# SRS Document

# (Team 7) – Faculty MIS

TA: Menna Maged

Habiba Ahmed Elbadawy – Section 3

Amr Akram – Section 2

Mina Maged – Section 3

Omar Amgad – Section 2

Saif AbdelElah – Section 2

Date: 16-5-2021

1. ***Introduction***

Each faculty is in need for a system software to manage, control and record each process done. Faculty MIS is a software which covers the needs of any faculty from managing the available courses and displaying the important information for the stakeholders to computing and calculating grades and fees for students. Faculty MIS enables the faculty to reduce the time lost in extracting any information from the database such as extracting a student’s telephone number, SSN, grades or GPA. The system offers the faculty with many features from the financial perspective such as calculating the revenues and expenses of the year and computing the profits as all the records are saved in the same place. The system also keeps track of the performance of all students and send them warnings which improves the educational process.

The system has many other features such as creating schedules for students and professors and enabling professors and TAs to send announcements to the students.

The system is connected with other inherited systems such as Masters Management System, Admissions System and others, also the system should be connected with the database of the faculty to ensure processing with the latest updates, in order to do any process the system gets input which is usually from the database and process on it then displays the output so the system should have a direct connection with the database, also the system will need some developers to be responsible for working on it for future maintenance in case the faculty needs to make huge updates for the system. To keep this system secure, each stakeholder should have an ID and password but, each account has authorities different from each other according to their position, as the admin or the developer has more access to the system than the student.

The system should be easy to use, reliable, has high execution speed and capable to allow many users to be online at the same instant without breaking down.

# Context Diagram

# User Requirements

* + - Manages the available courses & their timings.
    - Generates a separate schedule for each student, Dr. & TA depending on the courses they take/give.
    - Calculates the fees to be paid by student per semester.
    - Displays the grades of each student & their GPA.
    - Sends warnings to students who do not pay the fees and/or have a GPA < 2.0.
    - Allows Drs or TAs to share announcements that can be accessed by all students.
    - Calculates profits and expenses per semester.
    - Every student is allowed to register a maximum of 15 credit hours every semester.
    - Admin can access and edit information of courses offered and students enrolled using his/her username and password.

# Functional Requirements

**Calculate Fees:**

* + - Description: Computes fees for each student by multiplying number of credit hours by price per hour.
    - Input(s): Number of credit hours, cost of each hour.
    - Source (source of inputs that will be entered into the function): Number of hours from faculty UMS website. Price/hour from memory.
    - Pre-condition (what should be true so that function can start): Student must register in courses.
    - Post-condition (what will happen after the function execution): New fees replace old fees (null/of past semester).
    - Output (what will be displayed/generated after function execution): Fees to be paid displayed to student.

**Generate Schedule:**

* + - Description: A schedule is made for each student in faculty.
    - Input(s): Courses registered by student.
    - Source (source of inputs that will be entered into the function): Registered courses from faculty UMS website.
    - Pre-condition (what should be true so that function can start): Courses are registered.
    - Post-condition (what will happen after the function execution): New schedule replaces old schedule (null/of past semester).
    - Output (what will be displayed/generated after function execution): Display semester schedule.

**Calculate Grades:**

* + - Description: Compute each student grades for the current semester by adding midterm grade, year work grade and final exam grade.
    - Input(s): Midterm exam mark, year work mark, final exam mark.
    - Source (source of inputs that will be entered into the function): Faculty UMS website.
    - Pre-condition (what should be true so that function can start): All grades are available.
    - Post-condition (what will happen after the function execution): Each student grade is finalized.
    - Output (what will be displayed/generated after function execution): Grade of every course is displayed to student in addition to his/her GPA.

**Calculate Number of Absences:**

* + - Description: Check number of absences of each student in Tutorials/Labs recorded by TAs daily.
    - Input(s): Attendance sheet.
    - Source (source of inputs that will be entered into the function): Memory.
    - Pre-condition (what should be true so that function can start): Records of attendance needed are available.
    - Post-condition (what will happen after the function execution): Number of absences of each student are known.
    - Output (what will be displayed/generated after function execution): How many times each student was absent for every course.

**Send Warnings:**

* + - Description: Warn the students with many absences and/or low GPA.
    - Input(s): Cumulative GPA, number of absences.
    - Source (source of inputs that will be entered into the function): Faculty UMS website/ Memory.
    - Pre-condition (what should be true so that function can start): Number of absences exceeds the number allowed, GPA < 2.0.
    - Post-condition (what will happen after the function execution): A warning letter is prepared.
    - Output (what will be displayed/generated after function execution): Letter is sent to official (university) email address of student.

**Contact Dr./TA:**

* + - Description: Search for Dr./TA name to get their email address.
    - Input(s): Name of Dr./TA.
    - Source (source of inputs that will be entered into the function): Course’s info on the faculty UMS Website.
    - Pre-condition (what should be true so that function can start): A name is entered in the search bar.
    - Post-condition (what will happen after the function execution): Matching the name entered with the names stored in the memory.
    - Output (what will be displayed/generated after function execution): Dr./TA (university) email address.

# Non-Functional Requirements

* + - **Security**

Only authorized users, who are given specific rights, are allowed to access data or apply certain actions on it as needed by using their username and password.

* + - **Performance**

System must handle simultaneous usage of a certain number of users at a time. Ex: 1000.

All students registered on the system should be able to use it over an interval of 3 days during Course Registration.

* + - **Ease of use & Documentation**

UI should be user-friendly to easily navigate and use the system.

The system should have a simple precise documentation so that anyone can use it on his/her own and be aware of all its features and functions.

* + - **Backup**

There have to be a copy of data on an external source to prevent data loss.

* + - **Flexibility**

Administration at faculty should be able to easily carry out small modifications/changes without the need to contact the company every time.