Learning Management System

DATABASE IMPLEMENTATION

**Prepared by: Daniel Obiero**

**UHD CS3321**

**Version 1.0: 02/28/2019 (Created by: Daniel Obiero)**

**Version 2.0: 04/16/2019 (Updated by: Bariscan Yolcu)**

**Version 3.0: 4/23/2019 (Updated by: Ruthie McCasland)**

**Table of Contents**

[1. Overall Description 4](#_Toc524882795)

[1.1 Mission Statement 4](#_Toc524882796)

[1.2 Mission Objectives 4](#_Toc524882797)

[1.3 Major User Views 5](#_Toc524882798)

[1.4 Product Functions 6](#_Toc524882799)

[1.4.1 Login/Logout 6](#_Toc524882800)

[1.4.2 View/Add/Edit/Delete 6](#_Toc524882801)

[1.4.3 Create Students/Teachers/Courses/Classes 6](#_Toc524882802)

[1.4.4 Assign Students/Teachers/Courses 6](#_Toc524882803)

[1.4.5 Create reports 6](#_Toc524882804)

[1.5 Use Case Diagram 8](#_Toc524882805)

[1.6 Entity Relation Model 10](#_Toc524882806)

1.7 CRC Cards 11

[Definitions, acronyms, and abbreviations 13](#_Toc524882807)

[Current procedure 14](#_Toc524882808)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
| Bariscan Yolcu | 04/16/19 | Created fully descriptive Use Cases for each Entity | 2.0 |
| Ruthie McCasland | 4/23/19 | Updated UML Diagram to include the new grades entity that was created in our database | 3.0 |
| Ruthie McCasland | 4/23/19 | Added CRC Cards for each of the established classes | 3.0 |
| Ruthie McCasland | 4/23/19 | Created current procedure that includes testing procedure, functioning components and components that are needing to be added | 3.0 |

# Overall Description

## Mission Statement

The purpose of the Learning Management Database System (LMDBS) is to maintain data that is used and generated to support the academic participants educational delivery.

## Mission Objectives

The Learning Management Database System (LMDBS) will be a Database system for students, educators and administrators. This document details the requirements of the system based on research of teaching institutions and learning management systems that are currently in the market. This report provides user characteristics and primary functions of the Database system.

Mission Objectives for the Learning Management Database System

* To maintain (enter, update, and delete) data on teachers
* To maintain (enter, update, and delete) data on students
* To maintain (enter, update, and delete) data on courses
* To maintain (enter, update, and delete) data on semesters
* To maintain (enter, update, and delete) data on programs
* To track the number of courses assigned to teachers
* To track the number of courses assigned to students
* To track the time the courses are delivered
* To track the semester that courses are delivered
* To track the student GPA
* To track the student grades
* To report on student GPA
* To report on student grades
* To report on courses assigned to students
* To report on courses assigned to teachers
* To report on courses offered during a semester
* To report on completion of program courses

## Major User Views

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data** | **Access Type** | **Administrator** | **Teacher** | **Student** |
| **All Students** | Maintain | x |  |  |
|  | Query | x |  |  |
|  | Report | x |  |  |
| **Single Teacher** | Maintain | x | x |  |
|  | Query | x | x | x |
|  | Report | x | x |  |
| **All Students** | Maintain | x | x |  |
|  | Query | x | x |  |
|  | Report | x | x |  |
| **Single Student** | Maintain | x |  |  |
|  | Query | x | x | x |
|  | Report | x | x | x |
| **All Courses** | Maintain | x |  |  |
|  | Query | x | x |  |
|  | Report | x | x |  |
| **Single Course** | Maintain | x | x | x |
|  | Query | x | x | x |
|  | Report | x | x | x |
| **All Classes** | Maintain | x |  |  |
|  | Query | x | x |  |
|  | Report | x | x |  |
| **Single Class** | Maintain | x | x | x |
|  | Query | x | x | x |
|  | Report | x | x | x |
| **All Course Grades** | Maintain |  | x |  |
|  | Query | x | x |  |
|  | Report | x | x |  |
| **Single Course Grade** | Maintain |  | x |  |
|  | Query | x | x | x |
|  | Report | x | x | x |
| **Class Report** | Maintain | x | x |  |
|  | Query | x | x |  |
|  | Report | x | x |  |
| **Course Grade Report** | Maintain |  | x |  |
|  | Query | x | x | x |
|  | Report | x | x | x |

Table 1.3 Major user views for the LMDBS.

## Product Functions

Participants will log in to the system to gain access to menu functions based on preset user privileges. The product will allow users to enter critical data at the beginning of the semester.

