**Requirement Gathering and Analysis Phase**

**Solution Architecture**

|  |  |
| --- | --- |
| Date |  |
| Team ID | SWTID1719939027 |
| Project Name | Project – Book Nest |
| Maximum Marks |  |

**Solution Architecture for BOOKNEST:**

**Goals:**

1. **Solve Existing Business Problems:**
   * **Business Problems:** Efficient book catalog management, secure transactions, personalized recommendations.
   * **Technology Evaluation:** Evaluate options (e.g., React.js, Node.js, MongoDB) for scalability, security, and cost-effectiveness.
2. **Describe the Structure, Characteristics, Behavior, and Other Aspects of the Software:**
   * **Component Diagram:** Visualize frontend (web app), backend (microservices), and database interactions.
   * **Quality Attributes:** Define expectations for performance, security, and reliability.
   * **Behavioral Models:** Use UML diagrams to depict user interactions.
3. **Define Features, Development Phases, and Solution Requirements:**
   * **Key Features:**
     + **Book Purchase:** Users can browse and buy books.
     + **Cart and Wishlist:** Users can add books to their cart or wishlist.
     + **Order Tracking:** Users can check the status of their orders.
     + **Seller Portal:** Sellers manage inventory and fulfill orders.
     + **Admin Portal:** Admins oversee the entire system.
4. **Provide Specifications for the Solution:**
   * **User Portal:**
     + Add/remove books in cart and wishlist, manage orders, buy and view books.
   * **Seller Portal:**
     + Add/remove books, manage inventory, process orders.
   * **Admin Portal:**
     + Full system control, including order status management.

**Solution Requirements:**

**User Requirements:**

* **Book Purchase:** Users should browse books, view details, and make purchases.
* **Cart and Wishlist:** Users can add books to their cart or wishlist.
* **Order Tracking:** Users should track the status of their orders.

**Seller Portal Requirements:**

* **Inventory Management:** Sellers can add, update, or remove books from the catalog.
* **Order Fulfillment:** Sellers process orders placed by users.

**Admin Portal Requirements:**

* **Full Control:** Admins have access to all features (user management, order status changes).
* **Monitoring:** Admins can monitor system health and performance.

**Security Requirements:**

* **Authentication:** Secure login for users, sellers, and admins.
* **Authorization:** Role-based access control (user, seller, admin).
* **Data Encryption:** Protect sensitive data (e.g., user passwords).

**Performance Requirements:**

* **Responsiveness:** Fast loading times for book listings and order tracking.
* **Scalability:** Handle increasing user traffic efficiently.
* **Caching:** Cache frequently accessed data (e.g., book details).

**Development Phases:**

**Web Application (MERN Stack):**

* **Frontend (React.js):** User interface for browsing, purchasing, and managing orders.
* **Backend (Node.js, Express):** Handles user requests, authentication, and business logic.
* **Database (MongoDB):** Stores book data, user profiles, and order details.

