by



**Group Details**

|  |  |  |
| --- | --- | --- |
| Team Name | Fuzzy Logic | |
| Team Members | **Talha Vawda**  Luqmaan Haffejee  Azhar Mohamed  Ahmad Jawaad Shah | |
|  | | |
| SQL Server Login Details | Username | ist2hw |
| Password | ufjufh |
|  | | |
| Application Login Details  These are login details you can use to test the application for different types of users. However should you wish to test the application with other users, their username (Student/Admin number) and password can be obtained from the Student and Admin tables. | Administrator | |
| Username | 1234567890 |
| Password | Jkane123 |
| Student 1 (enrolled 2019; registered) | |
| Username | 2019000001 |
| Password | collins98243 |
| Student 2 (enrolled 2019; not registered) | |
| Username | 2019360606 |
| Password | brett22 |
| Student 3 (enrolled 2017; registered for 2019) | |
| Username | 2017000001 |
| Password | yolo1 |
| Student 4 (enrolled 2017; not registered for 2019) | |
| Username | 2017000006 |
| Password | harvarduni4me |

**Project Background**

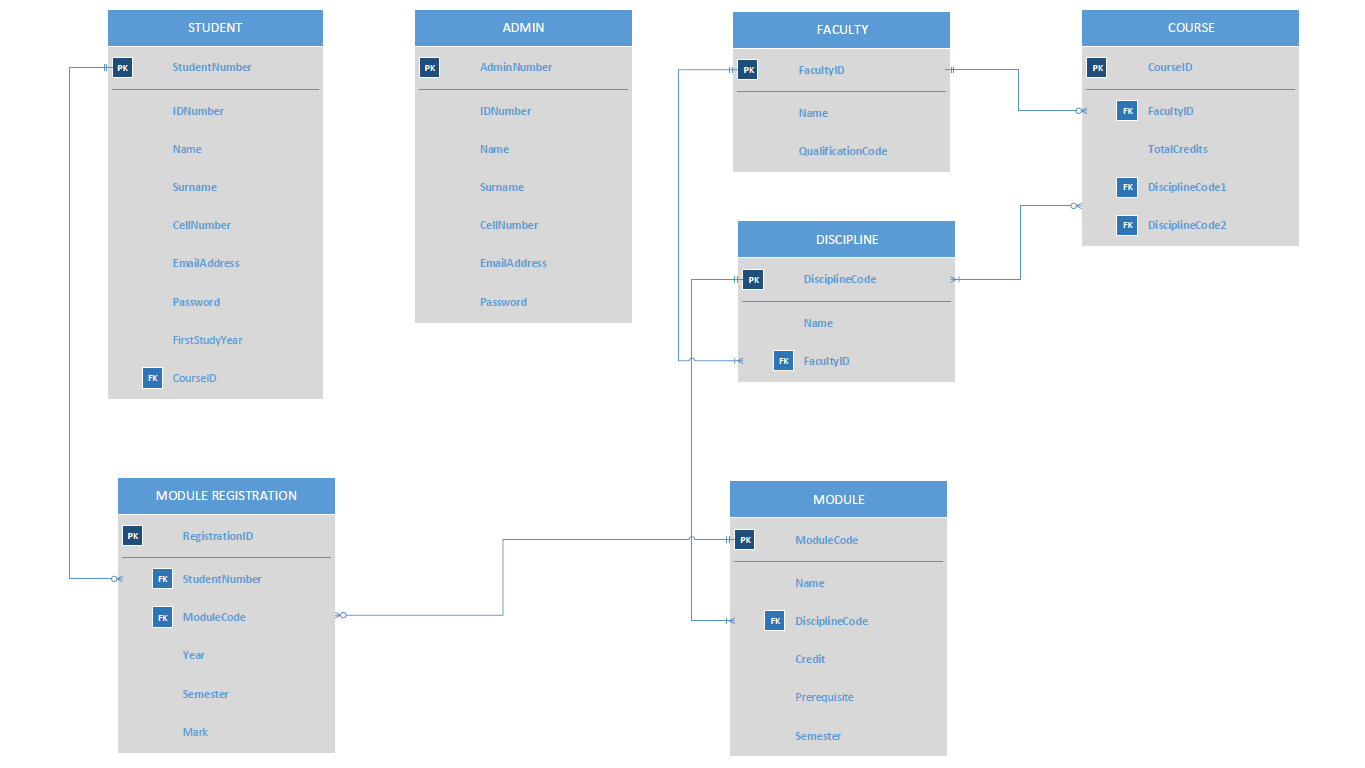
Fuzzy Logic has undertaken the task of researching, designing, and developing an Information System for Imperial College that allows them to manage the personal and registration details of their student cohort. We have thus aptly named this project Student Management System.