Critical data include:

* Student Information
* Teacher Information
* Administrator Information
* Course Information
* Class Information

### Login/Logout

This function will allow users to access different levels of the system based on their level of privilege. The user will be required to provide a username and password. At the end of user session, they are required to logout of the system. The logout function terminates the user session automatically if left unused for fifteen minutes. The operating system log-out function will be set to lock the computer after one minute of idle time.

### View/Add/Edit/Delete

With appropriate permission, the users will be able to perform these tasks based on user credentials.

### Create Students/Teachers/Courses/Classes

This function will allow the administrator to create new Students/Teachers/Courses into the system for related access. The students will be able to assign themselves the courses pertinent to their programs.

### Assign Students/Teachers/Courses

This function will allow the administrator to assign:

* Courses to Teachers
* Courses to Classes

### Create Reports

The end of semester registration report will include:

* Number of students registered
* Number of teachers registered
* Students registry per course
* Teacher registry per course
* Course time table

## Use Case Diagram

A close up of a map
Use Case Diagram

Figure 1.5 Use Case Diagram for LMDBS

**A list of uses cases that include actor and step-by-step descriptions:**

Use Case: Administrator adds a new student into the school’s database.

Actor: Administrator

Steps: Administrator creates a new student profile with a new student ID, default GPA (4.0) and any course the student has enrolled in into the database.

Use Case: Administrator adds new Teacher to database.

Actor: Administrator

Steps: Administrator creates new teacherID and name for the new incoming teacher and the system will create a new teacher in the database ready to be assigned to any courses that they will teach.

Use Case: Administrator adds new Course to database.

Actor: Administrator

Steps: Administrator creates new courseID and name for the new course being added to the school and the system will create a new course in the database ready to be assigned to any students/teachers.

Use Case: Administrator adds new Program to database.

Actor: Administrator

Steps: Administrator creates new Program (ex: Computer Science) with course name/ID for the new Program being added to the school and the system will create a new Program option in the database ready to be assigned to future courses belonging to the program.

Use Case: Teacher declares a course grade for a particular student.

Actor: Teacher

Steps: Teacher calculates and updates new course grade for a particular student in a course they are teaching which updates for the student in the database and teacher updates a report card for each student.

Use Case: Teacher declares a course to teach for the upcoming semester and checks attendance.

Actor: Teacher

Steps: Teacher declares a course to teach for the upcoming semester and the teacher will be assigned a course from the list of available courses in the database which will give the teacher access to regular course functions including ability to take attendance from list of students enrolled in the course. Attendance will be taken by pulling up the list of students enrolled in their course and cross referencing with who showed up to class.

Use Case: Teacher generates full student report.

Actor: Teacher

Steps: Teacher generates full student report by accessing database and returning everything related to the student including all course grades and GPA which is returned by the database for the full report view.

Use Case: Student enrolls in a new course.

Actor: Student

Steps: Student selects a new course to enroll in which updates the database by assigning the selected course to the student and also updating the student’s full report that can be accessed at any time. Student’s GPA is affected by performance in this selected course and is assigned by the teacher.

Use Case: Student drops an existing course they are currently enrolled in.

Actor: Student

Steps: Student selects a course they are currently taking and drops the course from their list of assigned courses. The database will updates the changes and the student will no longer be assigned to the course and their GPA will not be affected by this course anymore.

Use Case: Student views grades.

Actor: Student

Steps: Student pulls up grades from database by selecting the course in which they want to see their assigned grade by the teacher and the database returns the grade and also updated GPA for the student in the student view.

Use Case: Student enrolls in the School and fills admission.

Actor: Student

Steps: Student chooses to enroll in the School and is added to the School database by an Administrator and the student is also updated into the Student database for the School who are eligible to start choosing classes and have a full student report available. Student must fufill this step in order to add/drop any courses.

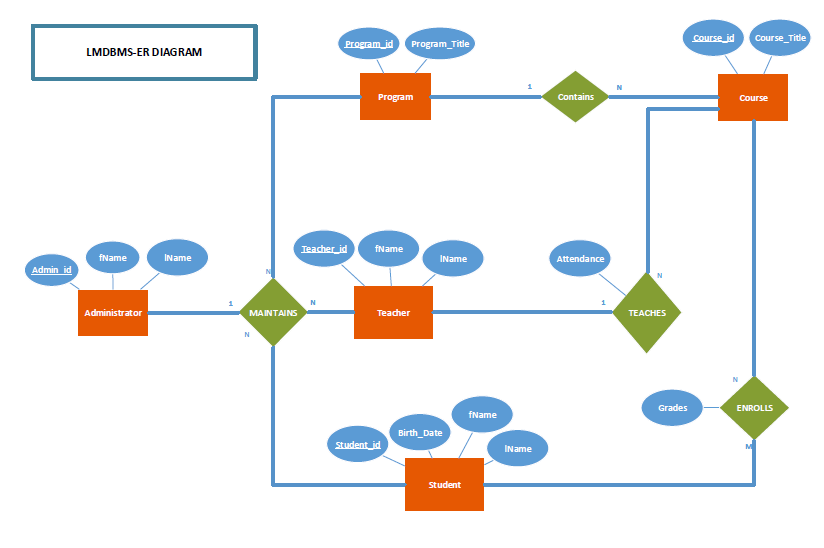
Use Case: School generates full teacher report in order to analyze workload.

