**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 24 June 2025 |
| Team ID | LTVIP2025TMID53161 |
| Project Name | SB Foods - On-Demand Food Ordering Platform |
| Maximum Marks | 4 Marks |

**Technical Architecture Overview:**

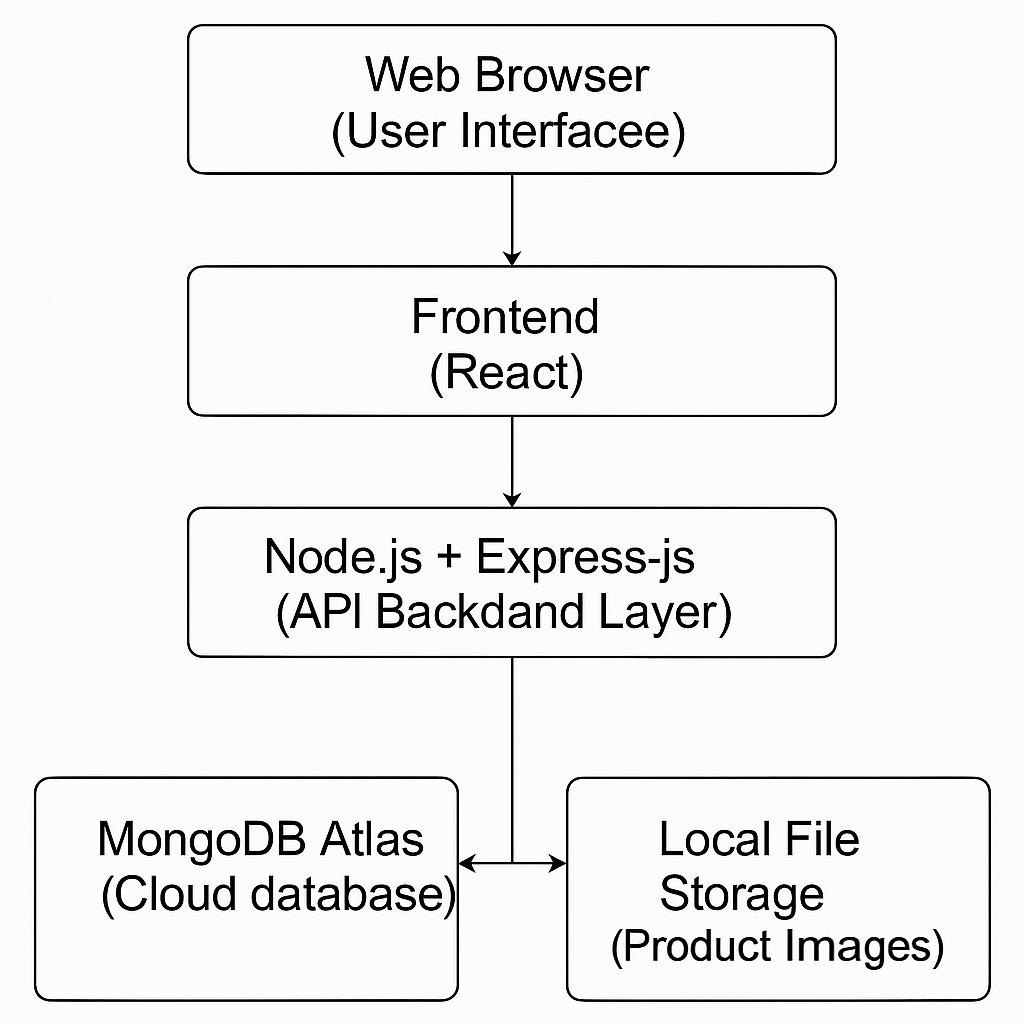
The SB Foods system is built using a **3-tier architecture**:

* **Frontend (Presentation Layer)** using React.js
* **Backend (Application Logic Layer)** using Node.js + Express.js
* **Database Layer** using MongoDB (NoSQL)

This ensures separation of concerns, easy scaling, and maintainability.

**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/)



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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | Web UI accessible via desktop/mobile | React.js, HTML, CSS, JavaScript |
| 2 | Application Logic-1 | Handles product listing, ordering, user flows | Node.js, Express.js |
| 3 | Application Logic-2 | OTP/Email Confirmation Service | Nodemailer / Twilio APIs |
| 4 | Application Logic-3 | Admin Dashboard logic, CRUD operations | Express.js |
| 5 | Database | NoSQL database for products, orders, users, etc. | MongoDB |
| 6 | Cloud Database | Cloud-hosted DB instance | MongoDB Atlas |
| 7 | File Storage | Product images and static files | Local FileSystem / Cloudinary |
| 8 | External API-1 | Pincode validation / address autocomplete | India Post API / Google Maps API |
| 9 | External API-2 | Payment integration | Razorpay / Stripe |
| 10 | Machine Learning Model | (Optional/Future) Personalized food recommendations | TensorFlow.js / Scikit-learn (TBD) |
| 11 | Infrastructure | Deployment configuration | Local server / Vercel / Railway.app |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology / Approach** |
| 1 | Open-Source Frameworks | Frameworks used in development | React.js, Node.js, Express, MongoDB |
| 2 | Security Implementations | Password hashing, secure login, CORS, data validation, rate limiting | bcrypt.js, Helmet, HTTPS, JWT, SHA-256 |
| 3 | Scalable Architecture | 3-tier, microservice-ready backend, separation of concerns | Express Router, Modular Folder Structure |
| 4 | Availability | 99.9% uptime, hosted on cloud, ready for scaling | Vercel / Railway / MongoDB Atlas |
| 5 | Performance | Fast API response, CDN for images, lazy loading, caching, pagination | React lazy, MongoDB indexing, Redis (optional) |

**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)