# IDEAS:

1. **Student profile pages**: Each student should have a profile page that displays their personal information, such as their name, student ID, class, date of birth, and status (e.g. full-time or part-time).
2. **Enrollment system**: The website should allow students to browse and enroll in available courses. This could include a search function to help students find courses based on subject, professor, schedule, etc.
3. **Grade tracking**: The website should allow students to view their grades and academic progress. This could include a transcript or grades overview, as well as detailed information about each course and assignment.
4. **Attendance tracking**: The website should allow students to view their attendance records for each class. This could include the number of absences and tardies, as well as any excused absences that have been approved by the instructor.
5. **Payment system**: If the university charges tuition or other fees, the website should include a system for students to make payments and view their financial account balance.
6. **Student resources**: The website could include links to helpful resources for students, such as tutoring services, academic advisors, and campus support services.
7. **News and events**: The website could include a section for news and events related to the university and its students, such as campus-wide announcements, guest lectures, and social events.

# REQUIREMENTS: (sites only for students)

1. A database to store student information, including personal details such as **name, contact information, and enrollment details**.
2. A login system for students, faculty, and administrators to access the website and perform various actions, such as enrolling in **courses, viewing grades, and updating personal information.**
3. A course catalog and enrollment system, allowing students to **browse available courses** and **register** for them (including start date and end date of the course in each semester)
4. A grade management system, allowing faculty to input and **view grades** for students, and allowing students to view their own grades or credit or a course and **how many credits they have to obtained in each semester and in total of a course** to be promoted for next year or to graduate.
5. **A notification system**, allowing the website to **send alerts and updates to students** a timetable for the courses enrolled and faculty about important events.
6. **An interface for administrators** to manage the website and perform tasks such as adding new courses, managing user accounts, and generating reports.
7. A system for **managing student transcripts and other academic records**.
8. A system for **managing course schedules and class schedules**, including the ability to search for available classes and view class locations.
9. Responsive design, ensuring that the website is accessible and easy to use on a variety of devices, including desktop computers, tablets, and smartphones.

# Assessment:

* Start a simple website with Django: <https://docs.djangoproject.com/en/4.1/intro/tutorial01/>
* Setup a SQL/MySQL and have a management tool to manage data:
* Create table “User” with “Id” as primary key and columns “Username”, “Email”, “Password”, role of user (student, faculty, or administrator)
* Add some data by using query instead of input directly (Postgresql/mysql/sqlite)
* Connect the website with database
* Create a form using html with “Username” textbox, “Password” textbox, “Login” button
* Create an endpoint function with params “username” and “password”
* Get user information from database by matching “username”
* Add some logic to verify user:
  1. If user not exist – return to browser “User is not exist”
  2. If user exist but wrong password – return to browser “Password Is incorrect”
  3. If user is valid – navigate to other page (Welcome page) and show “Welcome <name>”
* Add “Logout” button in Welcome page. If user click on it, navigate to “Login” page
* Using browser/server session
  1. If user doesn’t log in, they can’t enter to Welcome page by copy-paste URL and must be navigate back to Login page (check if session is existed)
  2. If user logged in, they will automatically enter Welcome page even they input Login page
  3. If they click on “Logout” button, the session is removed, and they will be navigate to Login page
  4. Considerate prob: Configurate timeout of session so after a while, they will be kick out to Login page again
* Add button “Register/Sign up” and “Sign in” to Login page to handle the log in requst.
* Add action to show “Username”, “Name” and “Password” with “Register button.
  1. If any field is empty – return to browser “All input need to be completed”
  2. If “Username” is exist in database – return to browser “User is exist”
  3. If “Username” isn’t exist. Create session and auto login user to Welcome page. User can use password to login again.
* Add button “Reset password” to Welcome page
* Add action so the “Current password” textbox, “New password” textbox and “Save” button will show up. Also the “Reset password” button is hidden. (1)
* Add “username” into session and try to get it
* Add endpoint to receive “Current Password”, “New Password”, “username” and verify it
  1. If “ Username” and “Current Password” is correct, reverse step 1 and user can login again with new password
  2. If “ Current Password” is wrong – return to browser “Password is incorrect”
* Create a logout view function to handle the logout request. This function need to be done by using the authentication system to log out and redirect to the main page or login page (or both).
* For a course catalog and enrollment system:
  1. Define a model to represent course in the database (course code, name of course, description, credits, and the requirement to apply for this course).
  2. Create new function in Views to display the course catalog. This function should query the database for a list of available courses. (There are many requirement to fulfill the condition to enroll for the course such as years of studying, has to be passed the related courses, etc)
  3. Create a view function to handle the enrollment request as a validate form, update the database to reflect + display the enrollment with confirmation message.
  4. A function to add, edit or delete course.
* A grade management system:
  1. Define a model to represent grades in database with foreign key. This model should include fields for student name, student ID, course, and a grade.
  2. Create a view function to input grades by displaying a form for faculty member to select the student, course, and grade + submit button to update the info.
  3. Create a fuction to query the database thatcontains the student’s grades and display them to the user who send the request.
* Notification system:
  1. Define a model that includes fields of message, recipient and relevant information (such as date and time, name of event) that the notification system should send.
  2. Create a view function for administrator/Faculty who has the authority to enter the message and recipient.
* An interface for administrator:
  1. Create a view function for administrators to access the administrative interface. This need to be sure that only the administrators can log in.
  2. Create view function for tasks that only the administrator can perform such as managing account, authority setting.
* A system to manage student transcripts:
  1. Define a model to represent transcript in database. This model should include fields for the students, courses, grades. (considerate about semester, year of graduate student)
  2. Create a view function for student to view their own transcripts by querying the database for the students transcripts and display to the user that sent request.
* A system to manage course schedule:
  1. Define a model to represent course schedules in the database including fields of name of course, professor/instructor, location, code of the course, start and end time of the course, day of the week (normally they don’t give the schedule of the week after due to some misinfomation about the future) and lastly, the message fields to give extra info about the course if needed.
  2. Create a view fuction for student to search for the available course including some button or fields to enter the criteria to get the information of the course.