Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	26-05-2025
Team ID	
Project Name	LEARNHUB
Maximum Marks	4 Marks

Technical Architecture: The technical stack for a Learning Management System (LMS) like LearnHub typically includes a combination of frontend, backend, database, hosting, and other supporting technologies. Here's a modern and scalable tech stack you can use:

🔧 **Frontend (Client-Side)**

This is what users interact with.

React.js – Fast, component-based UI.

Redux / Context API – State management.

Tailwind CSS / Bootstrap / Material UI – Styling and UI components.

Vite – For faster development and build tooling (instead of CRA).

React Router – Client-side routing.

🧠 **Backend (Server-Side)**

Handles business logic, API routes, and database communication.

Node.js – JavaScript runtime environment.

Express.js – Web application framework for Node.js.

JWT – Authentication and session management.

bcrypt.js – For password hashing.

🗃️ Database

Stores user data, courses, progress, quizzes, etc.

MongoDB – NoSQL database, flexible document structure.

Mongoose – ODM (Object Data Modeling) for MongoDB in Node.js.

> Optionally, you could use PostgreSQL (SQL) if your LMS needs complex relational queries.

☁ **Cloud & Hosting**

Where the app lives and runs.

Render / Vercel / Netlify – Hosting for frontend (React).

Render / Railway / Heroku / Cyclic – Hosting backend Node.js/Express API.

MongoDB Atlas – Cloud MongoDB database.

☁ **File & Media Storage**

For course videos, images, documents.

Cloudinary / AWS S3 – Image/video storage and optimization.

💬 **Email & Notification Services**

For user communication.

Nodemailer – Sending transactional emails (e.g., registration, reset password).

Firebase Cloud Messaging (FCM) or OneSignal – Push notifications.

📊 **Admin Dashboard (Optional)**

For managing users, courses, enrollments.

React Admin or Custom Admin Panel – Built with the same frontend stack.

Chart.js / Recharts – Data visualization for stats.

✅ **Others / Dev Tools**

Git & GitHub – Version control.

Postman – API testing.

Dotenv – Manage environment variables.

ESLint / Prettier – Code quality and formatting.

🧪 **Testing (Optional)**

Jest – Unit testing for React components.

Supertest + Mocha/Chai – API endpoint testing.

Summary Diagram:

Frontend: React + Vite + Tailwind

|

API: Node.js + Express + JWT + Mongoose

|

Database: MongoDB Atlas

|

Media: Cloudinary

|

Hosting: Vercel (frontend) + Render/Railway (backend)

Let me know if you want the stack tailored for mobile apps, multilingual support, payment integration, or microservices architecture.

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web and mobile-friendly interface for patients and providers	HTML, CSS, JavaScript / React Js etc.
2.	Application Logic-1	Appointment booking, calendar management, reminders	Node.js, Express.js
3.	Application Logic-2	Admin panel, provider management, reporting	React js, Node js
4.	Database	Stores user profiles, appointments, provider datas	MongoDB

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
5.	Open-Source Frameworks	Frontend frameworks	React.js, Node.js, BootStrap, Tailwind CSS
6.	ecture	3-tier architecture with RESTful APIs	Microservices

References:

React.js Documentation

Node js Best Practice

JSON Web Server Referance

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d