**Web Application Development Project for Educational Management**

Project objective: Develop a web application that allows the efficient management of courses, qualifications and educational activities for a school institution. The application will be designed to address the needs of teachers, parents and students, providing specific modules for each user group.

# Main features

**Teacher Module:**

* Register of teachers with personal and contact information.
* Creation and management of courses.
* Record of grades and feedback for students.
* Scheduling and management of educational tasks and activities.
* Notifications about school events, activities and important announcements.
* Internal communication with other teachers, parents and administrators
* Generation of reports on the academic performance of students in their courses.

**Parent Module:**

* Registration of parents or guardians with relevant information.
* Access to your children's grades and academic progress.
* Notifications about school events, activities and important announcements.
* Communication with teachers and other parents through the platform.

**Student Module:**

* Student registration with personal and contact information.
* Visualization of class schedules and course details.
* Access to grades, progress reports, comments and feedback from teachers.
* Upload of tasks and digital deliveries.
* Participation in forums and academic discussions.

**Administration Module:**

* User Management.
* Management of courses, schedules and school calendars.

# Functional requirements by modules

**Teacher Module:**

**Teacher Registration:**

RF1: Teachers must be able to create user accounts by providing their personal information, such as full name, email address, password, academic grade, teaching grade, subjects taught.

RF2: Teachers should be able to edit their personal and contact information at any time.

RF3: Teachers must be able to log in using their registered credentials.

**Creation and Management of Courses:**

RF4: Teachers must be able to create new courses by providing details such as name, description and educational level, school grade where it is taught (courses must be associated with the teachers responsible for their teaching).

RF5: Teachers must be able to edit and update existing course information.

**Record of Ratings and Feedback:**

RF7: Teachers should be able to record student grades on different assignments and tests.

RF8: Teachers must be able to provide written feedback along with grades.

RF9: Teachers must be able to generate reports on the academic performance of students in their courses.

**Scheduling and Task Management:**

RF10: Teachers should be able to create educational assignments and activities with details like title, description, and due date.

RF11: Teachers should be able to mark assignments complete and assign grades.

**Notifications about School Events:**

RF12: The system must allow the teacher to send notifications about school events, important dates and planned activities related to their courses.

RF13: Teachers should be able to choose their notification preferences and how they want to receive them (email, messages on the platform, etc.).

**Internal communication:**

RF14: Teachers must be able to communicate with other teachers, parents and administrators through messages on the platform.

**Parent module**

**Registration of Parents or Guardians with Relevant Information:**

RF1: Parents or guardians must be able to register on the platform by providing their name, email address, address, contact telephone number and password. In addition, they must be able to complete their registration with additional information such as the number of children in the school and the student code of their child(ren).

RF2: Parents must be able to login using their registered credentials.

**Access to Your Children's Grades and Academic Progress:**

RF 3: Parents must be able to access their children's grades and see their academic progress on the platform.

RF4: Parents must be able to view the grades of different subjects and activities to assess their children's performance.

RF5: Parents should be able to view a grade history and summary of progress over time.

**Notifications about School Events, Activities and Important Announcements:**

RF6: Parents should receive push notifications about school events, key dates, and important announcements.

RF7: Parents should be able to choose their notification preferences and the way they want to receive them (email, messages on the platform, etc.).

**Communication with Teachers and Other Parents Through the Platform:**

RF9: Parents must be able to communicate with their children's teachers through messages on the platform.

RF10: Parents must have the ability to participate in forums and discussions with other parents to share information and experiences.

**Student module**

**Student Registration with Personal and Contact Information:**

RF1: Students must be able to register on the platform by providing their name, email address, student code, and password.

RF2: Students must be able to login using their registered credentials.

**Viewing Class Schedules and Course Details:**

RF3: Students must be able to access their class schedules.

RF4: Students must be able to view the details of the courses corresponding to their school year.

**Access to Teacher Grades, Comments and Feedback:**

RF5: Students must be able to access their grades obtained in homework, exams and other academic activities.

RF6: Students must be able to see the comments and feedback provided by teachers regarding their academic performance.

RF7: Students must be able to view their grades for different subjects and activities to see their performance.

RF8: Students must be able to view a grade history and a summary of progress over time.

**Upload of Tasks and Digital Deliveries:**

RF9: Students must have the ability to upload assignments and digital submissions in their courses.

