Student Management System

Use Case Document

1. Use case specification.

1.1. Applying for admission.
1.2. User login
1.3. Managing admissions applications
1.4. Registering/Updating/Removing students/faculty members
1.5. Downloading course documents
1.6. Uploading course documents
1.7. Submitting assignments
1.8. Adding/Issuing assignments
1.9. Checking grades/Giving tests
1.10. Adding tests.
1.11. Checking attendance
1.12. Updating attendance
1.13. Checking timetable.
1.14. Paying fees.

Use case specification.

1. Applying for admission.

1.1. Description: This use case describes a prospective student applying to the university for admission.

1.2. Actor(s): Prospective Student

1.3. Basic Flow:

1) The user navigates to the homepage of the university.
2) The user clicks on the "Admissions" button in the navigation bar.
3) The admissions form opens up with the appropriate fields.
4) All fields are mandatory.
5) The user updates the fields and on clicking the "Apply Now" button the details are saved in the database.

1.4. Alternate Flow 1:

1) The user skips some fields or does not upload required documents in the right format.
2) The user clicks on the "Apply Now" button.
3) The user is prompted to fill all the details or upload all the required documents. No details are saved in the database.

1.5. Alternate Flow 2:

1) The user fills all fields and uploads all required documents in the right format.
2) The user clicks on the "Apply Now" button.
3) An error occurs. Error may be related to the user's connection to the host server or database.
4) The details will not be saved in the database. The user is shown the message, "An error has occurred, please try again!".

1.6. Pre-condition: None.

1.7. Post-condition: On successful updation, the user is notified, "Form submitted! You will be mailed if you are selected into the program.". The application is submitted.

2. User login.

2.1. Description: This use case describes the user login process.

2.2. Actor(s): Administrator, Students, Faculty members.

2.3. Basic Flow:

1) The user navigates to the homepage of the university.
2) The user clicks on the “Login” button in the navigation bar.
3) The user enters his username and password and clicks the “Login” button to login.

2.4. Alternative Flow 1:

1) The user enters incorrect username or password.
2) On clicking the “Login” button, the message will be displayed “Invalid username or password, please try again!”. The user will have to again enter the correct details and login.

2.5. Alternative Flow 3:

1) The user enters his username and password and clicks the “Login” button to login.
2) The network or database connection fails at this point.
3) A message is displayed, “An error occurred, please try again!”. The user will have to try to login again.

2.6. Alternative Flow 2:

1) The user forgets his password. They click on the “Forgot Password?” link.
2) They are asked to enter their recovery email address.
3) If the recovery email addresses match, they are asked to enter a new password, and also confirm it.
4) The password is changed and the database is updated.

2.7. Alternative Flow 3:

1) The “new password” and “confirm password” fields do not match.
2) The user is prompted to re-enter both.

2.8. Alternate Flow 4:

1) On clicking the “Change Password” button, a network or database error occurs.
2) A message is displayed, “An error occurred, please try again!”. The user will have to try changing the password again.

2.9. **Pre-condition:** None.

2.10. **Post-condition:** On successful login, the user can access their dashboard to perform various functions.

3. Managing admission applications.

3.1. **Description:** This use case describes the administrator managing admission applications.

3.2. **Actor(s):** Administrator

3.3. **Basic Flow:**

1) In the administrators dashboard, they click on the “Admissions” tab.
2) They can review and select the applications for further processing.
3) The selected applications will be saved in the database.

3.4. **Alternative Flow 1:**

1) On clicking the “Save list” button, a network or database error occurs.
2) A message is displayed, “An error occurred, please try again!”. The administrator will have to review and save again.

3.5. **Pre-condition:** The administrator must be logged in.

3.6. **Post-condition:** The selected students will get an admission confirmation mail and remaining students will get a mail to wait for the next round.

4. Registering/Updating/Removing students/faculty members.

4.1. **Description:** This use case describes the administrator creating accounts for new students/faculty members.

4.2. **Actor(s):** Administrator

4.3. **Basic Flow:**

1) In the administrators dashboard, they click the “Add/Update/Remove Student Batch” or “Add/Update/Remove Faculty Member” link.
2) The administrator fills the relevant details and chooses the desired option.

3) The administrator clicks the “Add/Update/Remove” button to complete the process.
4) The administrator logs out.

4.4. Alternative Flow 1:

1) While adding/removing , the administrator skips some fields but clicks the “Add/Update/Remove” button.
2) The administrator is prompted to fill all the details. No changes are made to the database.

4.5. Alternative Flow 2:

1) On clicking the “Add/Update/Remove” button, the network or database connection fails.
2) The administrator will have to login and perform all the steps again.

4.6. Pre-condition: The administrator is registered and logged in to the system.

4.7. Post-condition: The new student batch/faculty member is successfully added to the database.

5. Downloading course documents.

5.1. Description: This use case describes a student downloading course documents.

5.2. Actor(s): Student

5.3. Basic Flow:

In the students dashboard, they click the “Course Documents” link.

They choose the desired course. The list of available documents appears.
They click the document link to start the download.

