COLLEGE CODE : 9623

COLLEGE NAME : Amrita College of Engineering and Technology

DEPARTMENT : Computer Science and Engineering

STUDENT NM-ID :

ROLL NO : 962323104054

DATE : 12-09-2025

Completed the project named as Phase 2

TECHNOLOGY PROJECT NAME : Dynamic Image Slider

SUBMITTED BY,

Name: Krishnaprasad. P. S

Mobile no : 7305575907

PHASE 2 – Solution & Architecture

Tech Stack Selection

1. Frontend:

HTML5 – Structure

CSS3 / Tailwind CSS – Styling & responsiveness JavaScript

(ES6+) – Slider logic & DOM manipulation

2. Framework:

React.js (for scalability & component reusability)

Or Vanilla JS (for a simple slider)

3. Animations:

GSAP (Optional) – Advanced animations

4. Backend (Optional):

Node.js + Express.js – API for dynamic image

MongoDB / MySQL – Store image data

5. Tools & Deployment:

Git + GitHub – Version control

Vite / Webpack – Build & bundling (if using React)

Netlify / Vercel – Hosting and deployment

UPI Structure / API Schema Design

- Header – Displays project name and logo
- Image container – Shows current active image dynamically
- Prev/next Buttons – Manual navigation between images

- Footer - End of the page

API Schema Design 1.

Base URL

https://api.dynamic-slider.com/

2. Endpoints

/api/images - Get all slider images

/api/images/:id - Get a specific image by ID

/api/images - Add a new image (Admin)

/api/images/:id - Update existing image details (Admin)

/api/images/:id - Delete an image (Admin)

3. Image Object Schema


```
{
  "id": 1,
  "title": "Sunset View",
  "imageUrl": "https://cdn.dynamic-slider.com/images/sunset.jpg",
  "caption": "Beautiful sunset over the fields",
  "order": 1,
  "isActive": true,
  "createdAt": "2025-09-12T10:00:00Z"
}
```


Data Handling Approach

1. Frontend (Client-Side)
   - Sends a GET /api/images request to fetch image data.

- Receives JSON response and dynamically renders slider images.
- Updates in real-time without reloading the page.

2. Backend (Server-Side)

- Handles CRUD operations: Create, Read, Update, Delete images.
- Validates input before storing.
- Returns clean JSON responses to the frontend.

3. Database Layer

- Stores metadata (title, caption, order, status, image URL).
- Example: MongoDB or MySQL.

4. Cloud Storage / CDN

- Stores actual image files (e.g., Cloudinary, Firebase).
- Provides fast, optimized delivery through CDN links.

5. Data Flow

- User → Frontend → API → Backend → Database + Cloud Storage → Frontend

Component / Module Diagram

1. App Root Module

- Acts as the entry point of the application.
- Loads global components like Header, Footer, and routing configuration.

2. Header Module

- Provides navigation and branding for the application.
- Components:

- Logo / Brand Name (Top left)
- Navigation Menu (e.g., Home, Dashboard, Admin)
- Core module where the image slider is displayed dynamically.