**Participation in Forums and Academic Discussions:**

RF10: Students must be able to participate in forums and academic discussions related to the courses.

RF11: Students must be able to post questions, comments and answers in the forums, encouraging collaboration and knowledge sharing.

**Administration Module**

**User Management:**

RF1: The system must allow the creation of user accounts for teachers, parents and students.

RF2: Users must be able to log in to the platform using their registered credentials.

RF3: Users must be able to recover forgotten passwords through a secure password reset process.

RF4: The system must validate the uniqueness of email addresses to avoid duplicate registrations.

RF5: The system must verify the authenticity of the email address during the registration process.

RF6: Edit users

RF7: Delete user

**Management of Courses, Schedules and School Calendars:**

RF8: Administrators must have the ability to create courses on the platform.

RF9: Administrators must have the ability to edit courses on the platform.

RF10: Administrators must have the ability to delete courses on the platform.

RF11: Administrators should be able to schedule class times, assign classrooms, and update schedules as needed.

RF12: Administrators must be able to assign classrooms.

RF13: Administrators should be able to update schedules as needed.

# Non-functional requirements

**Security and Privacy:**

* To access the system, robust authentication and authorization mechanisms must be in place to guarantee adequate access to information.
* The system must allow the protection of personal data according to regulations and privacy standards.

**Design:**

* Responsive user interface for use on mobile, tablet and desktop.

**Notifications and Communication:**

* The system must allow notifications by email or messages within the platform to keep users informed.

**Performance:**

* Loading time should be adequate to avoid user frustration.

**Availability:**

* The system must be available most of the time so that users can access it when they need it.

**Scalability:**

* It must be able to support an increase in the number of users without degrading performance.
* It should be possible to increase system resources as needed.

**Usability:**

* The user interface should be intuitive and easy to navigate for all types of users (teachers, parents, students).
* The application must comply with web accessibility standards to ensure that all users can use it.

**Maintainability:**

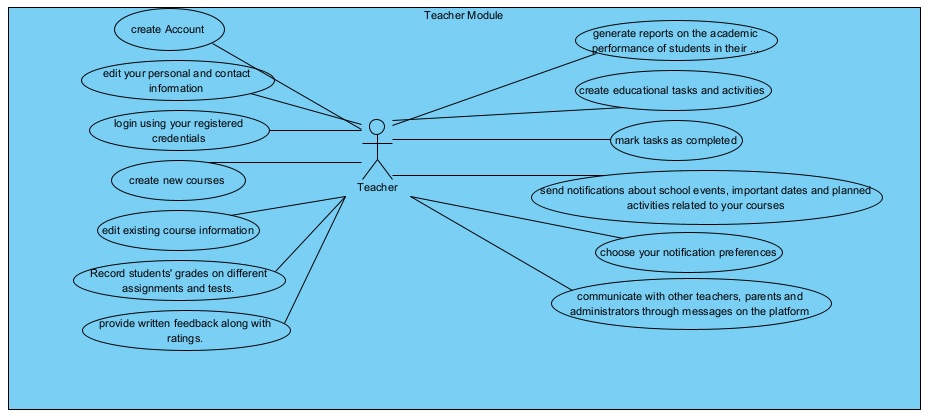
* The code should be well structured in modules to facilitate future updates and changes.
* There should be detailed documentation on the code and architecture to facilitate maintenance by future teams.

