**Roadmap — Sponsorship Management System (Modern Setup)**

**🔹 Phase 1: Planning & Requirements (1–2 weeks)**

**Goals:**

* Define scope and objectives.
* Identify user roles (Admin, Sponsor, Guardian).
* Finalize requirements (student mgmt, sponsorships, progress, reports, dashboards).
* Confirm tech stack (**Node.js + React + PostgreSQL**).

**Deliverables:**

* Requirements document.
* User stories & role definitions.
* Finalized architecture approach.

**🔹 Phase 2: System Design (2 weeks)**

**Goals:**

* Design PostgreSQL schema with Sequelize (Users, Students, Guardians, Sponsors, Education Records).
* Define ERD and system architecture.
* Create wireframes for **React pages** (Login, Dashboards, Reports, Management modules).
* Plan **RBAC middleware** and JWT authentication flow.

**Deliverables:**

* ERD diagram.
* System architecture diagram.
* Wireframes/mockups.

**🔹 Phase 3: Prototype Development (3–4 weeks)**

**Goals:**

* Setup Node.js + Express backend with Sequelize models/migrations.
* Implement authentication (JWT, bcrypt).
* Build initial React frontend with login + role-based dashboards.
* Create CRUD APIs for Students, Sponsors, Guardians.
* Display mock dashboard charts with Recharts.

**Deliverables:**

* Working prototype (login + dashboards).
* Feedback session with stakeholders.

**🔹 Phase 4: Core Development (4–6 weeks)**

**Goals:**

* Implement all major modules:
  + Student Management
  + Guardian Management
  + Sponsor Management
  + Education Progress Tracking (upload & view results)
  + Report generation (PDF/Excel with jsPDF + exceljs)
* Implement **RBAC middleware** (Admin vs Sponsor vs Guardian).
* Secure file upload (Multer + storage integration).
* Add **search & filtering APIs** for students/sponsors/education progress.

**Deliverables:**

* Fully functional backend (REST API).
* React frontend with role-based dashboards and forms.
* Data visualization (charts/graphs).

**🔹 Phase 5: Testing & QA (2 weeks)**

**Goals:**

* Unit tests (Jest for backend, React Testing Library for frontend).
* Integration tests (API + frontend flows).
* UAT with sample data.
* Security tests (JWT handling, password strength, role restrictions).

**Deliverables:**

* Test reports.
* Bug fixes & improvements.

**🔹 Phase 6: Deployment (1 week)**

**Goals:**

* Package backend & frontend in Docker containers.
* Setup docker-compose (API, DB, frontend).
* Deploy to cloud (Heroku, AWS, DigitalOcean, or local server).
* Configure DB backup and logs.

**Deliverables:**

* Live system accessible at a public URL.
* Admin and Sponsor access.
* Deployment guide & training notes.

**🔹 Phase 7: Maintenance & Scaling (Continuous)**

**Goals:**

* Monitor system performance.
* Apply security patches.
* Plan future upgrades (mobile app, SMS notifications, AI-based progress prediction).

**Deliverables:**

* Maintenance logs.
* Future roadmap updates.

**📅 Suggested Timeline (Approx. 3–4 Months)**

* **Month 1**: Planning + Design
* **Month 2**: Prototype Development
* **Month 3**: Core Development
* **Month 4**: Testing + Deployment
* **Ongoing**: Maintenance & Scaling