**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 15 April 2025 |
| Team ID | SWTID1743689010 |
| Project Name | Book Store |
| 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/>

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 | Web UI where users can browse, list, and  wishlist books. | HTML, CSS, JavaScript, React.js, Tailwind CSS |
|  | Application Logic-1 | Authentication logic and session management | Node.js, Express, bcrypt |
|  | Application Logic-2 | Logic for book listings, search, wishlist | Node.js, Express |
|  | Application Logic-3 | N/A (no AI or STT services used) | N/A |
|  | Database | Stores user data, books, and wishlist entries | MongoDB (Local via Compass or MongoDB Atlas) |
|  | Cloud Database | Optional cloud MongoDB for scalability | MongoDB Atlas (Optional) |
|  | File Storage | Stores book cover images | Cloudinary |
|  | External API-1 | N/A | N/A |
|  | External API-2 | N/A | N/A |
|  | Machine Learning Model | Not used in prototype | N/A |
|  | Infrastructure (Server / Cloud) | Local deployment using Node.js server | Localhost (Node.js + MongoDB), optional Vercel/Render |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | UI: React.js, Tailwind CSS; Backend: Node.js,  Express.js | Open-source JavaScript frameworks |
|  | Security Implementations | Password hashing for user auth; no other  security as per prototype goals | bcrypt; login via sessions or JWT |
|  | Scalable Architecture | 3-tier architecture (Frontend, API Layer, Database) | Scalable if later moved to cloud services |
|  | Availability | Localhost only for now. Easily deployable to  public cloud when needed | Local server; No performance optimizations yet |
|  | Performance | Not a concern for prototype. Caching and CDN  can be added later if needed | Local server; No performance optimizations yet |

**References:**

· <https://c4model.com/>

· https://www.lucidchart.com/blog/c4-model

· https://www.freecodecamp.org/news/how-to-create-software-architecture-diagrams-using-the-c4-model/

· https://www.ibm.com/architectures/patterns

· <https://aws.amazon.com/architecture/>

· https://cloud.google.com/architecture/framework

· https://miro.com/diagramming/c4-model-for-software-architecture/

· https://c4model.com/diagrams