**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 15 April 2025 |
| Team ID | SWTID1743870576 |
| Project Name | SpendSmart: Your Personal Finance Companion |
| 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/)

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | How user interacts with the application: Web UI, responsive dashboard, analytics, and transaction entry. | React.js, HTML, CSS, JavaScript |
|  | Application Logic-1 | User authentication and registration (secure login, JWT-based sessions). | Node.js, Express.js, JWT |
|  | Application Logic-2 | Transaction management: add, edit, delete, categorize income and expenses. | Node.js, Express.js |
|  | Application Logic-3 | Analytics and visualization: generate charts, summaries, and financial insights. | Node.js, Express.js, Chart.js |
|  | Database | Stores user profiles, categorized transactions, and summaries. | MongoDB (NoSQL) |
|  | Cloud Database | Cloud-hosted database for scaling and high availability. | MongoDB Atlas |
|  | File Storage | (Optional/future) Storage for receipts or export files. | Local Filesystem / Cloud Storage |
|  | External API-1 | (Optional/future) Bank or wallet integration for auto-import of transactions. | Bank API / Plaid API (future) |
|  | External API-2 | (Optional/future) Currency conversion or financial news integration. | Exchange Rate API (future) |
|  | Machine Learning Model | (Optional/future) Predictive analytics for spending trends or anomaly detection. | Python, TensorFlow, Scikit-learn |
|  | Infrastructure (Server/Cloud) | Application deployment on cloud for high availability and scalability. | Vercel (frontend), Render (backend), MongoDB Atlas (database) |



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

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Utilizes open-source frameworks for full-stack web development, ensuring flexibility and community support. | React.js, Node.js, Express.js, MongoDB |
|  | Security Implementations | Implements JWT authentication, password hashing, secure API endpoints, and HTTPS to protect user data and sessions. | JWT, bcrypt, HTTPS, Helmet.js |
|  | Scalable Architecture | Modular MERN stack architecture supports horizontal scaling, cloud deployment, and easy integration of new features. | MERN stack, MongoDB Atlas, RESTful APIs |
|  | Availability | Cloud-hosted on platforms like Vercel (frontend), Render (backend), and MongoDB Atlas (database) for high uptime and reliability. | Vercel, Render, MongoDB Atlas |
|  | Performance | Optimized API endpoints, efficient data rendering, and use of Chart.js for real-time analytics ensure fast and smooth performance. | Node.js, Express.js, React.js, Chart.js |

**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)