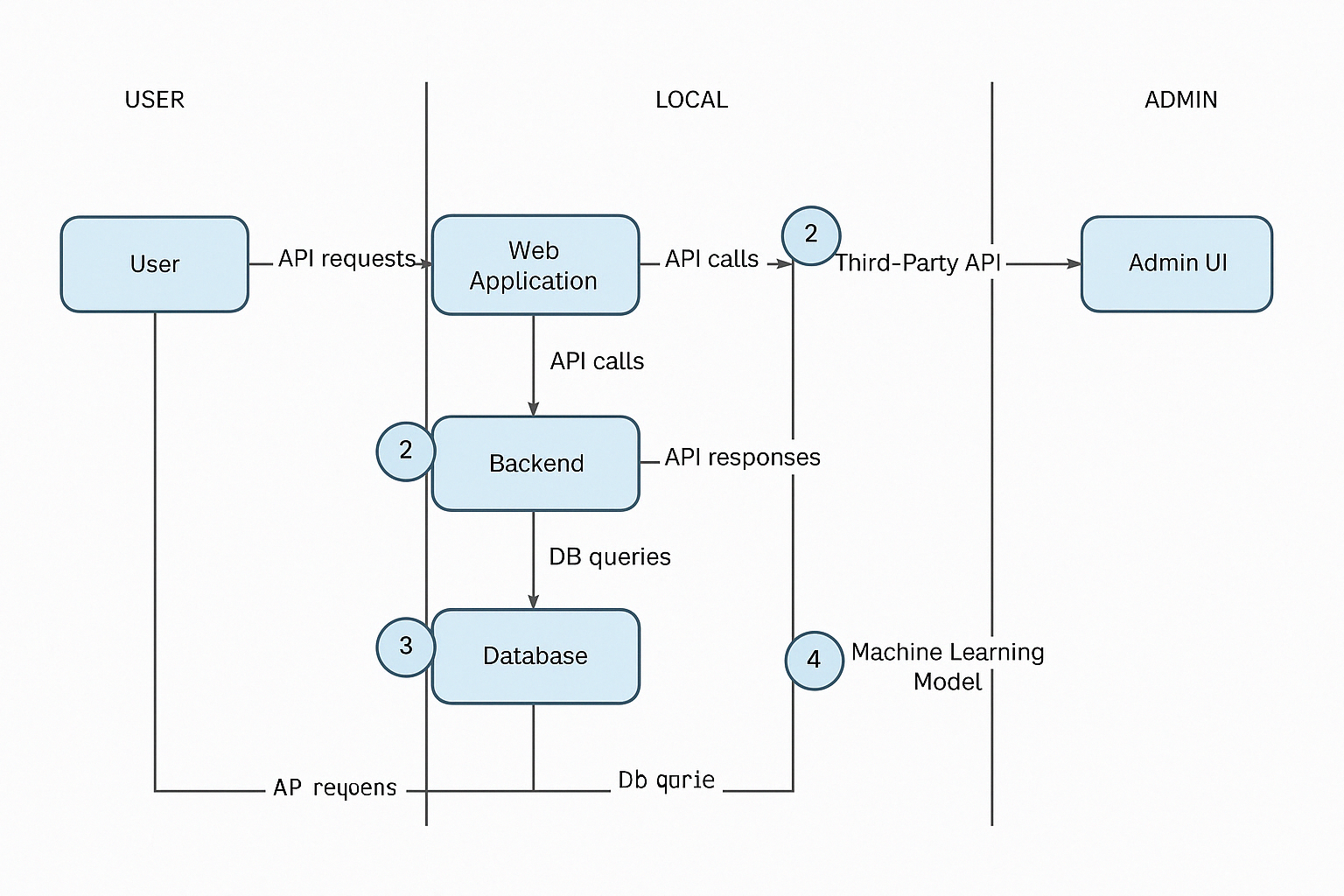
**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 11 April 2025 |
| Team ID | SWTID1743607402 |
| Project Name | ShopEZ: E-commerce Application |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

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



**Table-2: Application Characteristics:**

| Sr.No | Component | Description | Technology |
| --- | --- | --- | --- |
| 1. | User Interface | Web and mobile interfaces for customer/admin interactions | React.js, HTML5, CSS3, JavaScript |
| 2. | Application Logic-1 | Business logic for user registration, login, order management, admin panel | Node.js, Express.js |
| 3. | Application Logic-2 | Payment processing logic | Razorpay API / Stripe API |
| 4. | Application Logic-3 | Notification logic (emails, order updates) | Nodemailer, Firebase Cloud Messaging (FCM) |
| 5. | Database | Primary data storage for users, products, and orders | MongoDB (Mongoose ORM) |
| 6. | Cloud Database | Optional cloud-based DB deployment | MongoDB Atlas |
| 7. | File Storage | Product images and other file assets | Cloudinary / AWS S3 / Local File System |
| 8. | External API-1 | Shipping and logistics tracking | Shiprocket API / Delhivery API |
| 9. | External API-2 | Email & SMS notifications | SendGrid / Twilio |
| 10. | Machine Learning Model | Product recommendation or fraud detection (future scope) | Custom ML model (Python/Scikit-learn) on Flask API |
| 11. | Infrastructure (Server / Cloud) | Hosting of backend & frontend, database, file storage | Localhost (dev), Vercel (frontend), Render / AWS EC2 / Railway.app (backend) |

| S.No | Characteristics | Description | Technology |
| --- | --- | --- | --- |
| 1. | Open-Source Frameworks | Frameworks used to build frontend and backend | React.js, Node.js, Express.js, Mongoose, MongoDB |
| 2. | Security Implementations | JWT-based authentication for protected routes, password hashing, CORS control, secure headers, and best practices | JWT (Authorization: Bearer <token>), bcrypt, Helmet, CORS, HTTPS, OWASP Top 10 |
| 3. | Scalable Architecture | Modular, layered architecture allowing future scaling and deployment flexibility | RESTful APIs, /CD ready |
| 4. | Availability | Cloud-based deployment with options for distributed systems and auto-scaling | Railway / Render / AWS EC2, Load Balancer, Horizontal Scaling |
| 5. | Performance | Efficient routing, use of cache and CDNs, minimized API payloads, and lazy loading | Redis (caching), CDN (for static assets), Lazy Loading, MongoDB Indexes |

**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)