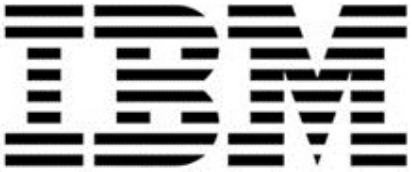


COLLEGE CODE : 9111

COLLEGE NAME : SRM Madurai College For Engineering And Technology

DEPARTMENT : B.E Computer Science and Engineering

STUDENT NM-ID : 5AD5D2FE0760609B6F048D417B3D393E

REGISTER NO : 911123104026

DATE : 15/09/2025

Completed the project named as : IBM-FE-Dynamic Image Slider

SUBMITTED BY,

NAME : Kirubananthan M

MOBILE NO : 97875 89869

Tech Stack Selection

The technology stack was carefully chosen to balance performance, scalability, and ease of development.

- **Frontend:** React.js (for building responsive and reusable UI components).
- **Backend:** Node.js with Express (to provide REST API endpoints for image management).
- **Database:** MongoDB (for storing image metadata such as URL, title, and description).
- **Styling:** CSS3 / Tailwind CSS (for responsive and modern UI design).
- **Version Control:** Git & GitHub (for code collaboration and tracking changes).
- **Testing Tools:** Postman (API testing) and Jest (unit testing).
- **Deployment:** Render / Vercel (for hosting backend and frontend separately).

UI Structure / API Schema Design

UI Structure:

- **Header Section:** Contains project title/logo.
- **Slider Component:**
    - Displays current image.
    - Navigation arrows (left/right).
    - Dots indicator for current slide.
    - Auto-play functionality with pause on hover.
- **Footer Section:** Contains credits/info.

API Schema Design:

- **Image Object:**

```
{
"id": "1",
"url": "/images/img1.jpg",
```

"title": "Sample Image",

"description": "This is the first image in the slider"

}

API Endpoints:

- GET /api/images → Fetch all images.
- POST /api/images → Upload a new image.
- PUT /api/images/:id → Update an image's metadata.
- DELETE /api/images/:id → Delete an image.

Data Handling Approach

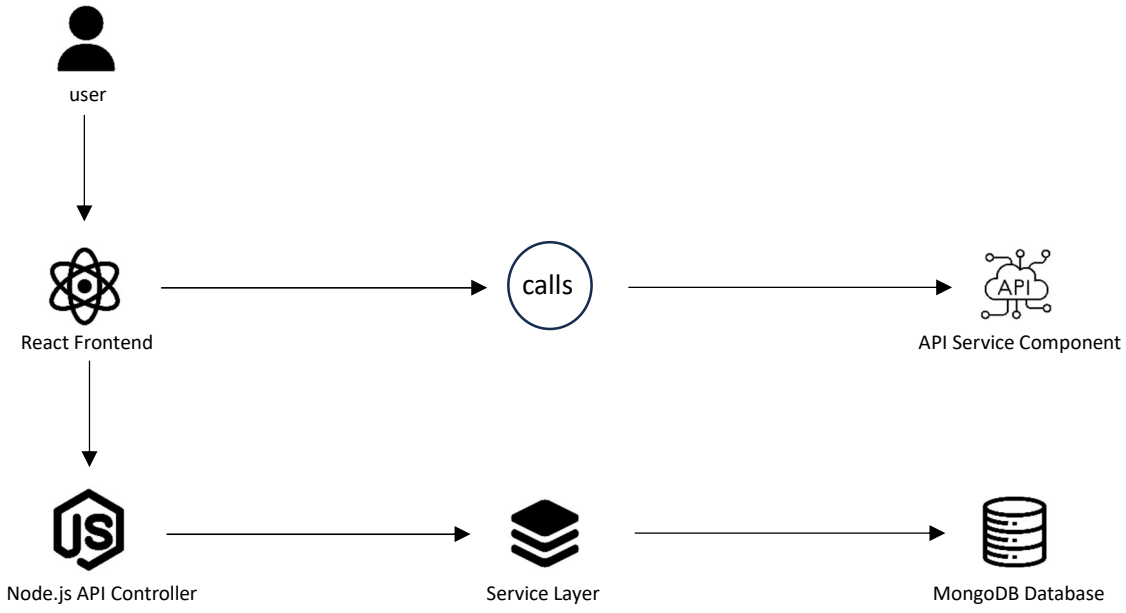
- **Data Flow:**
  - The **backend API** manages all CRUD operations for images.
  - **Frontend (React)** fetches data dynamically using fetch() or Axios.
  - On update/addition, the new data is immediately reflected without redeployment.
- **Performance Handling:**
  - Images are lazy-loaded for faster page rendering.
  - Caching mechanism in Node.js for repeated API requests.
  - Error handling for broken image links.
- **Security:**
  - Validation on API requests.
  - Only authorized requests can upload or delete images.

4. Component / Module Diagram

Modules:

- **Frontend (React):**
    - Slider Component
    - Navigation Component
    - Dots Indicator Component
    - API Service Component
- **Backend (Node.js):**
    - API Controller (handles requests)
    - Service Layer (business logic)
    - Database Layer (MongoDB operations)

Component Diagram (simplified):



Basic Flow Diagram

Workflow:

1. User opens the website → React loads slider component.
2. Frontend sends a **GET request** to Node.js API → retrieves image data.
3. Data is rendered dynamically inside the slider.
4. On new image upload, user sends a **POST request** → API stores metadata in MongoDB.
5. Updated data is fetched and displayed instantly in the slider.