5.4. Alternate Flow 1:

A connection error occurs while retrieving the document.
The user will have to re-download the file.

5.5. Pre-condition: The student must be logged in.

5.6. **Post-condition:** None.

6. Uploading course documents.

6.1. **Description:** This use case describes a faculty member uploading course documents.

6.2. **Actor(s):** Faculty members

6.3. **Basic Flow:**

1) In the faculty dashboard, they click the “Course Documents” link.
2) They choose the desired course and click on "Upload".
3) They click the “Upload” button to start uploading.

6.4. **Alternate Flow 1:**

1) The upload fails either because its size was bigger than the allowed limit or due to a connection error.
2) The user will have to re upload the document.

6.5. **Alternative Flow 2:**

1) The faculty tries to upload a document having format not supported by the system.
2) It will prompt the message "File not supported, try again.”.

6.6. **Pre-condition:** The faculty member must be logged in.

6.7. **Post-condition:** A message, “Upload complete!”.

7. Delete course documents

7.1. **Description:** This use case describes a faculty member deleting course documents.

7.2. **Actor(s):** Faculty members

7.3. **Basic Flow:**

1) In the faculty dashboard, they click the “Course Documents” link.
2) They choose the desired course and click on "Delete".
3) The user can then checklist the documents in that course which have to be deleted.

4) Click on "Delete". The system will ask to confirm again the deletion and click "Yes".

7.4. Alternate Flow 1:

1) The faculty clicks on delete without checklisting any documents.
2) Message will prompt "No files selected, try again."

7.5. Alternative Flow 2:

1) The deletion fails due to connection error.

7.6. Alternative Flow 3:

1) The user clicks "No" to confirm delete.
2) No deletion will take place and the user can try again**.**

7.7. Pre-condition: The faculty member must be logged in.

7.8. Post-condition: A message, “Deletion complete!”.

8. Submitting assignments.

8.1. Description: This use case describes a student submitting an assignment.

8.2. Actor(s): Student

8.3. Basic Flow:

1) In the students dashboard, they click the “Submissions” link.
2) They choose the desired course. The list of available assignments appears.
3) They click on the desired assignment, choose a file to upload.
4) Click the “Submit” button to submit the assignment.

8.4. Alternate Flow 1:

1) The upload fails either because its size was bigger than the allowed limit or due to a connection error.
2) If the faculty rejects the submission.
3) The user will have to re upload the document.

8.5. Pre-condition: The student must be logged in.

8.6. Post-condition: A message, “Assignment added!”.

9. Adding/Issuing assignments.

9.1. **Description:** This use case describes faculty adding an assignment.

9.2. **Actor(s):** Faculty

9.3. **Basic Flow:**

1) In the faculty dashboard, faculty goes to the "Select Course" section.
2) Click "Add Assignment " to give an assignment for that course.
3) Upload and name the Assignment and add the due date for it. There will also be a text box for instructions, if any.
4) Finally checklist the batches who will be given the assignment and "Submit".
5) Later, faculty can check the assignment and give marks or reject it.

9.4. **Alternate Flow 1:**

1) Faculty can rename the Assignment name
2) Faculty can add/remove batches from the batch list.
3) Faculty can extend the deadline of the Assignment.

9.5. **Alternate Flow 2:**

1) The upload fails either because its size was bigger than the allowed limit or due to a connection error.
2) The user will have to re upload the document.

9.6. **Pre-condition:** The faculty member must be logged in.

9.7. **Post-condition:** A message, "Assignment added!".

10. Checking Grades.

10.1. **Description:** This use case is for Students to view their results and faculties to view any student's results.

10.2. **Actor(s):** Student, faculty

10.3. **Basic Flow:**

1) The user clicks on the Result tab.
2) Then asked to enter Roll no,Student can enter his/her roll no only.
3) Semester wise CPI/SPI will be shown.

4) Clicking on the semester number will further show course wise marks and pointer.
5) Further clicking on the course user can see weightage and marks obtained of each component.

10.4. Alternate Flow 1:

1) Failure to retrieve data due to a connection error. The user will have to start again.

10.5. Alternate Flow 2:

1) Students enter the wrong roll number and will be asked to enter again.

10.6. Pre-condition: The faculty member must be logged in.

10.7. Post-condition: None.

11. Attempting a Quiz.

11.1. Description: This use case is for students and faculty to attempt the quiz.

11.2. Actor(s): Student, Faculty

11.3. Basic Flow:

1) In the dashboard, “Select Course”
2) Find the Quiz name and click on it.
3) Wait for the start time to attempt the quiz.
4) Click “Attempt Quiz” and now you can see the questions.A timer will start for the given time duration.
5) Once you finish the quiz Click ‘Finish Attempt and Submit”.
6) You can view your marks once the test is completed.
7) And review your answers if available.

11.4. Alternate Flow 1:

1) If a user misses any answer,he will be asked to attempt it before submitting.
2) The quiz will end once the timer goes off even if some questions are left to attempt.

11.5. Alternate Flow 2:

1) Due to connection error,the quiz might interrupt and resume once the connection is established again.

