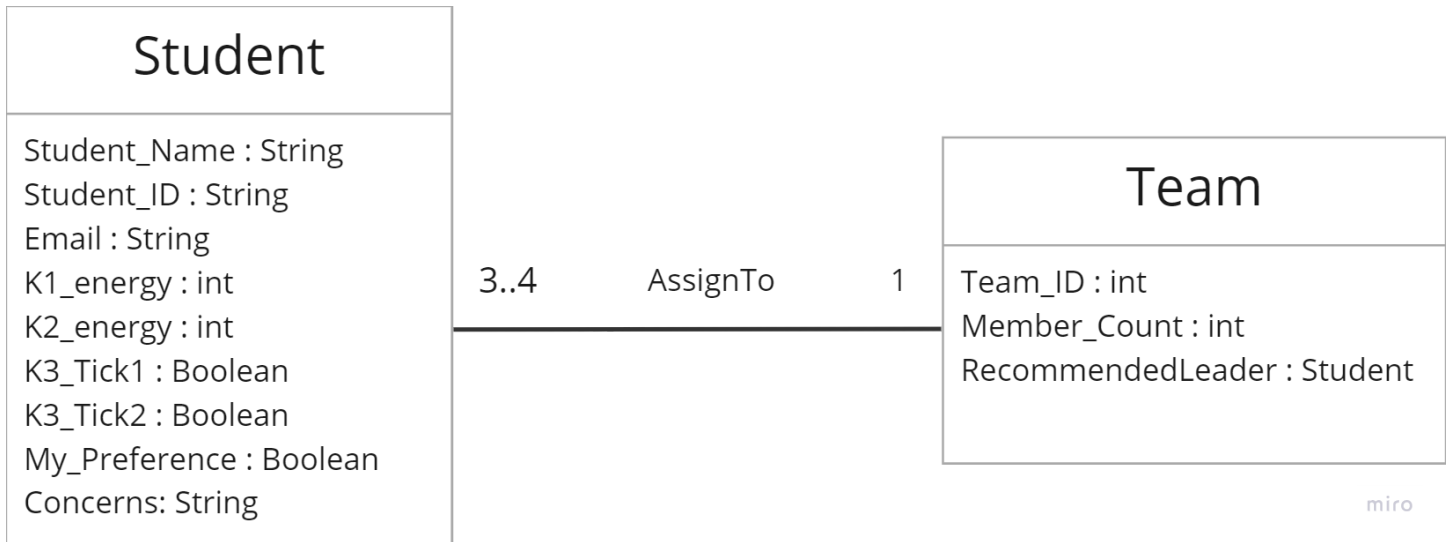
A student is assigned to 1 and only 1 team.

A team can have 3 to 4 students.

Use Case Diagram (for overall system)

There will be two actors, namely Teaching Team Member and Student, interacting with the system.

The Teaching Team Member can initiate 5 use cases while a Student can initiate only 2.

Use Case Specification: Task (<mark>INPUT or PROCESS or OUTPUT</mark>)

Administration

Administrative Use Case Specification 1

Use case	User validation
Brief description	This use case identifies the user's privilege, and decide rather the system should display the buttons for performing extra actions.
Participating actors	TeachingTeamMember, Student
Preconditions	The system has a pre-defined passcode.
Flow of events	Basic flow: 1.The use case begins when the user get onto the system. 2.The system asks for a passcode. {Enter passcode} 3. The system listens to the user input. 4 The system processes the passcode. {Process passcode} 5. The use case ends. Alternative flow: A1. Empty passcode At {Process passcode} if the user input is empty, 1. The system recognizes the user as a non-Teaching Team Member and hides the buttons for performing the extra actions which include read-chart, upload csv. file, and run algorithm. 2. The use case ends. A2. Corredt passcode At {Process passcode} if the user input is equal to the predefined passcode, 1. The system recognizes the user as a Teaching Team Member and shows the buttons for performing the extra actions which include read-chart, upload csv. file, and run algorithm. 2. The use case ends. A3. Incorredt passcode At {Process passcode} if the user input is not equal to the predefined passcode, 1. The system creates a pop-up window to inform the user that the passcode is incorrect. 2. The flow of events is resumed at {Enter passcode}.
Postcondition	if entered the passcode correctly, the system shows the buttons for performing the extra actions, else, the system hides the buttons for performing the extra actions.

