

**MAKERERE UNIVERSITY BUSINESS SCHOOL**

**FACULTY OF COMPUTING AND INFORMATICS**

**DEPARTMENT OF COMPUTING AND APPLIED IT**

**ADVANCED WEB APPLICATION DEVELOPMENT**

**COURSEWORK TWO PROJECT**

**GROUP MEMBERS**

|  |  |
| --- | --- |
| **NAME** | **REG NUMBER** |
| ODOI TIMOTHY | 22/U/6767 |
| NATURINDA HUSSINAH | 22/U/6691 |
| NABUNJE SHAMIM | 22/U/6491 |

A **Student Management System (SMS)** is a comprehensive software solution designed to streamline the administrative and academic processes in educational institutions. It enables efficient management of student-related data, simplifying tasks such as admissions, attendance tracking, grade management, and communication.

By integrating various functions into a centralized platform, this system helps schools, colleges, and universities to manage student information securely and efficiently, reducing paperwork and improving data accuracy.

Key features often include student enrollment and registration, attendance monitoring, grade recording, fee tracking, and communication modules for connecting students, parents, and faculty. The system aims to enhance productivity and foster transparency, providing administrators, teachers, and students with easy access to essential information. A well-designed Student Management System ultimately supports a more organized, data-driven, and user-friendly educational environment, allowing institutions to focus more on quality education and student success.

**KEY FEATURES OF HTS STUDENT MANAGEMENT SYSTEM**

**Student Records Management**

* Centralized storage of student information, including personal data, academic history, courses done
* Allows easy updates to student profiles, including contact details, name, courses done, emails etc

**Course Management**

* Facilitates the creation, modification, and organization of courses, including course prerequisites, credit hours etc
* Tracks student enrollment in courses and monitors their progress toward graduation requirements.

**Grade and Examination Management**.

* Provides automated grade calculations, report cards, and transcripts based on defined grading criteria.

**Report Generation and Analytics**

* Offers data visualizations, dashboards, and summaries to aid in analyzing student and school performance trends.
* Produces reports for regulatory compliance, accreditation, and administrative use.

.

**Document and Records Management**

* Manages digital copies of important student information
* Ensures secure, centralized storage of documents with easy access for staff and authorized users.
* Generates documents on demand, helping streamline administrative processes.

**Student Management System Requirements**

**1. Functional Requirements**

**1. Student Information Management**

* The system shall allow administrators to create, read, update, and delete (CRUD) student profiles, including personal details, emergency contacts, and medical information.
* The system shall allow students to view their personal profile details, including academic history.
* The system shall allow administrators to keep information about courses through using the CRUD (create ,read ,update and delete) functionality

**2 .Academic History and Progress Tracking**

* The system shall store students' grades, information and academic progress for each semester
* The system shall provide a dashboard for students to view progress in each course.

**3. Class and Course Registration**

* The system shall allow students to register for available classes, view class availability, and add/drop courses.
* The system shall allow administrators to update course availability and students information.

**2. Non-Functional Requirements**

**1 .Usability**

* The system shall have an intuitive, user-friendly interface accessible by all user types (students, parents, administrators).
* The system shall support mobile devices for on-the-go access.

**2. Performance**

* : The system shall handle up to 10,000 simultaneous users without performance degradation.
* Data retrieval (e.g., profile view or attendance report generation) shall take less than 2 seconds.

**3. Reliability**

* The system shall be available 99.9% of the time, ensuring minimal downtime.
* Backup data shall be created daily to prevent data loss.

**4. Security**

* The system shall ensure role-based access control, with permissions for administrators.
* The system shall use secure encryption (e.g., SSL/TLS) for all sensitive data transfers.
* All user data shall be encrypted and stored securely, following data protection regulations.

**5. Scalability**

* The system architecture shall support scalability to accommodate future growth in the number of users and data.

**6 .Maintainability**

* The system shall allow easy updates to features or data structures, enabling administrators to adapt to evolving school requirements.

**7. Compatibility**

* The system shall be compatible with major web browsers (Chrome, Firefox, Safari, Edge).

**Student Management System (SMS) architecture** **diagram and its components**:

**System Architecture Diagram Components**

1. **User Interfaces**
   * **Student Portal**: Access for students to view student lists, assignments, grades, and notifications.
   * **Admin Portal**: System administration and configuration, managing users, course schedules, and reporting.
2. **Front-End Components**
   * **Web Application (HTML/CSS/JavaScript)**: Provides responsive user interfaces for students, parents, teachers, and admins.
3. **API Gateway**
   * **API Layer**: Manages all communication between the front end and back end, and exposes endpoints for various system functionalities (e.g., user authentication, data access, and notifications).
4. **Application Server**
   * **Authentication and Authorization Service**: Manages user login, session management, and role-based access control.
   * **Student Management Module**: Handles student information storage and management, including profiles.
   * **Reporting and Analytics Module**: Generates reports and analytics dashboards for academic performance, attendance, and other KPIs.
5. **Database Layer**
   * **Relational Database (SQL)**: Stores structured data like student profiles, grades and courses
   * **File Storage**: Stores files such as uploaded documents, assignment submissions, and report cards.
   * **Data Backup and Recovery**: Maintains backups of the database to ensure data recovery and integrity.
6. **Security Components**
   * **Data Encryption**: Ensures sensitive data (e.g., passwords information) is encrypted in transit and at rest.

**Diagram Description**

The architecture is commonly visualized in a layered structure as follows:

* **Top Layer (User Interfaces)**: The user interacts with the system through web applications, which connect to the back-end services through the API Gateway.
* **Middle Layer (Application Server)**: This is the business logic layer containing modules for managing students, courses and reports.
* **Data Layer (Database)**: Databases store all student, courses data, with data backups for resilience.
* **Security Layer**: Ensures secure, role-based access across all components, providing data encryption and secure communication.

**DIAGRAM**

**Frontend (UI)**

Student UI Admin UI

**Database Layer**  
Students Table Lecturers Table Courses Table

**API Server**   
Authentication Service   
Business Logic

**External Services**   
 File Storage

**ENTITIES**

**1. Student**

**2. Course**

**3. Lecturer**

**4. Grade**

**5. Enrollment**

**ATTRIBUTES**

**1. STUDENT**

**Id**

Name

Reg\_no

Email

Phone\_number

Course\_enrolled

2. **COURSE**

Id

Name

Description

Credit\_hours

**3. LECTURER**

Id

Name

Address

Contact

**4. GRADE**

Id

Category

Student\_id

Course\_id

5. **ENROLLMENT**

Id

Course

Date

Student\_id

Status

Semester

**ENTITY RELATIONSHIP DIAGRAM**

#### 

**Student**

**Course**

**Grade**

**Lecturer**

#### RELATIONSHIPS:

* **Students** **to course**:
  + **Relationship**: Many-to-Many
  + **Description**: One or many students can have multiple courses, and each course can be associated to many students.
* **Students to** **Lecturer**:
  + **Relationship**: Many-to-Many
  + **Description**: Each course can have multiple enrollments, but each enrollment is associated with only one course.
* **Grade to lecturer**
* **Relationship:** one-to one or many
* **Description:** one lecturer can assign one or many grades to students
* **Lecturer to course**
* **Relationship:** many to many
* **Description:** One or many lecturers can teach one or many course units and one or many courses can be taught by one or many lecturers.
* **Lecturer to student**
* **Relationship:** many to many
* **Description:** one or many lecturers can teach one or many students and one or many students can be taught by one or many lecturers.