11.6. Pre-condition: The user must be logged in.

11.7. Post-condition: A message, “Quiz Finished!”.

12. Adding Quiz.

12.1. Description: This use case describes a faculty member adding a Quiz.

12.2. Actor(s): Faculty

12.3. Basic Flow:

1) In the faculty dashboard, faculty goes to the “Select Course ” section.
2) Click “Add quiz ” to add a Quiz for that course.
3) Add name and date for the Quiz.Add time duration and start and end time for the Quiz.
4) Now enter the questions and options one by one.Click “+” to create more questions and “X” to remove a question.
5) Now click on submit.
6) The students enrolled in that course will receive the Quiz.

12.4. Alternate Flow 1:

1) Faculty forgets to enter some options or empty questions.
2) The time slot of the quiz is already taken by any other event.

12.5. Alternate Flow 2:

1) A connection error occurs. The user will have to start again.

12.6. Pre-condition: The faculty member must be logged in.

12.7. Post-condition: A message, “Quiz added!”.

13. Checking Attendance.

13.1. Description: This use case describes a student his/her attendance.

13.2. Actor(s): Student

13.3. Basic Flow:

1) In the student dashboard, click “attendance”.
2) Users can select the current semester or previous semesters.
3) Students can now see individual attendance of each lecture/lab.

4) Further clicking on it, Students can view date wise present/Absent.

13.4. Alternate Flow 1:

1) The student selects the semester number more than current semester.

13.5. Alternate Flow 2:

1) The student tries to access the attendance which is not updated yet.

13.6. Pre-condition: The student must be logged in.

13.7. Post-condition: None.

14. Updating Attendance.

14.1. Description: This use case describes a faculty member adding Attendance.

14.2. Actor(s): Faculty

14.3. Basic Flow:

1) The Faculty selects the “Attendance” link from the Faculty Dashboard.
2) Select “Date”, “Course” and type of activity (Lecture/Lab) .
3) Now the list of students will display who have opted that course.
4) Select Radio button “Present” or “Absent” against each student (Default all radio buttons will be selected as “Present”).
5) Click update attendance.

14.4. Alternate Flow 1:

1) Faculty selects the wrong date, for instance Saturday or Sunday.

14.5. Alternate Flow 2:

1) The update fails due to a connection error. The user will have to start again.

14.6. Pre-condition: The faculty member must be logged in.

14.7. Post-condition: A message, “Attendance Updated!”.

15. Checking Timetable.

15.1. Description: This use case describes a user's timetable.

15.2. Actor(s): Students, Faculty members

15.3. Basic Flow:

1) In the dashboard, Users click the "Timetable" link.
2) On the timetable ,User can just view or click on the subject,User will be redirected to that meeting link on click.

15.4. Pre-condition: The user must be logged in.

15.5. Post-condition: None.

16. Paying fees

16.1. Description: This use case describes a student paying the fees.

16.2. Actor(s): Student,Bank

16.3. Basic Flow:

1) Students click on the "Pay Fees" tab.
2) The system will show pending fees and asks to click on "Pay Now"
3) The student clicks on "Pay Now" and will be asked to enter debit/credit/internet banking information.
4) Then student will be redirected to Bank application
5) After successful payment on Banking application it will show "Fees Paid"

16.4. Alternate Flow 1:

1) Bank server not available.
2) Internet Connection is unstable.