Fuzzy Logic was approached by Imperial College, a relatively new institution, because they needed a bespoke application tailored for them as the shrinkwrap solutions they had tried to implement did not seem to fulfil their requirements.

We have created two user types for the application, a student and an administrator, as it is more practical and efficient for each student to enrol themselves onto the system (adding their details and selecting their majors) and registering than for an admin to add each student.

|  |  |  |  |
| --- | --- | --- | --- |
| **STUDENT** | | **ADMIN** | |
| Add personal details | Create |  |  |
| Select two disciplines as their majors | Create |  |  |
| Register for both semesters of the current year | Create |  |  |
| View their own personal details | Read |  |  |
| View their past registrations | Read |  |  |
| View their marks for the modules that they have registered for and completed | Read |  |  |
| Change some of their personal details | Update |  |  |
| Change their password | Update |  |  |

**Program Functionality (CRUD)**

**Entity Relationship Diagram**

**SQL Statements**

1. **An SQL statement that extracts and displays all data from a database table**

SELECT Name FROM FACULTY

*Display all the Faculties to a student when they are enrolling so that they can select which faculty they want to go into and from there they can select 2 Disciplines as their majors*

1. **An SQL statement that will update values of a table**

UPDATE STUDENT

SET Password = @Password

WHERE (StudentNumber = @StudentNumber) AND (IDNumber = @IDNumber) AND (CellNumber = @CellNumber)

*Allow the student to change (update) their password*

1. **At least three SQL statements with a WHERE clause**
2. Get all Student Numbers’ of the students that have passed a particular module

SELECT StudentNumber, ModuleCode, Year, Semester, Mark, RegistrationID

FROM [MODULE REGISTRATION]

WHERE StudentNumber = @StudNum AND Mark >= 50

*The administrator can use this for administrative purposes*

1. Add the final mark for a particular module for a particular student

UPDATE [MODULE REGISTRATION]

SET Mark = @Mark

WHERE (StudentNumber = @StudentNumber) AND (Year = @Year) AND (ModuleCode = @ModuleCode)

*It is the administrator’s job to enter students’ final marks. This SQL query which is part of the code will add (update; -1 represents no mark) the mark for a student for a module they did in a particular year*

1. Get the course that a student has registered for:

SELECT CourseID FROM STUDENT WHERE (StudentNumber = @StudentNumber)

*This will be used to get the 2 majors of the student and the qualification type which will be displayed in the student’s details tab and also on their academic record*

1. Get all modules belonging to a specific discipline

SELECT ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester

FROM MODULE

WHERE (DisciplineCode = @Disc)

*This is used to display all the modules in a combo box based on a discipline selected so that the administrator can select a module to view statistics and analytics on that module*

1. **At least two SQL subqueries**
2. Get available Semester 1 modules for a specific year based on completion of prerequisites

SELECT ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester  
FROM MODULE  
WHERE (Prerequisite IN (SELECT ModuleCode FROM [MODULE REGISTRATION]WHERE (StudentNumber = @StudNo) AND (Mark >= 50) AND (Year = @Year))) AND (DisciplineCode = @Major1 ORDisciplineCode = @Major2) AND (Semester = 1)

*Display all the modules a student can register for semester 1 of the year so that they can register for that semester. This is done by selecting all the modules that are part of their majors and filtering out all the modules they cannot do (they can only do a module if they passed the prerequisite module for that module [First-year modules have no prerequisite])*

1. Get available Semester 2 modules for a specific year based on completion of prerequisites

SELECT ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester  
FROM MODULE  
WHERE (Prerequisite IN (SELECT ModuleCode FROM [MODULE REGISTRATION]WHERE (StudentNumber = @StudNo) AND (Mark >= 50) AND (Year = @Year))) AND (DisciplineCode = @Major1 ORDisciplineCode = @Major2) AND (Semester = 2)

*Display all the modules a student can register for semester 2 of the year so that they can register for that semester. This is done by selecting all the modules that are part of their majors and filtering out all the modules they cannot do (they can only do a module if they passed the prerequisite module for that module [First-year modules have no prerequisite])*

1. **At least two aggregation queries**
2. Get number of student registered for a module

SELECT COUNT(\*) AS TotalStudents

FROM [MODULE REGISTRATION]

WHERE (ModuleCode = @Mod) AND (Year = @Year)

*The administrator can view the number of students registered for a particular module which they can use to determine the appropriate classroom the have the lectures for that module in, and also to plan seating arrangements for exams*

1. Get the highest mark for a specific module in a specific year

SELECT MAX(Mark) AS Highest

FROM [MODULE REGISTRATION]

WHERE (ModuleCode = @Mod) AND (Year = @Year) AND (Mark <> - 1)

*The administrator can use the highest mark for a module to evaluate performance of that module and to award Certificate of Merit to the student with the highest mark*