**UniConnect: Project Blueprint & Roadmap:**

1. Executive Summary

UniConnect is a full-stack MERN web application designed to be a central academic ecosystem for university students. The platform directly addresses the common challenges of fragmented study materials, difficulty in finding suitable study partners, and inefficient communication for academic queries. By providing course-specific communities, a resource-sharing hub, and integrated AI-powered study tools, UniConnect aims to foster a collaborative and efficient learning environment. This project demonstrates a full development lifecycle, from ideation and architecture to deployment, showcasing proficiency in modern web technologies and real-world problem-solving.

------------------------------------------------------------

2. Problem & Solution

The Problem:

- Scattered Resources: Students waste valuable time searching for notes, past exams, and other study materials spread across various chats and drives.

- Finding Collaboration: Identifying classmates with compatible study habits and schedules for group work is often a challenge.

- Inefficient Q&A: Academic questions posted in large, unstructured social media groups are easily missed or left unanswered.

- Lack of Community: There is no dedicated, centralized space for students of the same course to connect, discuss topics, and share knowledge.

The Solution:

UniConnect will be a one-stop platform that provides:

- Centralized Resource Hubs: Organized by course, allowing easy upload and access to materials.

- Course-Specific Communities: Dedicated forums and member lists for each course.

- Study Buddy Finder: A feature to post and find study requests based on course and topic.

- AI-Enhanced Learning: Tools to automatically summarize notes and generate flashcards, making studying more effective.

------------------------------------------------------------

3. Core Features & Functional Requirements

User Management:

- Secure user registration and login using email and password.

- JWT-Based Sessions for secure user sessions and protected API routes.

- All user passwords must be hashed using bcrypt.js.

- Users can view and edit their profile, including name, bio, profile picture, and enrolled courses.

Course & Community:

- Users can browse and enroll in/unenroll from available courses.

- Upon selecting a course, users are taken to a dedicated dashboard for that course.

Resource Sharing:

- Users can upload files (PDFs, DOCX, etc.) to specific course hubs.

- Users can share external links to relevant resources.

- All resources for a course are displayed in an organized, chronological feed.

Discussion Forums:

- Users can create new discussion posts within a course community.

- Users can comment on existing posts to create threaded conversations.

- Users can only delete their own posts and comments.

Study Buddy Finder:

- Users can post a "study buddy needed" request for a specific course or topic.

- Users can view and filter active requests from other students.

- A simple "I'm interested" button to notify the post creator.

AI-Powered Study Tools:

- Note Summarizer: Automatically generates a concise summary of a document's content.

- Flashcard Generator: Automatically extracts key terms to create digital flashcards.

------------------------------------------------------------

4. Tech Stack & Architecture

Frontend: React.js, React Router DOM, Redux Toolkit, Axios, Material-UI (MUI).

Backend: Node.js, Express.js.

Database: MongoDB with Mongoose (hosted on MongoDB Atlas).

Authentication: JSON Web Tokens (JWT), bcrypt.js.

File Storage: Cloudinary (for file/image hosting) with Multer (for server-side handling).

AI Integration: OpenAI API or Google Gemini API for summarization and flashcard generation, using pdf-parse for text extraction.

Deployment: Vercel (Frontend), Render (Backend), MongoDB Atlas (Database).

------------------------------------------------------------

5. Detailed 8-Week Development Roadmap

Phase 1: Backend Foundation (Weeks 1-2)

Week 1: Planning & Setup

- Day 1-2: Finalize & Architect

- Solidify all features.

- Sketch basic UI wireframes.

- Design MongoDB schemas:

User: { name, email, password, profilePicture, bio, enrolledCourses: [CourseRef] }

Course: { name, code, description }

Resource: { title, fileUrl, uploadedBy: UserRef, course: CourseRef }

Post: { content, author: UserRef, course: CourseRef, comments: [CommentSchema] }

Comment: { content, author: UserRef }

- Day 3-5: Backend Project Setup

- Initialize Node.js project.