16.5. Alternate Flow 2:

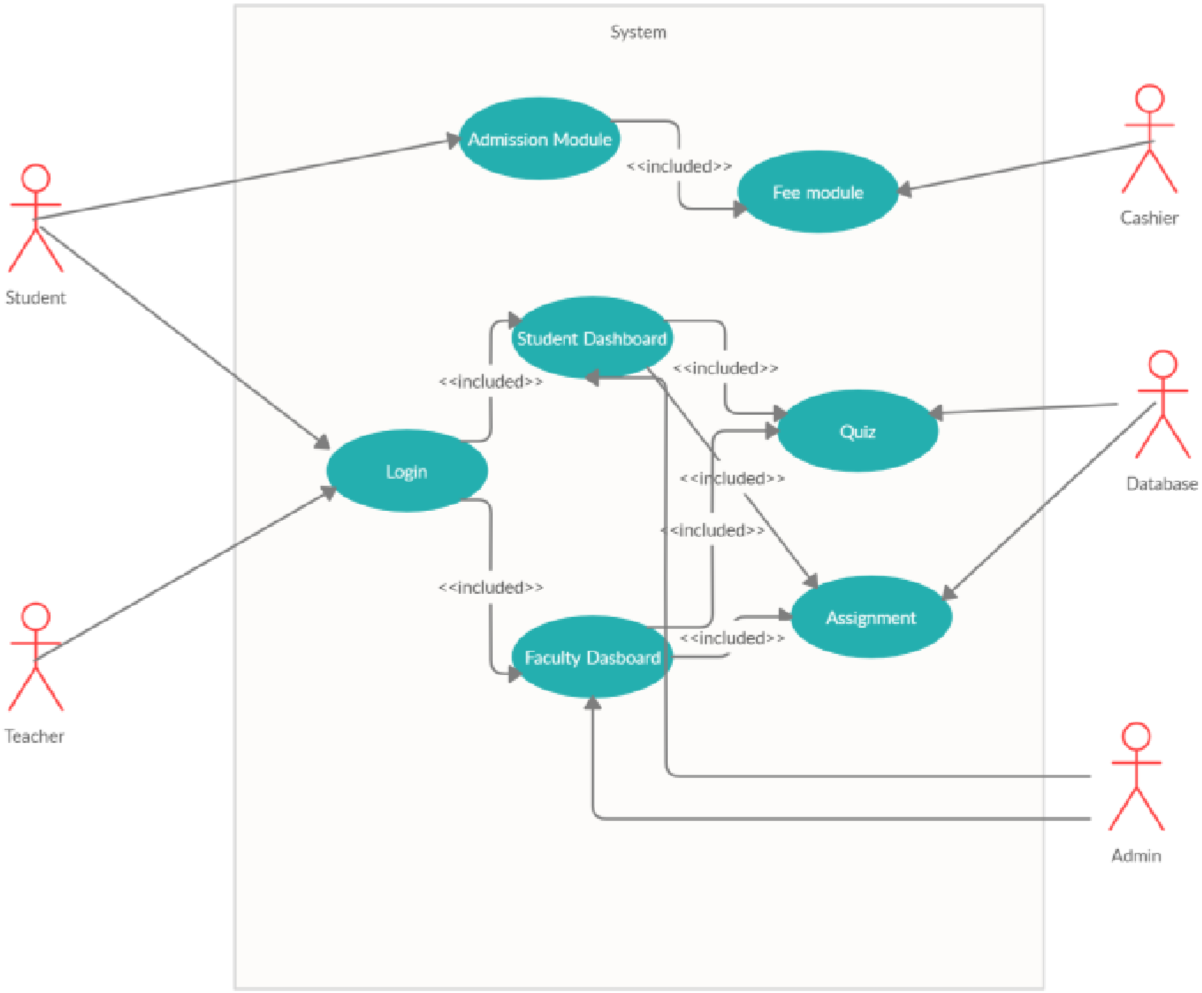
1) Fees already paid so no payment allowed