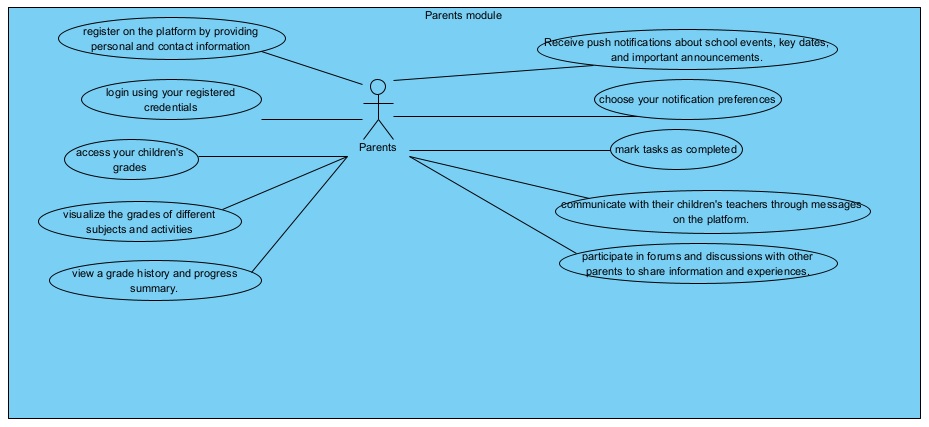
**Browsers and devices:**

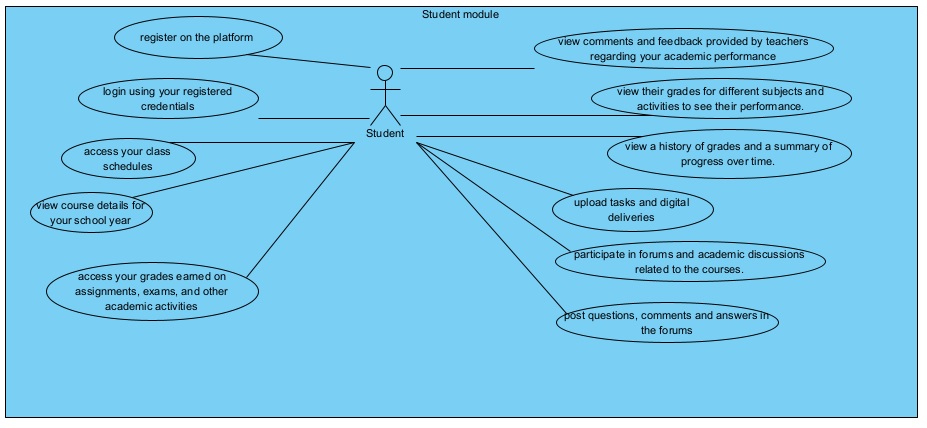
* The app must work consistently across different web browsers and devices.

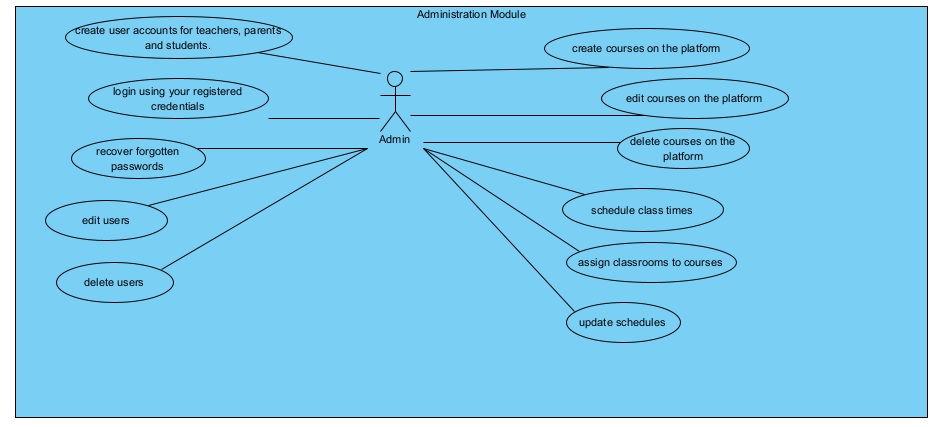
# Requirements analysis

**System Use Case Diagrams**

****

****

****



The requirements analysis includes a detailed description of each functional requirement in terms of use cases for the system. Due to the size of the system, only some cases of critical uses for the system will be selected, so that there is a representation of what a complete requirements analysis would be like.

|  |  |
| --- | --- |
| ID | 01 |
| Name | Create new courses |
| Actors | Teachers |
| Description | Teachers should be able to create new courses by providing details such as name, description, and education level, grade level (courses should be associated with the teachers responsible for teaching them). |
| Use Case Flow | 1. The system displays an interface so that the teacher can register the data of the new course, the fields to complete are: course name, description, educational level (drop-down menu), school grade (drop-down menu).  2. The teacher fills in the information and presses the button  Save course.  3. The system verifies that all the data is complete, that the data is valid and that the teacher does not have another course with the same values. If everything is correct, the course is saved in the DB and displays a success message. If any data is filled with invalid values, the system will not save the course and will indicate the value that must be corrected  4. Finish the use case. |
| Specific requirements | RF4 Teacher Module |
| Preconditions | The teacher must be authenticated in the system. |
| Postconditions | The new course must be saved in the DB. |

|  |  |
| --- | --- |
| ID | 02 |
| Name | Access to qualifications |
| Actors | Parents |
| Description | Parents should be able to access their children's grades and view their academic progress on the platform. |
| Use Case Flow | 1. The system displays an interface so that the parent can view grades previously recorded by a teacher. 2. The parent must select a course and press the Show Grades button. 3. The system verifies which course was selected and the grades recorded in that course for your child are printed on the screen. 4. Finish the use case. |
| Specific requirements | RF3 Parent Module |
| Preconditions | The parent must be authenticated in the system. |
| Postconditions | The system should display the parent options menu |

