**Project Design Phase-II**

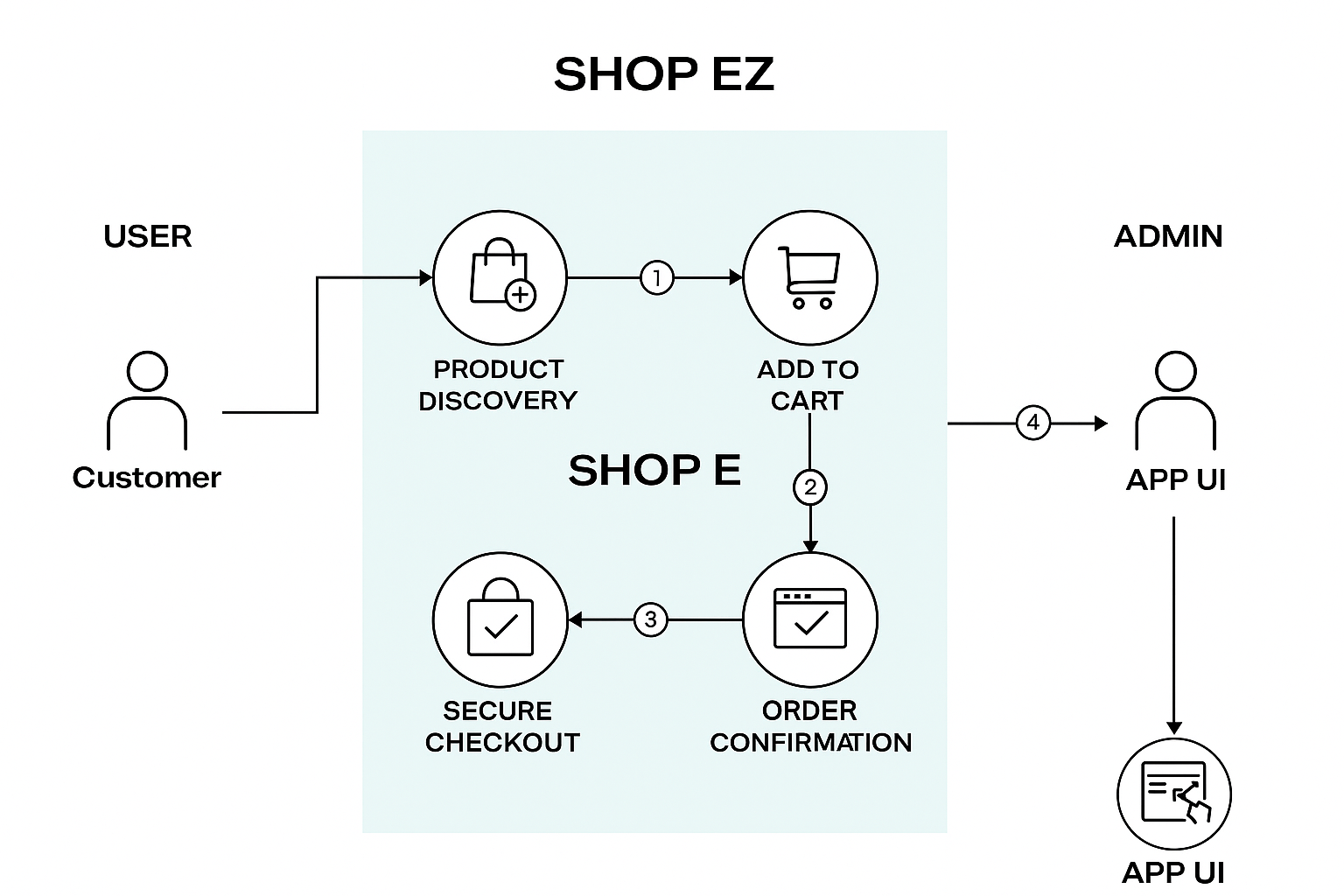
**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 31 January 3035 |
| Team ID | LTVIP2025TMID59486 |
| Project Name | ShopEZ: One-Stop Shop for Online Purchases |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

Guidelines:

The **ShopEZ** e-commerce platform is designed with a modular, full-stack architecture that ensures scalability, responsiveness, and efficiency. The application logic is divided into functional technology blocks including **user authentication, product discovery, cart management, checkout flow, admin dashboard, and review system**. The frontend is built using **React.js**, offering a dynamic and responsive user experience, while the backend leverages **Node.js and Express.js** to handle routing, APIs, authentication, and business logic.

****

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | User Interface | HTML, CSS, JavaScript, React.js |
|  | Application Logic-1 | Handles product listing, cart logic, authentication | Node.js, Express.js |
|  | Application Logic-2 | Logic for a process in the application | Admin dashboard logic (product & user management) |
|  | Application Logic-3 | Logic for a process in the application | IBM Watson Assistant |
|  | Database | Data Type, Configurations etc. | MySQL, NoSQL, etc. |
|  | Cloud Database | Hosted version of database | MongoDB Atlas (Cloud NoSQL Database) |
|  | File Storage | File storage requirements | IBM Block Storage or Other Storage Service or Local Filesystem |
|  | External API-1 | Purpose of External API used in the application | IBM Weather API, etc. |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Frontend and backend built using open-source libraries | React.js, Node.js, Express.js, Mongoose, Bootstrap |
|  | Security Implementations | Authentication, password encryption, secure payments | JWT, bcrypt, HTTPS, OAuth 2.0, OWASP best practices |
|  | Scalable Architecture | Designed using modular services with potential for microservices in future | MERN Stack, RESTful APIs |
|  | Availability | Cloud-hosted components ensure availability and can scale horizontally | MongoDB Atlas, Render, Vercel, GitHub Actions |
|  | Performance | Cloud-hosted components ensure availability and can scale horizontally. | Browser Caching, Lazy Loading, CDN (Vercel), Axios |

**References:**

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

[**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)