INPUT

INPUT Use Case Specification 1

Use case	Upload student information.
Brief description	The system receives the csv. file containing all the student's information. Then it structures the data.
Participating actors	Teaching Team Member
Preconditions	The system shows the corresponding button for this action. The data in csv. has all the student information. The input data is in the correct format.
Flow of events	Basic flow: 1. The use case begin when the UPLOAD button is pressed. 2. The system retrieve the user entered csv. file's absolution path+file name from the corresponding text field. {Find csv. file in user specified directory} 3. The system loads the csv. file from the given path. 4. The system receives the student information. 5. The system represents each student information with a self-defined data structure and put all the records in a container. 6. The system responses with a pop-up window showing all the students information stored in the system, count of the number of student and K3_ticks, and min/max/mean of the overall K1 and K2 Energy. 7. The use case ends. Alternative flow: A1. Incorrect directory At {Find csv. file in user specified directory} if there does not exist a csv. file with the inputed file name. 1. The system displays error message to tell the user type again. 2. The use case ends.
Postconditions	The correct csv. file is uploaded and each of the student information is correctly stored in an internal data structure.

miro

PROCESS

Process Use Case Specification 1

Use case	Initiate the processing of ATU Engine
Brief description	This use case describes the system interaction when the Teaching Team Member start the processing of the ATU Engine
Participating actors	Teaching Team Member
Preconditions	The system shows the corresponding button for this action. The system has accurate Student Information prepared and ready to be retrieved by the ATU Engine.
Flow of Events	Basic flows: 1. The use case begins when the START button for starting the ATU Engine algorithm is pressed. 2. Peform subflow *ATU Engine runs.* 3. The System retrieves the team-up records from the dataset with records of team id related to student id. 4. The system creates a pop-up window to display the records, e.g., number of team formed. 5. The use case ends. Sub flows: S1: ATU Engine runs {Begin Processing} 1. The ATU Engine retrieves and processes the Student Information to form teams. {Store team-up result} 2. The ATU Engine stores team-up results in a dataset with records of team id related to student id. 3. The system recommends a team leader for each of the teams based on the K1_energy of the teammates Alternative flows: A1: Invalid Student Information At {Begin Processing}, if stored student information is invalid, 1. The system creates a pop-up window to inform the user that student data is invalid. 2. The use case ends. A2: Invalid team-up result At {Store team-up result}, if no or invalid team-up results are present in the generated results by the ATU Engine, 1. The system informs the user the failure of the team-up process. 2. The use case ends.
Postcondition	All students have been assigned a team, where the team information is stored in a new dataset with record of Team and Student relationship. Each Team will have at least 3 at most 4 students with one of them being the recommended leader. Each Team will have at least one student whose K1_energy >= average_k1_energy. K1_Energy & K2_Energy for each team are balanced.

OUTPUT

OUTPUT Use Case Specification 1

Use case	Check Teaming Up Result.
Brief description	This use case describes the system interaction when a user checks the teaming up result using a student ID or student name.
Participating actors	Teaching Team Member, Student
Preconditions	The system has finished processing all information and generated teaming up result.
Flow of events	Basic flow: 1. The use case begins when the SEARCH button is pressed. 2. The system retrieves the student name or student id from the corresponding text field that the user entered. {Input validation} 3. The system checks the validity of the input. 4. The system creates a pop-up window to display the corresponding team information. 5. The use case ends. Alternative flow: A1. Invalid Input At {Input validation} if the user input is invalid, meaning that the given student ID or the student name does not exist in the system. 1. The system creates a pop-up window to display an error message telling the user to type again. 2. The use case ends.
Postconditions	The resulting team contains the student with the given ID or name.

miro

OUTPUT Use Case Specification 2

Use case	Check Student Key Energies Chart.
Brief description	This use case describes the system interaction when the Teaching Team Member checks student key energies displayed on a 2D-Line Chart.
Participating actors	Teaching Team Member
Preconditions	The system shows the corresponding button for this action. The system has all students' records (with valid energies) stored.
Flow of events	Basic flow: 1. The use case begins when the Check Chart Result button is pressed. 2. The system retrieves each student's record and get their K1 and K2 energy. 3. The system generates the Student Key Energies Zoom Out View Chart. 4. The system creates a pop-up window to display the chart. 5. The use case ends.
Postconditions	NA

Task Allocation:

Name:	Task ID:	Task Description:
Li Chun Tak	113	Set up Team Repo on GitHub
Li Chun Tak	121	Draw a class diagram
HE QiHao	122	Draw the use case diagram
Li Chun Tak	123-1	Use case specification - Input
HE QiHao	123-2	Use case specification - Process
SZE Wing Kwan	123-3	Use case specification - Output
SZE Wing Kwan	211-1	Meeting Minutes - 1
Li Chun Tak	211-2	Meeting Minutes - 2
HE QiHao	211-3	Meeting Minutes - 3
Li Chun Tak	212	Draw Gantt chart
Li Chun Tak	213	Draw Burndown chart
Li Chun Tak	214-1	Write at least one meaningful Git commit log - 1
HE QiHao	214-2	Write at least one meaningful Git commit log - 2
SZE Wing Kwan	214-3	Write at least one meaningful Git commit log - 3
HE QiHao	221	Unit testing report

HE QiHao	222	Coverage report
SZE Wing Kwan	223	Documentation with JavaDoc
Li Chun Tak	231A	Task INPUT
HE QiHao	231B	Task PROCESS
SZE Wing Kwan	231C	Task OUTPUT
HE QiHao	233-1	Perform unit testing
SZE Wing Kwan	233-2	Integrate all tasks
Li Chun Tak	233-3	Perform integration testing