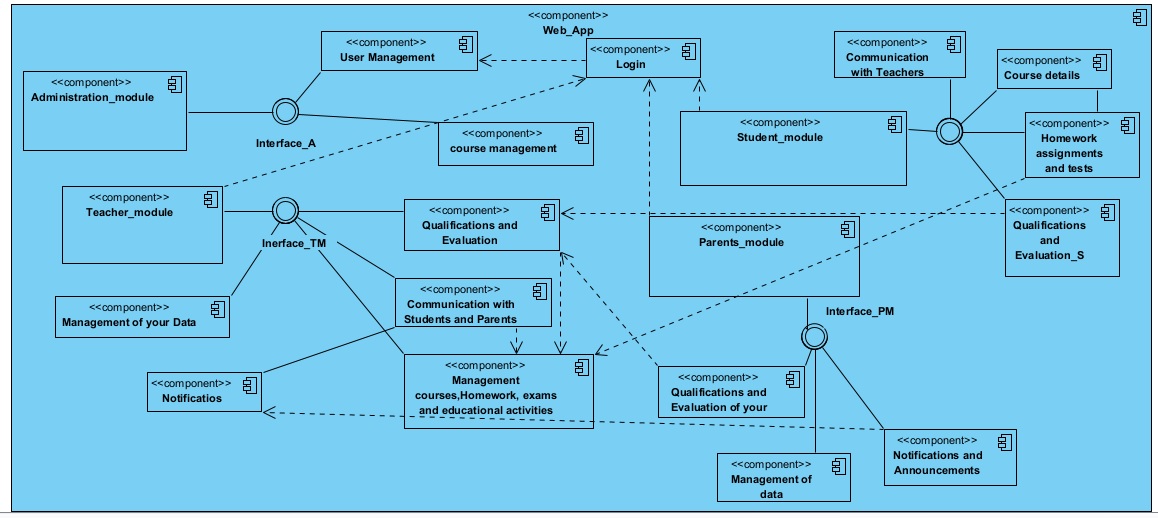
|  |  |
| --- | --- |
| ID | 03 |
| Name | Creating user accounts |
| Actors | Administrators |
| Description | The system must allow the creation of user accounts for teachers, parents and students. |
| Use case flow | 1. The system shows an interface so that the admin can register the data of the new account, the fields to complete are: first select the type of user (teacher, parent, student), depending on the type of user, this will be the data to to complete. (See RF of user registration of each module).  2. The administrator must complete the information and press the Register User button.  3. The system verifies that all the data is complete, that the data is valid and that there is no other user with the same email address. If everything is correct, the user is saved in the DB and shows a success message. If any data is filled with invalid values or the mail already exists in the database, the system will not save the user and will indicate the value that must be corrected.  4. Finish the use case. |
| Specific requirements | RF1 Administration Module |
| Preconditions | The administrator must be authenticated in the system. |
| Postconditions | The system must save the new account created in the DB |

|  |  |
| --- | --- |
| ID | 04 |
| Name | Upload tasks and digital deliveries |
| Actors | Students |
| Description | Students must have the ability to upload digital assignments and submissions in their courses. |
| Use Case Flow | 1. The system displays an interface for the student to upload their assignment, it will allow them to select the file from a location on their device  2. The student must upload the corresponding file and select the Send button.  3. The system checks that the selected file is in the defined format ( pdf , doc , jpg , zip , rar ) and that the size does not exceed the allowed amount. If both conditions are met, the file is sent, otherwise the system will throw an error message.  4. Finish the use case. |
| Specific requirements | RF9 Student module |
| Preconditions | The student must be authenticated in the system. |
| Postconditions | The system must save the assignment uploaded by the student in the DB |