16.6. Pre-condition: The faculty member must be logged in.

16.7. Post-condition: A message, "Fees Paid".

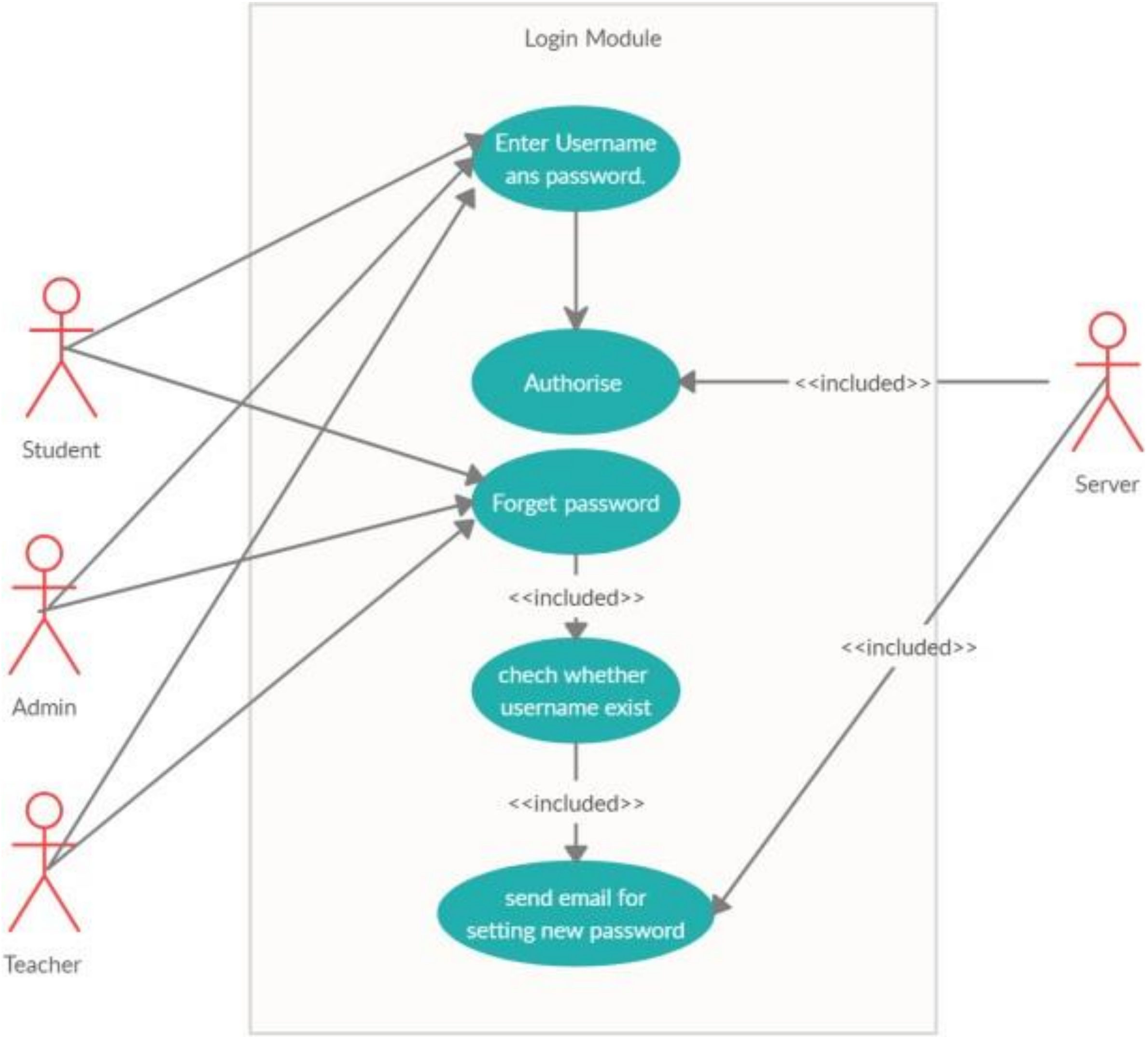
Use case diagrams

Main Usecase Diagram



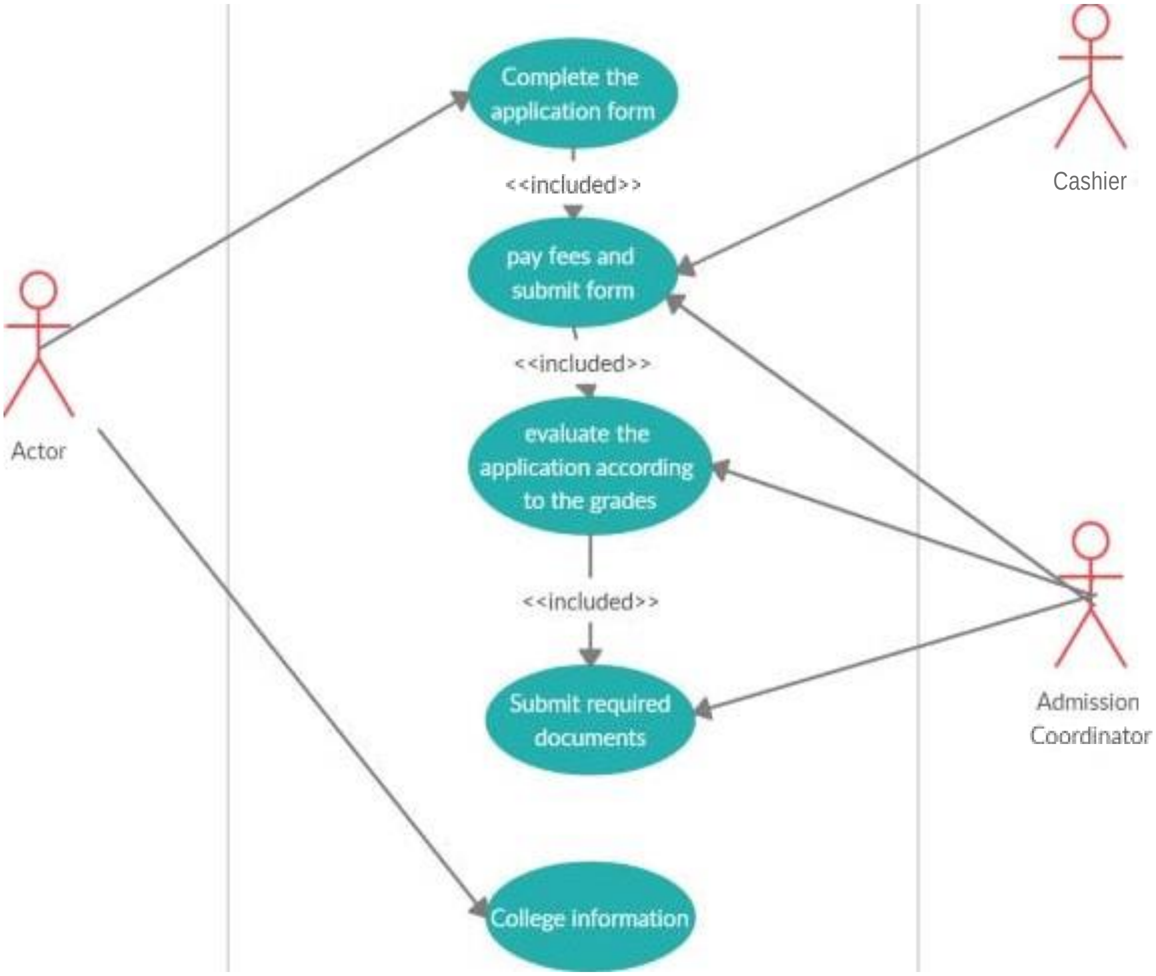
Module wise Sub Use Case diagram

1) Login Module

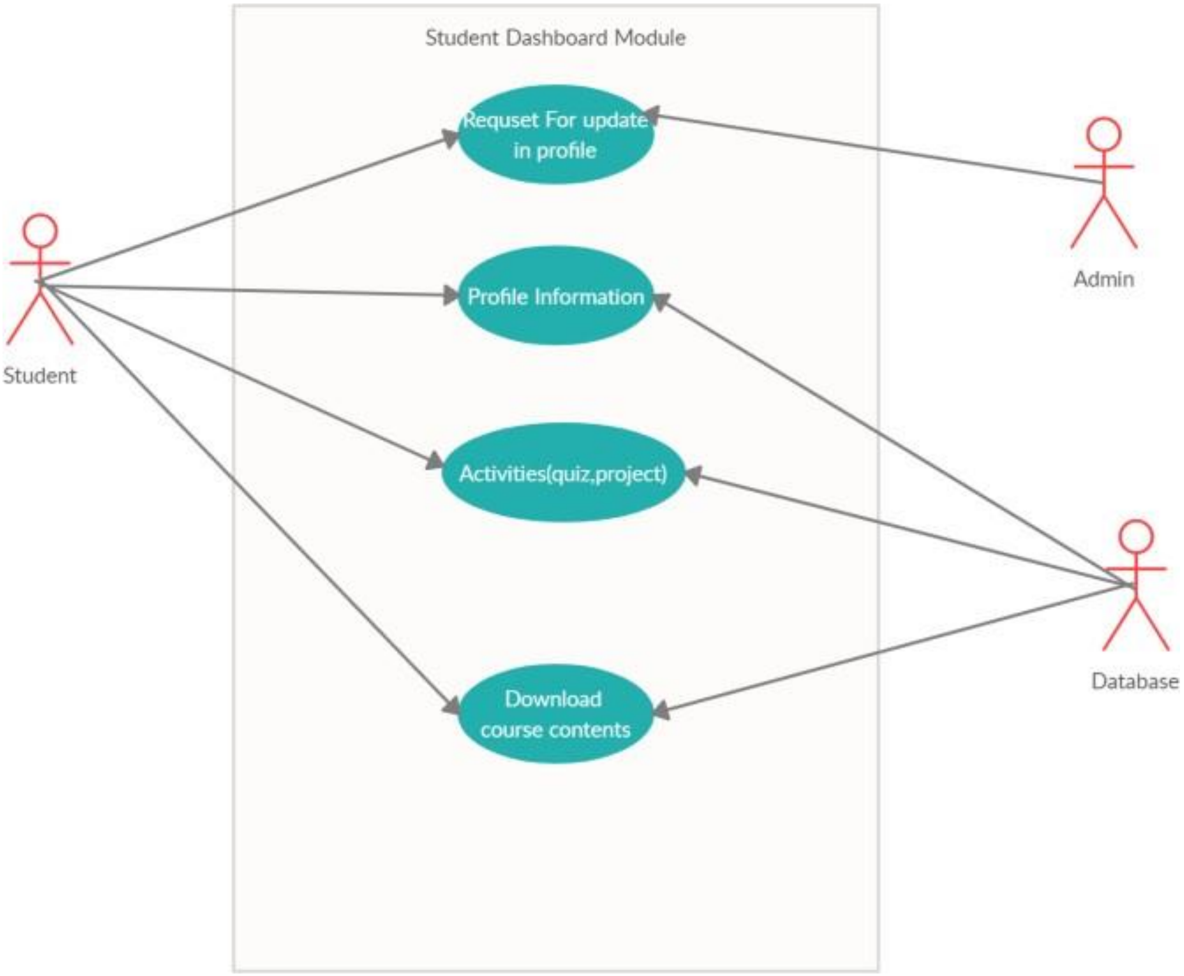


2) Admission Module