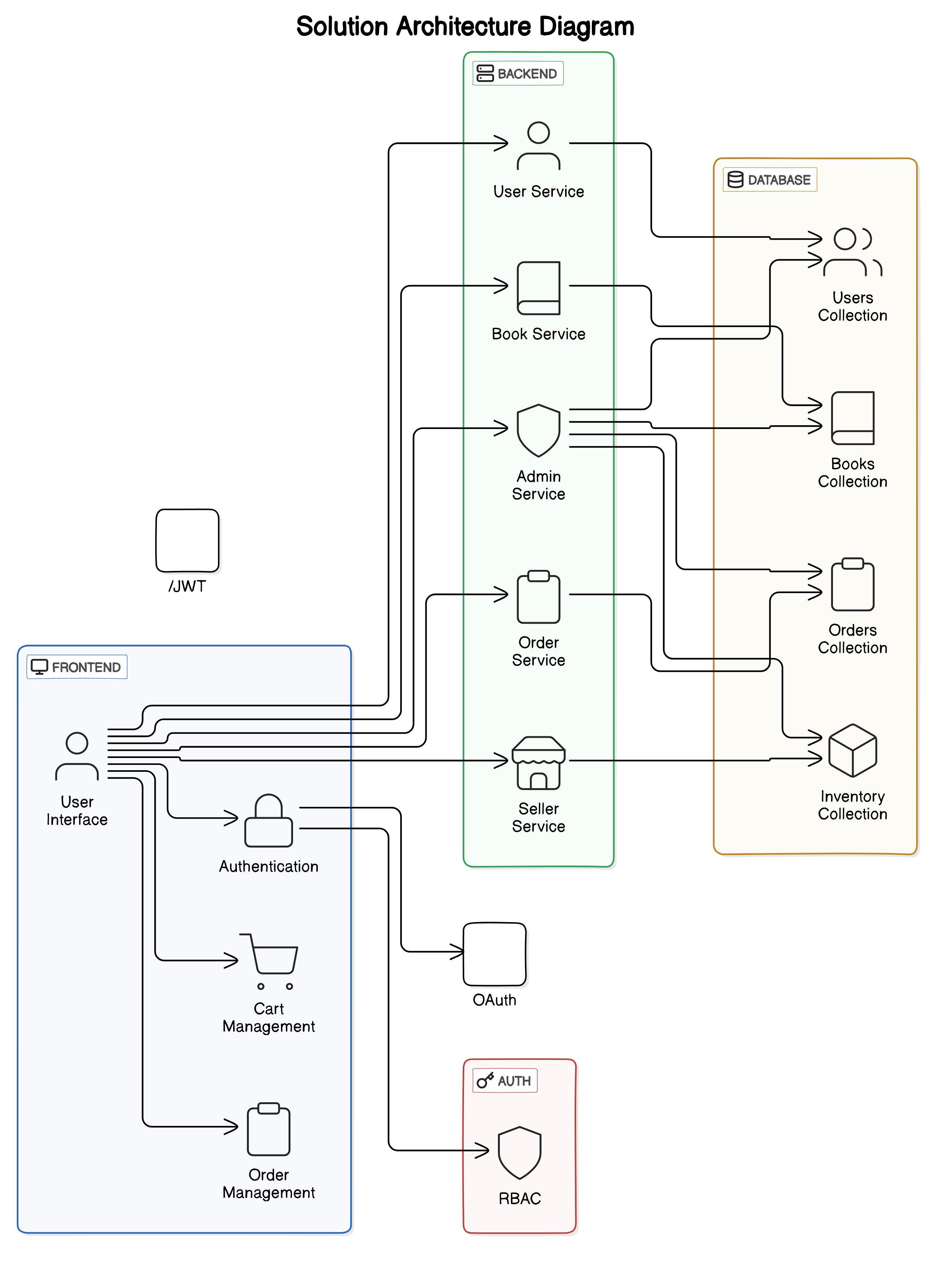
**Architecture Patterns:**

* **Microservices (Backend):** Divide functionality (e.g., user management, order processing) into separate services.
* **RESTful APIs:** Define clear API contracts for communication between frontend and backend.
* **State Management:** Manage application state (e.g., cart, wishlist) efficiently.
* **Authentication (OAuth or JWT):** Secure user logins and sessions.
* **Authorization (Role-Based Access Control):** Control access based on user roles (user, seller, admin).

**Solution Architecture Diagram:**

**Diagram Description:**

1. **Frontend (React.js):**
   * Components: User Interface, Authentication, Cart Management, Order Management.
2. **Backend (Node.js, Express):**
   * Microservices: User Service, Book Service, Order Service, Seller Service, Admin Service.
   * Communication: RESTful APIs.
3. **Database (MongoDB):**
   * Collections: Users, Books, Orders, Inventory.
4. **Authentication and Authorization:**
   * Services: OAuth/JWT for secure logins.
   * Role-Based Access Control (RBAC) for managing user permissions.

****