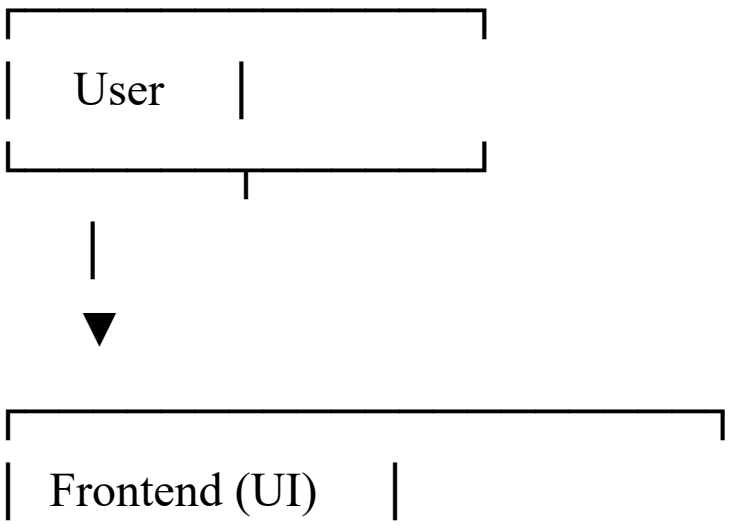
3. Main Dashboard Module

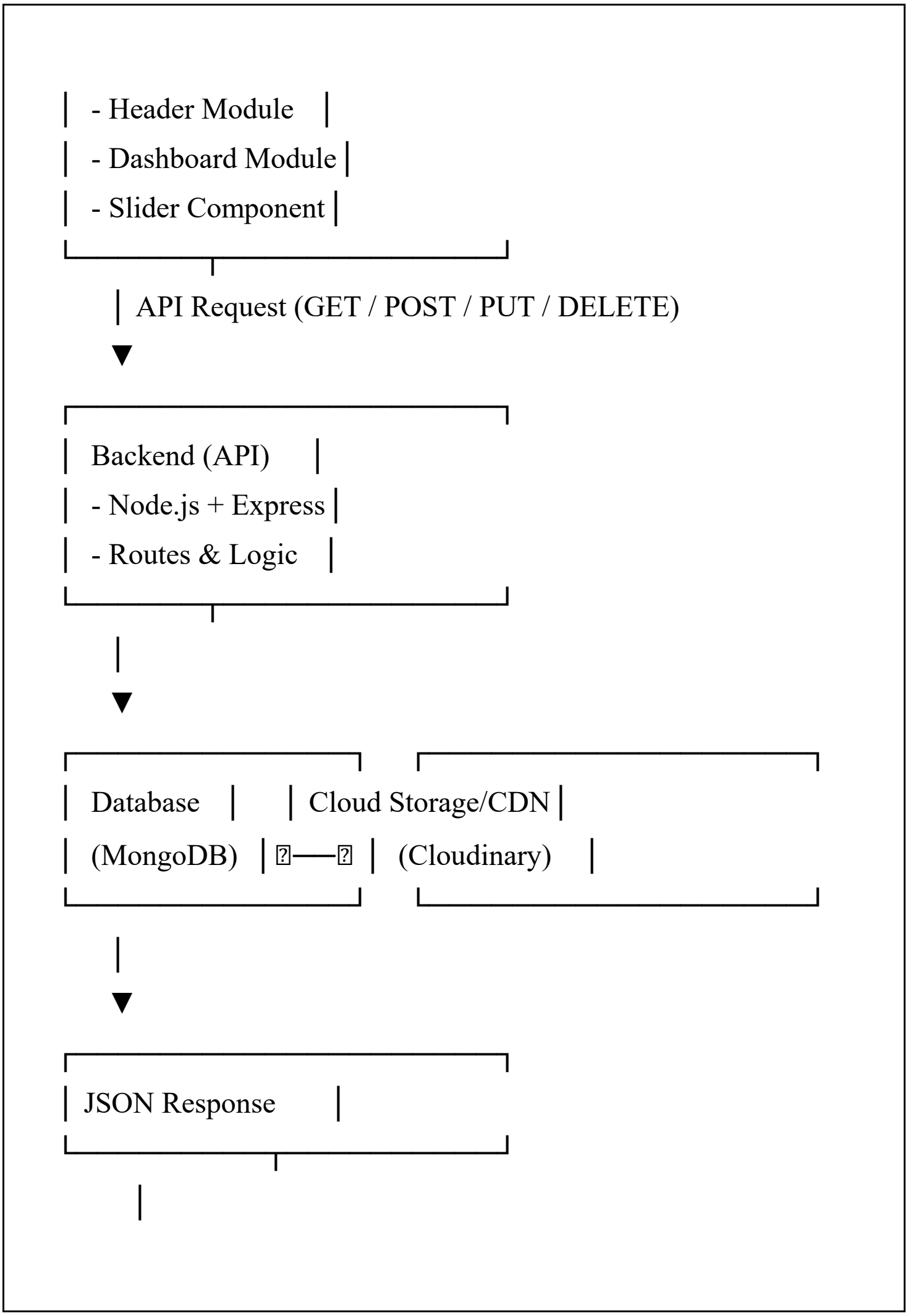
- Core module where the image slider is displayed dynamically.
- Fetches image metadata from the backend API.
- Allows admin users to add, edit, delete images (optional).
- Components:
- Slider Component: Displays dynamic images.
- Thumbnail Navigation (optional): Small preview images.
- Add/Edit Image Modal: For admin use.

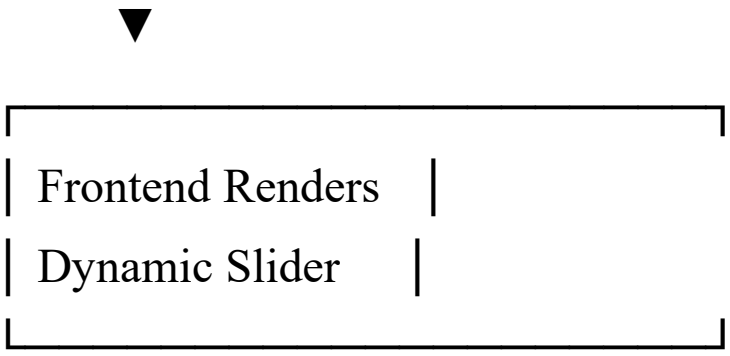
4. Data Flow Between Modules

- App Root → Header Module → Dashboard Module → Backend API
- App Root: Initializes the app and sets up routes.
- Header: Provides navigation across pages.
- Dashboard: Fetches data via API and renders slider.

Basic Flow Diagram




- Header Module
- Dashboard Module
- Slider Component
API Request (GET / POST / PUT / DELETE)
Backend (API)
- Node.js + Express
- Routes & Logic
Database
(MongoDB)
Cloud Storage/CDN
(Cloudinary)
JSON Response


Frontend Renders
Dynamic Slider