Project Goal:

To create a web-based and cloud-integrated platform where students can access syllabus, notes, previous exam papers, and other study material in a structured way. The portal will initially be developed for the Computer Science (CS) department of your college, covering the following courses:

1. B.Sc. (CS), 2. B.C.A. (Science), 3. M.Sc. (CS), 4. M.Sc. (CA)

Key Features:

* Structured Access to Study Materials: Organized by course, subject, year, and semester.
* Search Bar:
  + Quick access to materials.
  + Users should be able to search for any subject and get matching results.
  + Partial search feature (e.g., "cyb" shows "Cyber Security").
  + Displays a list of matching subjects with:
    - * 📚 Subject Name
      * 🎓 Course Name
      * 📅 Year & Semester
  + Clicking a search result should take users to the Study Material for that subject.
* Login/Signup: Separate login options for students and admins (teachers).
* Student Features:
  + Access to study materials.
  + View PDF before downloading.
  + Save materials in their account.
  + Personal storage space for saving their own notes and materials.
* Admin (Teacher) Features:
  + All student features (viewing, saving, downloading, storing).
  + Ability to add materials to respective courses.
* Cloud-Based Infrastructure:
  + Database: MongoDB Atlas (cloud-based NoSQL database).1
  + Cloud Storage: To be decided between Firebase Storage and AWS S3 for storing study materials.

Technology Stack:

1. Frontend: React with Vite
   * React for building the dynamic user interface with a component-based architecture.
   * Vite as a fast and efficient build tool for a streamlined development experience.2
2. Backend: Python with Flask
   * Python as the backend language for building the API.
   * Flask as a lightweight and flexible microframework for creating the API endpoints.3
3. Database: MongoDB Atlas
   * A cloud-based NoSQL database for storing application data (users, courses, materials, etc.).
4. Cloud Storage:

To be decided, with a slight inclination towards Firebase Storage for easier initial integration with the frontend and a generous free tier for a college project. However, AWS S3 is also a viable option offering scalability and industry-standard features.4

Development Steps (General Outline):

1. Detailed Requirements Gathering: Further refine user stories and content structure.
2. UI/UX Design: Create wireframes and mockups of key pages.
3. Backend Setup (Python/Flask):
   * Set up the Flask project structure.
   * Define API endpoints for user authentication, fetching materials, search functionality, and admin material upload.
   * Integrate with MongoDB Atlas.
4. Frontend Setup (React/Vite):
   * Create the React project using Vite.
   * Develop UI components for login, material listing, search, personal space, and admin dashboard.
   * Implement routing and state management.
   * Integrate with the backend API endpoints.
5. Cloud Storage Integration (Firebase Storage or AWS S3):
   * Set up the chosen cloud storage service.
   * Implement backend logic for handling file uploads by admins and retrieving file URLs for users.
   * Potentially integrate frontend for direct uploads with Firebase Storage (with appropriate security rules).
6. Implement Core Features:
   * Login/Signup functionality.
   * Displaying study materials based on course and subject.
   * Search bar implementation (frontend UI and backend API logic).
   * Personal storage space for users.
   * Admin material upload functionality.
   * PDF viewing capability on the frontend.
7. Testing: Thoroughly test all features and user flows.
8. Deployment: Deploy the frontend and backend to suitable hosting platforms.