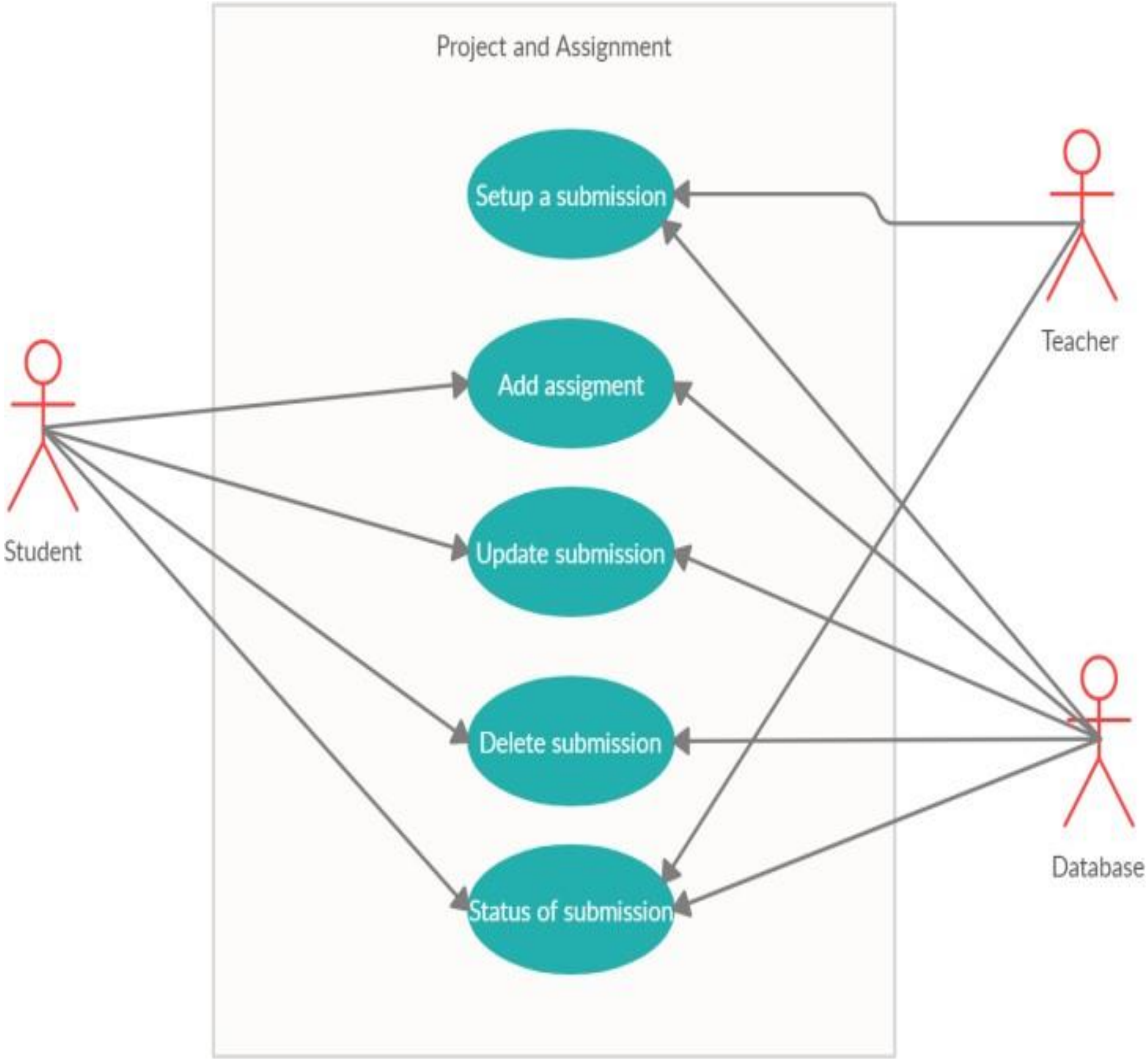
Adm‹ss›on module



3) Student Dashboard Module

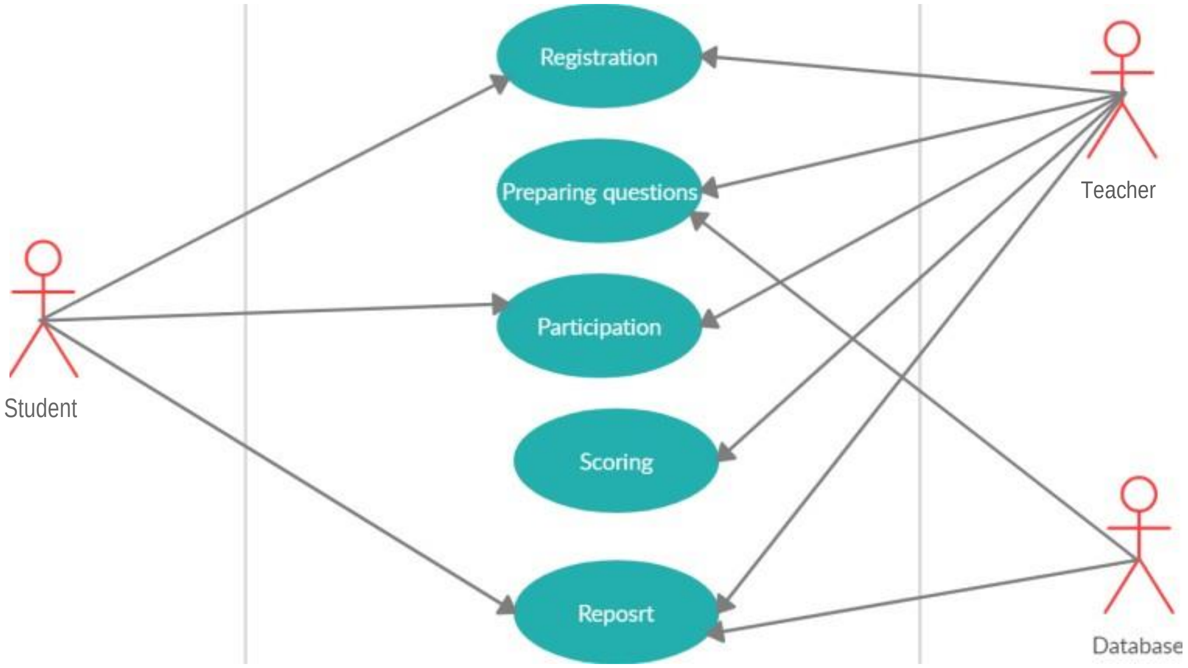


4) Assignment Module



5) Quiz Module

Qufz Module


Registration
Preparing questions
Participation
Scoring
Reposrt
Student
Teacher
Database

Student Management System

Scenarios

1. **Applying for admission**

   The user is a prospective student. They want to apply to the university. They search for the university on his search engine and end up on the university's home page. They click on the "Admissions" link on the navigation bar, and are presented with an admission form. They fill up the from, upload the required documents and click "Apply Now". The user has submitted the application.