# System design

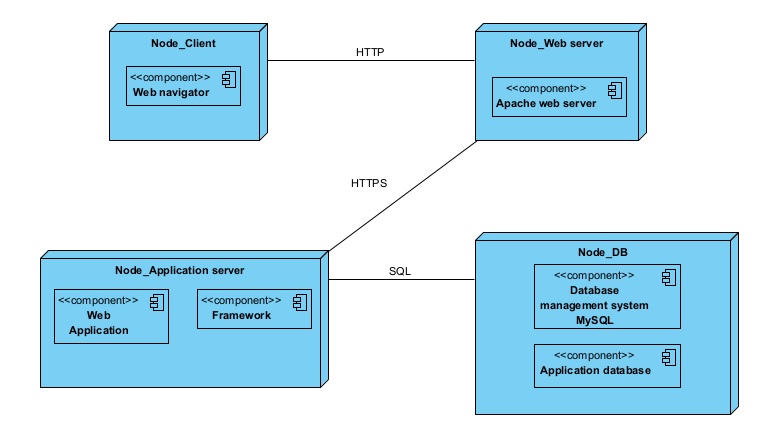
Component diagram

This component diagram describes the elements that make up the web application in terms of components

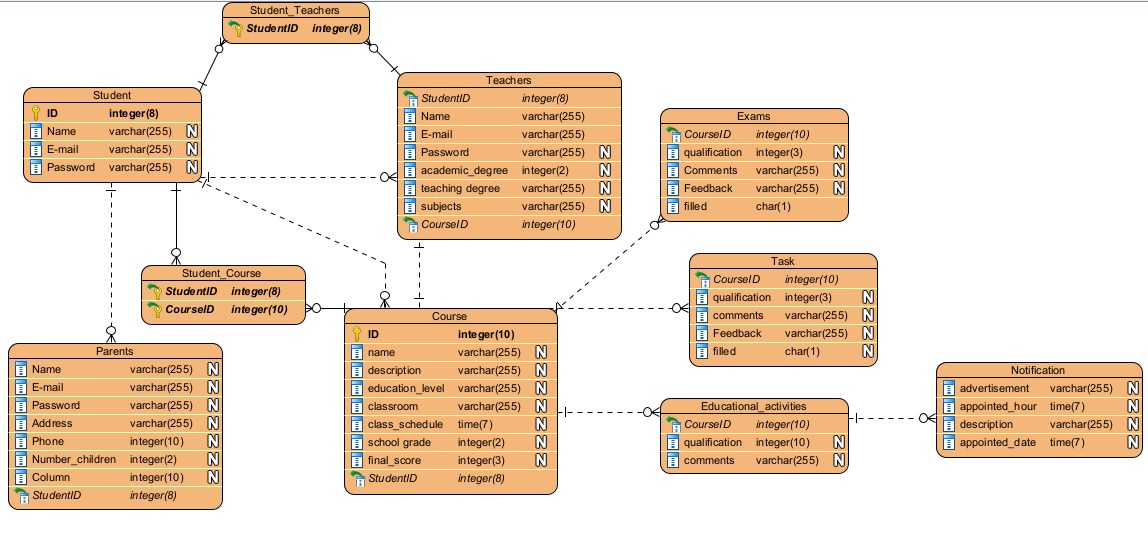


**Deployment diagram with components**

Below is a componentized deployment diagram that represents how components of a system or application are distributed and communicate in a specific deployment environment. This diagram is representative for a version 1 of the system, in later versions other nodes and components can be included.



**Entity relationship diagram**



This ER diagram represents the relationship between the database tables of the proposed system.

In the case of tables that are represented by more than one relationship, this representation is given because a table can be related to another table in different ways, including "One-to-Many" and "Many-to-Many" relationships.

For example: "Students" and "Courses". Students can attend multiple courses, which would create a "Many to Many" relationship between students and courses. However, you might also want to keep track of each student's attendance in each course, which would create a "One to Many" relationship between students and courses (a student can have many attendance records, one for each course).

In this case, we have a "Students" table, a "Courses" table, and a third "Students\_Courses" table that would relate students to courses and could contain additional information such as date of attendance and other relevant data.

The same happens with the relationship between the Student and Teacher tables that a new table arises from the relationship, which could be used for subsequent system functionalities. Up to this point in the design, only the functionalities listed above are being contemplated, the database design will be prepared for these new functionalities, if they are required in the future.