- Install dependencies: express, mongoose, dotenv, cors.

- Establish connection to MongoDB Atlas.

- Configure .env file.

- Day 6-7: API Router & Model Setup

- Create folder structure (models, routes, controllers, middleware).

- Implement Mongoose schemas and set up API route files.

Week 2: Core Backend APIs (Authentication & User Management)

- Day 1-3: User Authentication APIs

- Implement POST /api/auth/register with validation and password hashing.

- Implement POST /api/auth/login to generate and return a JWT.

- Create JWT verification middleware.

- Day 4-5: User Profile APIs

- Implement GET /api/users/me (protected).

- Implement PUT /api/users/me (protected).

- Day 6-7: Course Management APIs & Testing

- Implement GET /api/courses.

- Implement PUT /api/users/enroll/:courseId and PUT /api/users/unenroll/:courseId.

- Test all endpoints with Postman or Insomnia.

Phase 2: Backend Features & Frontend Kick-off (Weeks 3-4)

Week 3: Backend - Resource & Discussion APIs

- Day 1-4: Resource Sharing APIs

- Configure multer and Cloudinary SDK.

- Implement POST /api/courses/:courseId/resources for file uploads.

- Implement GET /api/courses/:courseId/resources.

- Implement DELETE /api/resources/:resourceId.

- Day 5-7: Discussion Forum APIs

- Implement POST and GET for posts and comments.

- Implement DELETE routes for posts and comments.

Week 4: Frontend Setup & Initial Integration

- Day 1-2: Study Buddy Finder APIs

- Implement POST and GET for /api/study-buddy/requests.

- Day 3-5: React Project Setup

- Initialize React project (Vite).

- Install dependencies: axios, react-router-dom, @reduxjs/toolkit, etc.

- Set up folder structure (pages, components, features, services).

- Day 6-7: Authentication UI & State

- Build Login and Register pages.

- Create Redux slice for authentication.

- Connect forms to backend, store JWT in localStorage, and manage auth state.

Phase 3: Frontend Development (Weeks 5-6)

Week 5: Core UI/UX

- Day 1-2: Main Layout & Dashboard

- Build Navbar and Footer components.

- Create a main Dashboard page for logged-in users.

- Day 3-4: Course Listing & Enrollment

- Build a page to display and enroll in courses.

- Day 5-7: User Profile Page

- Build a page to display and edit user profile information.

Week 6: Feature Integration

- Day 1-3: Resource Sharing UI

- Build a detailed page for each course.

- Implement components to list and upload resources.

- Day 4-7: Discussion Forum UI

- Build components to display posts and comments within the course page.

- Implement forms for creating posts and adding comments.

Phase 4: AI, Polish & Deployment (Weeks 7-8)

Week 7: AI Integration & Final Polish

- Day 1-3: AI Backend APIs

- Install pdf-parse and LLM SDK.

- Create POST /api/ai/summarize.

- Create POST /api/ai/generate-flashcards.

- Day 4-5: UI/UX Enhancements & Responsiveness

- Add loading states and error handling throughout the app.

- Ensure the application is fully responsive.

- Day 6-7: AI Frontend & Testing

- Add "Summarize" and "Create Flashcards" buttons to the UI.

- Connect buttons to the new AI APIs.

- Conduct full end-to-end testing.

Week 8: Deployment & Documentation

- Day 1-2: Final Touches & Buffer

- Fix any remaining bugs or make minor improvements.

- Day 3-4: Deployment

- Deploy frontend to Vercel/Netlify.

- Deploy backend to Render/Heroku.

- Configure production environment variables.

- Day 5-6: Project Documentation

- Create a comprehensive README.md on GitHub with a project overview, setup instructions, tech stack, screenshots, and a link to the live demo.

- Day 7: Final Review & Celebrate

- Do a final walkthrough of the live application.

- Share with friends for feedback.