**Project Design Phase**

**Solution Architecture**

|  |  |
| --- | --- |
| Date | 23 June 2025 |
| Team ID | LTVIP2025TMID20298 |
| Project Name | Docspot |
| Maximum Marks | 4 Marks |

### Smooth book discovery and ordering process

### Role-based dashboards for buyers, sellers, and admins

### Scalable user authentication and access control

### Secure and trackable order and inventory handling

### Real-time notifications for orders and stock status

### **Architecture Style:**

### Book Nest adopts a modular client-server architecture based on the MERN stack (MongoDB, Express.js, React.js, Node.js). This ensures performance, scalability, and ease of maintenance for dynamic e-commerce operations..

### **2. Components Breakdown**

* **Frontend (React.js + UI Libraries):**
  + Role-based dashboards for buyers, sellers, and admins
  + REST API integration using Axios
  + Responsive design using Tailwind CSS, Bootstrap, or Material UI
  + Navigation handled by React Router
  + Toast notifications and modals for interaction feedback
* **Backend (Node.js + Express.js):**
  + RESTful APIs for user auth, book management, cart, and orders
  + JWT-based authentication with role verification middleware
  + Business logic for order placement, stock validation, and admin approvals
  + File handling (book cover uploads) with Multer
* **Database (MongoDB + Mongoose):**
  + Collections: Users, Books, Orders, Wishlist
  + Flexible schema design with embedded documents (e.g., cart items)
  + Indexed fields for optimized book search and order history
* **Notification System:**
  + Email alerts (via SendGrid or Nodemailer) for order confirmation, cancellations, and approvals
  + Optional real-time toast alerts on client-side for UX enhancement

### **3. Data Flow Overview**

### **Buyer Action (e.g., place order) → Frontend Form Submission → Axios API Call → Express Backend Logic → MongoDB Data Save → Server Response → UI Update + Email Notification**

### **4. Deployment Environment**

* **Local Development:** localhost:3000 for client, localhost:5000 for server

### **Local Development**

### **Frontend: http://localhost:3000**

### **Backend: http://localhost:5000**

### **Production Deployment**

### **Hosted on platforms like Render, Vercel, or Netlify (frontend)**

### **Codebase Structure**

### **/frontend for client-side code**

### **/backend for server-side APIs**

### **Separate environment configs and startup scripts for each**

### **5. Key Qualities Ensured:**

**Quality Description**

**Scalability Modular routing, reusable React components, and indexed MongoDB queries handle scale easily.**

**Security JWT-based auth, protected routes, file validation, and role-based access.**

**Maintainability RESTful APIs, clean folder structure, and reusable code modules.**

**User Experience Fast-loading UI, smart filters, toast notifications, responsive design for all devices.**