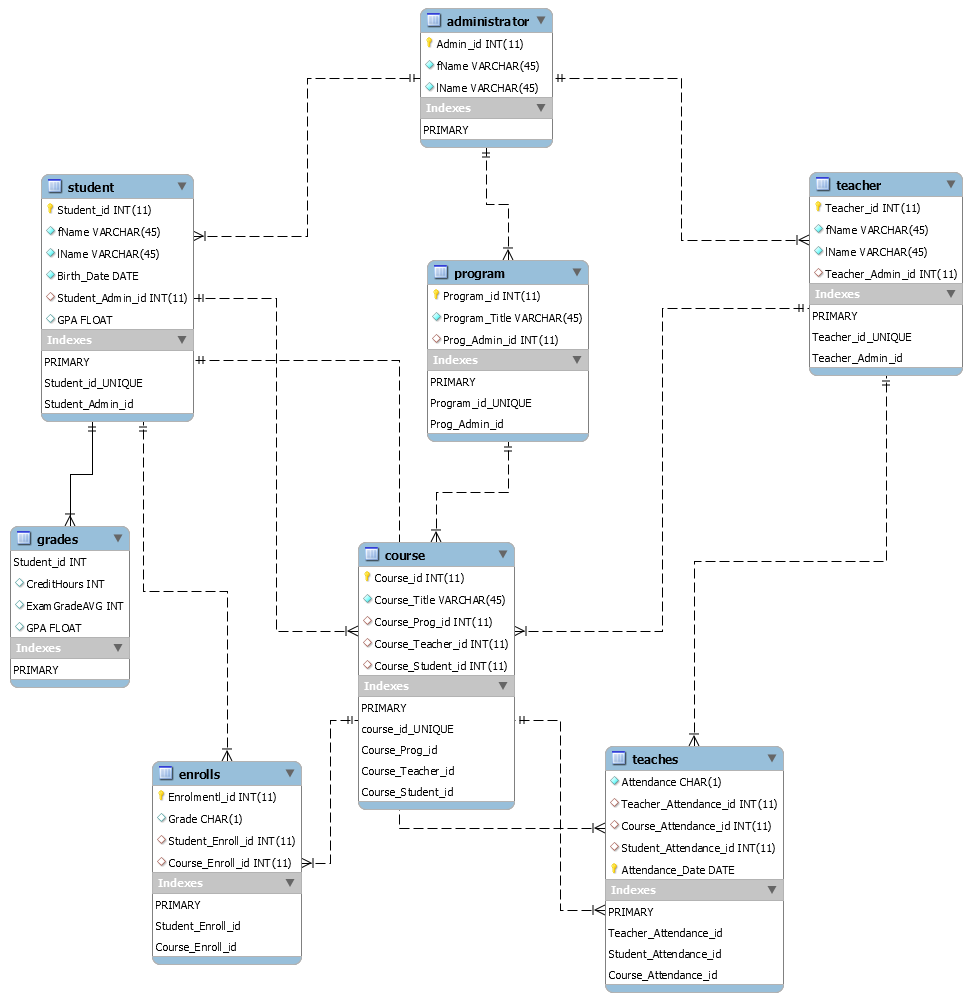
Actor: School

Steps: School actor will generate a full report on a specific teacher and the database will return every single course the teacher is currently teaching or has taught prior. This allows School officials to see the current and previous workloads of any of the teachers in the School view.

