**University of the West Indies, Mona Campus**

**FST, COMP3161 (Mr. Matthew Stone)**

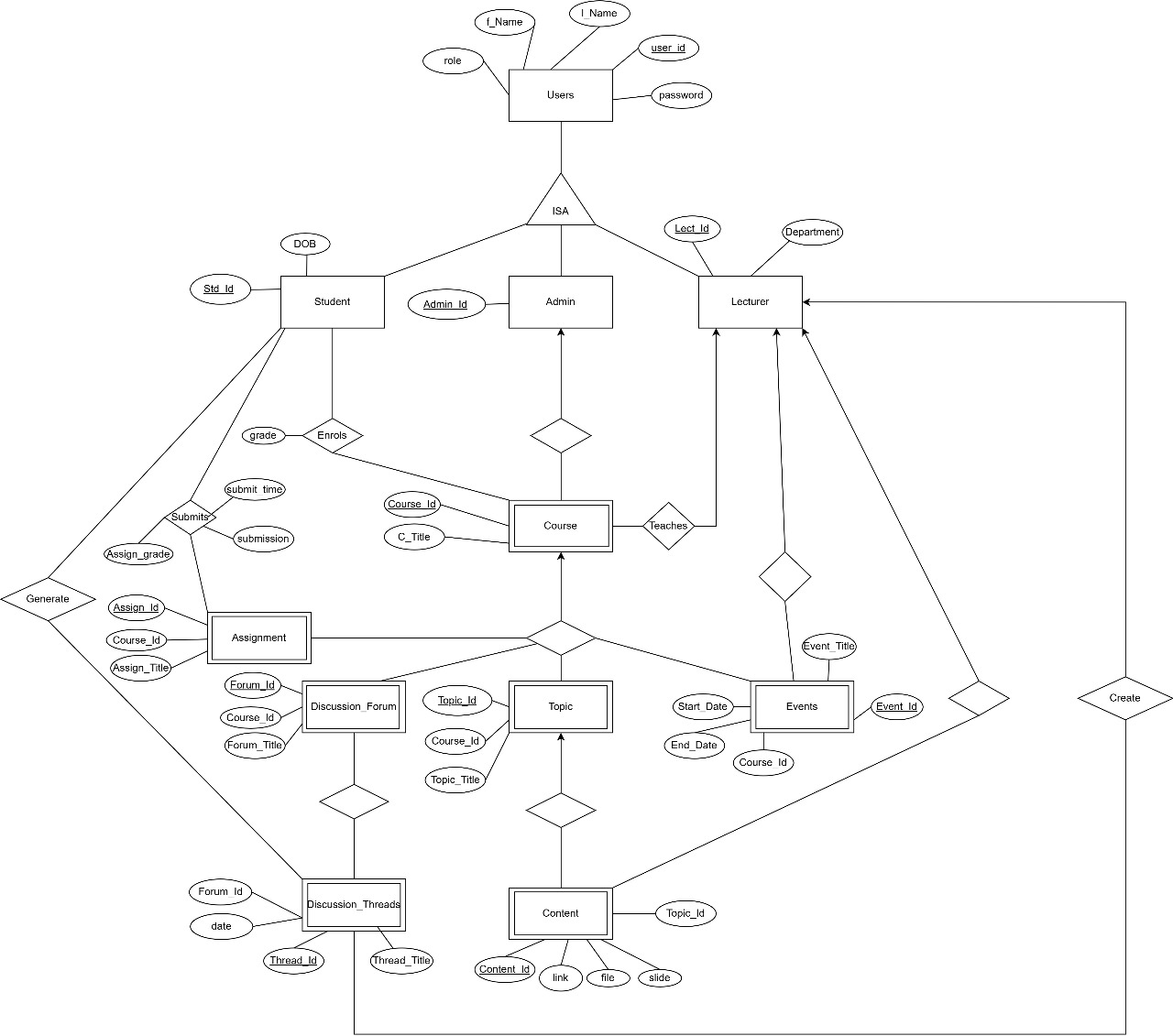
**COMP3161 Group Project**

**Group 20**

**Table Showing Group Members and Respective Assignments**

|  |  |  |
| --- | --- | --- |
| **Members** | **Student ID#** | **Assignments** |
| Abegail McCalla | 620157646 | Create Database (Create and Populate SQL Tables)  API: Add Grade to Assignment, Add Student Average, Retrieve Courses with Over 50 students, Retrieve Students with Over 5 Courses, Retrieve Students with Over 3 courses, Retrieve of the Top 10 MOST enrolled courses, Retrieve Top 10 student with the HIGHEST average grades  Frontend  Postman Collection  Testing |
| Carville Monteith | 620138741 | API: Register User, Login by User, Create Course, Retrieve Courses, Retrieve Courses by Lecturer’s Id, Register Student, Assign Lecturer to Course, Get Events by Course ID, Retrieve Threads, Create Threads, Reply to Threads, Add Section/Topic,Add Assignment, Retrieve Assignments by Course Id, Submit Assignment by Course Id  Frontend |
| Tanisha Comrie | 620156117 | ERD & Normalization  API:Retrieve All Members,Retrieve Events by Student Id and Start Date, Add Events, Add Forum, Retrieve Course Content by Course Id, Add Course Content, RetrieveForum  Import Docker  Cloud |

**Project Entity Relation Diagram (ERD)**



V

**Relations**

Users (user\_id, f\_Name, l\_Name, role, password)

Admin (Admin\_Id)

Course (Course\_Id, C\_Title, Admin\_Id)

Lecturer (Lect\_Id, Department)

Teaches (Lect\_Id, Course\_Id)

Student (Std\_Id, DOB)

Enrols (Std\_Id, Course\_Id, Grade)

Submits (Std\_Id, Assign\_Id, Assign\_grade, submit\_time, submission)

Assignment (Assign\_Id, Course\_Id, Assign\_Title)

Topic (Topic\_Id, Course\_Id, Topic\_Title)

Content (Content\_Id, Topic\_Id, link, file, slide)

Discussion\_Forum (Forum\_Id, Course\_Id, Forum\_Title)

Discussion\_Threads (Thread\_Id, Forum\_Id, Thread\_Title, date)

Events (Events\_Id, Course\_Id, Start\_Date, End\_Date, Event\_Title)

**Assumptions**

* When you are replying to a thread in a thread-rely situation you are still creating a thread thus there is no attribute to link to the thread in which you are responding.
* A new topic is created when a new content is created and vice versa.
* Not necessary to keep track of which lecturer created which course content or event. Only necessary that a lecturer can create a course content or an event.
* Not necessary to keep track of which lecturer or student created which thread. Only necessary that a lecturer or a student can create a thread.
* A thread is dependent on either a lecturer or a student but does not depend on both at the same time to exist.
* The post that started the thread can be a forum ID for a newly created thread and a thread ID for a reply thread.
* Since a lecturer **or** a student can create a thread, it is not necessary for the lecturer **and** the student to be able to create a thread in the frontend.
* Not every action has to be a table or even a relationship

**Possible Future Improvements**

* Using stored procedures and indexes
* Using authorisation mechanisms such as JWT tokens
* Proper error handling

**Project GitHub Link:** https://github.com/abegailmccalla/UWI-COMP3161-Final-Project.git