**INTI International College Penang School of Engineering and Technology**

**3+0 Bachelor of Science (Hons) in Computer Science, in collaboration with Coventry University, UK**

**3+0 Bachelor of Science (Hons) in Computing, in collaboration with Coventry University, UK**

**Coursework cover sheet**

**Section A - To be completed by the student**

|  |  |
| --- | --- |
| Full Name: Chan Kok Han | |
| CU Student ID Number: P21013717 | |
| Semester: 3 | |
| Session:  **April 2022** | |
| Lecturer:  **Nadhrah Abdul Hadi (nadhrah.abdulhadi@newinti.edu.my)** | |
| Module Code and Title:  **4067CEM Software Design** | |
| Assignment No. / Title:  **Continuous Assessment** | % of Module Mark:  **50** |
| Hand out Date:  **22nd April 2022** | Due Date:  **Task 1: 13 May 2022, by 11.59pm**  **Task 2: 1 July 2022, by 11.59pm**  **Task 3: 17 June 2022, by 11.59pm.**  **Task 4: 17 June 2022, by 11.59pm.**  **Task 5: 17 June 2022, by 11.59pm.** |
| Penalties: No late work will be accepted. If you are unable to submit coursework on time due to extenuating circumstances, you may be eligible for an extension. Please consult the lecturer. | |
| Declaration: I/we the undersigned confirm that I/we have read and agree to abide by the University regulations on plagiarism and cheating and Faculty coursework policies and procedures. I/we confirm that this piece of work is my/our own. I/we consent to appropriate storage of our work for plagiarism checking.  Signature(s): Han | |

**Section B - To be completed by the module leader**

|  |  |  |
| --- | --- | --- |
| Intended learning outcomes assessed by this work:  1. Understand and apply appropriate concepts, tools and techniques to each stage of the software development  2. Understand and apply design patterns to software components in developing new software  3. Demonstrate an understanding of project planning and working to agreed deadlines, along with professional, interpersonal skills and effective communication required for software production  5. Demonstrate an awareness of, and ability to apply, social, professional, legal and ethical standards as documented in relevant laws and professional codes of conduct such as that of the Malaysian National Computer Confederation. | | |
| Marking scheme | Max | Mark |
| 1. User Story Mapping 2. Setting up a GitHub Repository 3. Creating a Class diagram and design pattern selection 4. Creating a Prototype User Interface and Usability Testing 5. Discuss the ethical issue related to the software | 20  10  30  20  20 |  |
| Total | 100 |  |

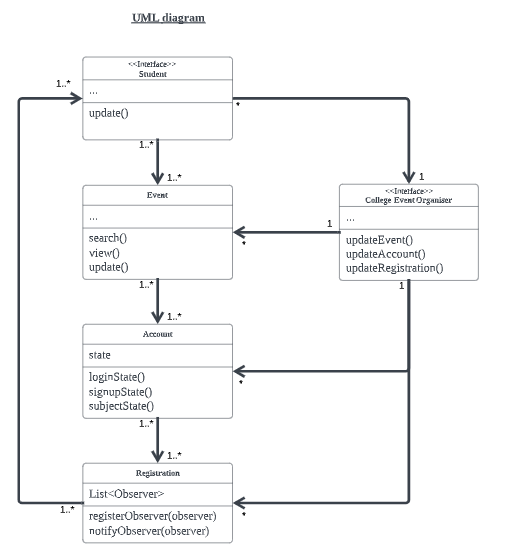
# Class Diagram

# The College Events System for Students consists of five classes, and each responsibility is as follows: -

1. College Event Organiser class manages all the operations of the Event, Account, and Registration of a College Event Organiser/Admin.
2. Student class manages all the operations of Event, Account, and Registration of Student.
3. Account class manages all the operations of user accounts.
4. Registration class manages all the operations of event registrations.
5. Event class manages all the operations of an event.

# As per the class diagram above, College Event Organiser and Students can search/view events without logging in or creating an account. College Event Organiser must log into the system to manage Event, Account, and Registration. Students can sign up for the events by registering and logging in and will be notified when registering for the event.

**UML Diagram**



**Problem**

How do students join the event?

There are several events every time for all the students to join and register. Students can check daily whether an event is available for registration or be notified by someone when an event is happening. This registration is meaningless for students signing up to participate in this event, as they generally do not know their eligibility. Students waste time or fail to check registration for events.

Ideally, students will be notified of the event after successful registration and are satisfied with their event registration.

Therefore, considering the above problem, the Observer design pattern is suitable to implement in the College Events System for Students and is a behavioural pattern that defines a one-to-many relationship that notifies other objects when any state changes and is automatically updated. It also wishes to be informed about events happening in the system.

As shown in the UML (Unified Modeling Language) diagram above, which represents a class diagram as a design pattern UML, the Observer pattern consists of various components and associations that include all the observer, concrete subject, subject, and concrete observer classes used to represent the problem and the justification as follows: -

**Observer**

The Student class is an Observer. It needs to be informed about changes and register with the subject to receive notifications from the Registration class. It is an updated interface that notifies the student of any changes, and students receive notifications after registering for the event when the subject provides its data.

**Concrete Subject**

The Event and Account classes are Concrete Subjects and store state of interest and send a notification to the Registration class when a state changes and have the ability for students to search, view, and register for the events. The student who wants to register for the event will sign up or log in to their account and select the event before the registration. The subject state will notify the Registration class when the Registration class state changes inconsistent with its class state.

**Subject**

The Registration class is a Subject and contains a list of registered observers and maintains student lists. The data the student observer wants to know belongs to this Subject class, and multiple students can observe a Registration. It provides an interface for registering and posting notifications about changes in the Subject state. The post notifications do not need to be changed if we have a new student sign up for an account, and the list is automatically updated whenever a new student registers for the event.

**Concrete Observer**

The College Event Organiser class is a Concrete Observer and upholds reference to Event, Account, and Registration classes and implements update functions. It also contains information specific to the current instance and implements the interface for responding to change using information about the change obtained from the concrete subjects.