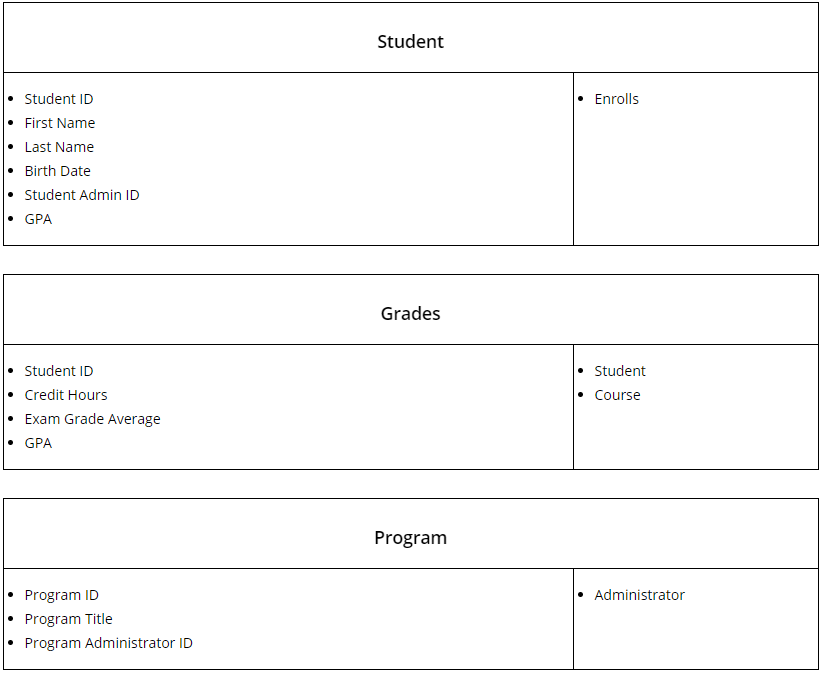
## Entity Relation Model

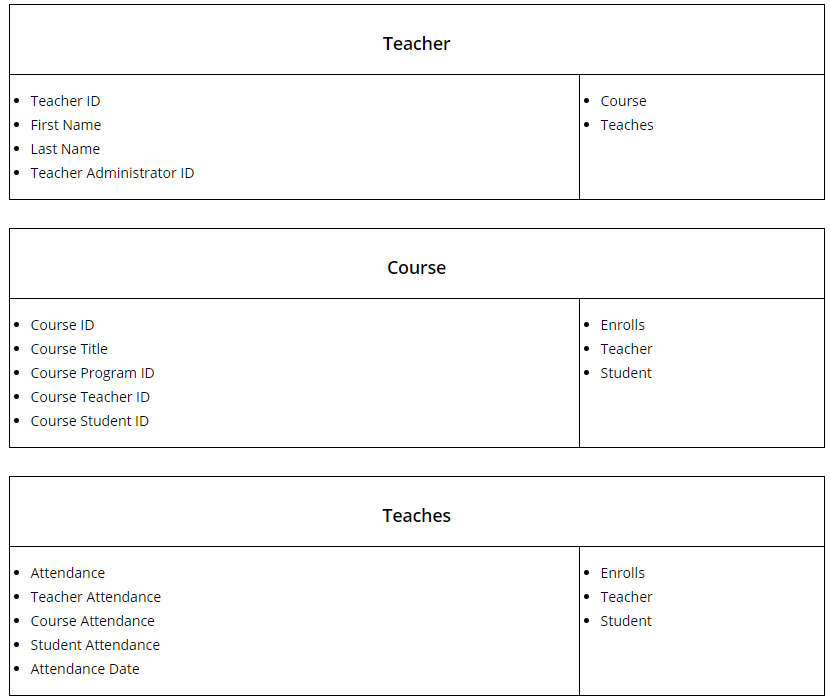
Figure 1.6 Entity-Relationship Diagram of LMDBS





**CRC Cards**

****



**Appendix A:**

## Definitions, acronyms, and abbreviations

Administrator Persons responsible to manage the database

School Place where courses are delivered

Courses Subjects associated with programs

LMDBS Learning Management Database System

System Learning Management Database System

Student Subjects enrolling in the programs to learn

**Current Procedure:**

***Testing Procedure***

Testing is to be done by both the person working on the code, and by a peer that has not worked on the specific code that is being tested. Testing needs to be done frequently and thoroughly. Any major issues with the code needs to be discussed with the entire team and documented. All testing is to be done:

* As any single unit of the program is completed, to ensure all is working as intended.
* As you implement a single unit with a different unit or implement a unit to a larger part of the entire program, to ensure all parts are working well together and as intended.
* To any line of code that is modified.
* When deleting any code.
* When adding any new chunks of code to existing code.

***Functioning Components***

* The current running program allows us to login and check for username and password. If all is valid, you are granted entry to the main screen for administrator
* The current program only has a view for administrator that is able to manipulate and add all data.
* Ability to add new students, see current student list, look at general student stats of all students entered in the database, edit/delete/manage students.
* Ability to add new course, edit/delete/manage courses.
* Ability to add new programs, edit/delete/manage courses.
* Ability to add new teacher, edit/delete/manage teachers.
* Database is able to store all information inputted, with no known issues.

***Components to be Added***

* Need to be able to look at current teacher list, and be able to look at general teacher stats of all teachers entered in the database (same manipulation powers as the student column)
* Be able to show who is all currently enrolled in each course, both students and teachers