2. **User login**
   The user is an administrator/student/faculty member. They want to perform a related function. They search for the university on his search engine and end up on the university's home page. They click on the "Login" link on the navigation bar, enter their credentials and login. They perform the function they wanted.

3. **Managing admission applications**

   The user is an administrator. They want to manage the admission applications. They navigate to the university's home page and login. On their dashboard, they choose the "Admissions" option.

4. **Registering/Updating/Removing students/faculty members**

The user is an administrator. They navigate to the university's home page and login. On their dashboard, they choose the "Add/Update/Remove Student Batch/Faculty Member" option depending on the scenario. They fill the relevant details and perform the function.

5. **Downloading course documents**

   The user is a student. They navigate to the university's home page and login. On their dashboard, they choose the "Course Documents" option. They select the desired subject and a list of available documents appears. They click on the desired document, which starts the download.

6. **Uploading course documents**

   The user is a faculty member. They navigate to the university's home page and login. On their dashboard, they choose the "Course Documents" option followed by the "Upload" option. They select the desired subject, choose a file and click the "Upload" button. The document is uploaded and available for download by the student.

7. **Delete course documents**

   The user is a faculty member. They navigate to the university's home page and login. On their dashboard, they choose the "Course Documents" option followed by the "Upload" option. They select the desired subject, choose a file and click the "Upload" button. The document is uploaded and available for download by the student.

8. **Submitting assignments**

   The user is a student. They navigate to the university's home page andlogin. On their dashboard, they choose the "Submissions" option followed by the course they want to submit an assignment for. A list of available assignments appears. The user can choose the desired assignment, upload the submission and hit the "Submit" button to complete the assignment submission successfully.

9. **Adding/Issuing assignments**

The user is a faculty member. They navigate to the university’s home page and login. On their dashboard, they choose the “Submissions” option followed by the course. Then they click on the “Add” button to add a new assignment. They fill the relevant details, and hit “Issue” to issue the assignment. The student can now submit the assignment.

10.Checking grades

The user is a student. They navigate to the university’s home page and login. On their dashboard, they choose the “Grades” option. They choose the semester they wish to see the marks for, and the mark sheet appears.

11.Attempting tests

The user is a student. They navigate to the university’s home page and login. On their dashboard, they choose the “Tests” option. The student can see in the examination window. Students can see the upcoming exam name,Total questions, positive marks , negative marks, and time limit for the exam. And click the “start” button to attempt the test. Then students can show the questions and 4 options of each question according to the selection of options score will add to each right answer. Then finally displays the result that includes total questions, right answers, wrong answers and score. Another feature is that students can see its test history, and its ranking of each test.

12.Adding tests

The user is a faculty member. They navigate to the university’s home page and login.Then going into the “examination” section and then select dropdown manu named “quiz”. Choose options like “add exam” and “remove exam”. In the “add exam” Faculty add quiz details like title, no. of questions, marks on right answers, marks on wrong answer (negative marking), and description about quiz. After “submit”. In the first bar faculty add questions, and remaining 4 box faculty add 4 options and last one in the dropdown menu faculty select right and then press “Next”. To add another question. Another feature is that faculty can see its test history, and student ranking of that test.

13. Checking attendance

The user is a student. They navigate to the university’s home page and login. On their dashboard, they choose the “Attendance” option. They choose the semester they wish to see the attendance for, and the attendance report appears.

14. Updating attendance

The user is a faculty member. They navigate to the university’s home page and login. Then faculty click into “Add attendance” then faculty select “Programme”, “Batch", “Subjects", by clicking the list of the student will be displays and each student name left side two options will be available “Present”, “Absent” and other information included like Total lecture, Total attended lecture, and then %attendance is display. If faculty select present or absent according to changes will occur in that students bars. And this bar will also display on that student dashboard attendance details.

15. Checking timetable

The user is a student or a faculty member. They navigate to the university’s home page and login. On their dashboard, they choose the “Course Documents” option. Their timetable is visible on screen.

16. Paying fees

The user is a student and fee manager. This process of paying fee will be offline when a student pays a fee according to its programme the fee manager updates its fee information. There are two parts on the fee manager side paid and unpaid students. If a student pays full fee then this student will be listed as a paid student and if the student did not pay full fee then it was listed as an unpaid fee. And according to the fee collection fee manager going to the student information list by “Find Student” according to the students unique ID and then clicking “update fee” lists and then checking its paying fee history and then clicking “Add amount” and the student's personal fee collection will be updated.