**1. Functional Requirements**

These are the specific features your website must have to serve its users:

**For Students:**

* **Account Management**:
  + Users can create, update, and delete accounts.
  + Users can log in and log out.
* **Course Catalog**:
  + Users can browse through courses by category (e.g., programming, web development).
  + Users can search for courses using a search bar.
  + Users can view course details (title, description, duration, price, etc.).
* **Enrollment and Purchase**:
  + Users can add courses to the cart.
  + Users can purchase courses securely using a payment gateway.
  + Users can view their purchased courses.
* **Course Access**:
  + Users can access purchased courses through their account.
  + Users can download or stream course materials (e.g., PDFs, videos).

**For Admin:**

* **Course Management**:
  + Admins can add, update, or delete courses.
  + Admins can set course details (description, price, category, etc.).
* **User Management**:
  + Admins can view users and manage their course enrollments.
* **Order Management**:
  + Admins can view and manage orders, refunds, and transactions.

**2. Non-Functional Requirements**

These are the qualities or attributes your website must have:

* **Performance**:
  + The website should load within 3 seconds for optimal user experience.
  + Course videos and files should be served quickly without long buffering.
* **Security**:
  + The website must use SSL for secure communication.
  + Payment transactions must be securely encrypted.
  + User data should be protected, and passwords should be hashed.
* **Scalability**:
  + The platform should be able to handle a growing number of users and courses.
* **Responsiveness**:
  + The website must be mobile-friendly and responsive on different screen sizes.

**3. Technical Requirements**

This covers the tools and technologies you'll need to build the website:

* **Frontend**:
  + HTML, CSS, JavaScript (with **React.js** or **Vue.js** for dynamic UI).
* **Backend**:
  + **Node.js** with **Express.js** (for APIs and server-side logic).
  + Alternatively, **PHP** with **Laravel** can also be used.
* **Database**:
  + **MongoDB** (NoSQL) or **MySQL** (relational) to store user data, courses, orders, etc.
* **Payment Gateway**:
  + Use **Razorpay** or **Paytm** for payment processing.
* **Version Control**:
  + **Git** for source code management, hosted on **GitHub** or **GitLab**.

**4. User Stories**

These help clarify what users will do and ensure you’re building features around real user needs:

* **As a student, I want to**:
  1. Browse available courses by category.
  2. Search for specific programming courses.
  3. View course details and pricing.
  4. Purchase a course securely online.
  5. Access my enrolled courses in my account.
* **As an admin, I want to**:
  1. Add, update, or delete courses.
  2. View and manage users and their enrollments.
  3. Manage orders and refunds.