**Technology Stack (Architecture & Stack):**

|  |  |
| --- | --- |
| Date | 23rd May 3035 |
| Team ID | LTVIP2025TMID53019 |
| Project Name | BookNest |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Example: Order processing during pandemics for offline mode**

**Reference:** [**https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/**](https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/)



Guidelines:

Include all the processes (As an application logic / Technology Block)

Provide infrastructural demarcation (Local / Cloud)

Indicate external interfaces (third party API’s etc.)

Indicate Data Storage components / services

Indicate interface to machine learning models (if applicable)

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | How user interacts with application  Web UI | HTML, CSS,ReactJS+Vite/Bootstrap, CSS etc. |
|  | Application Logic-1 | Logic for a process in the application | JavaScript. |
|  | Database | Data Type, Configurations etc. | MongoDB, Mongoose. |
|  | File Storage | File storage requirements | MongoDB Cluster storage. |
|  | External API-1 | Purpose of External API used in the application |  |
|  | External API-2 | Purpose of External API used in the application |  |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Frontend uses React (via Vite), Tailwind CSS, Bootstrap for UI components, Axios for HTTP requests. Backend is built using Node.js with Express. | React, Vite, Tailwind CSS, Bootstrap, Axios, Node.js, Express.js |
|  | Security Implementations | Passwords are encrypted using bcrypt. CORS is implemented for secure cross-origin communication. Input validations prevent injection attacks. | bcrypt, CORS, express-validator, Helmet (optional) |
|  | Scalable Architecture | Follows a modular architecture separating frontend, backend, and database (3-tier). Can be containerized using Docker for scaling. | Node.js Microservices (optional), |
|  | Availability | Application can be deployed on cloud platforms (e.g., Heroku, Render, AWS) with horizontal scaling. Load balancers can be used if demand increases. | Cloud platforms (Render, AWS, etc.), Nginx (optional) |
|  | Performance | Efficient API calls with Axios, caching static content using CDN. MongoDB handles high-volume reads/writes efficiently. | Axios, MongoDB, CDN (e.g., Cloudflare), Compression |

**References:**

[**https://c4model.com/**](https://c4model.com/)

[**https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/**](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/)

[**https://www.ibm.com/cloud/architecture**](https://www.ibm.com/cloud/architecture)

[**https://aws.amazon.com/architecture**](https://aws.amazon.com/architecture)

[**https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d**](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d)