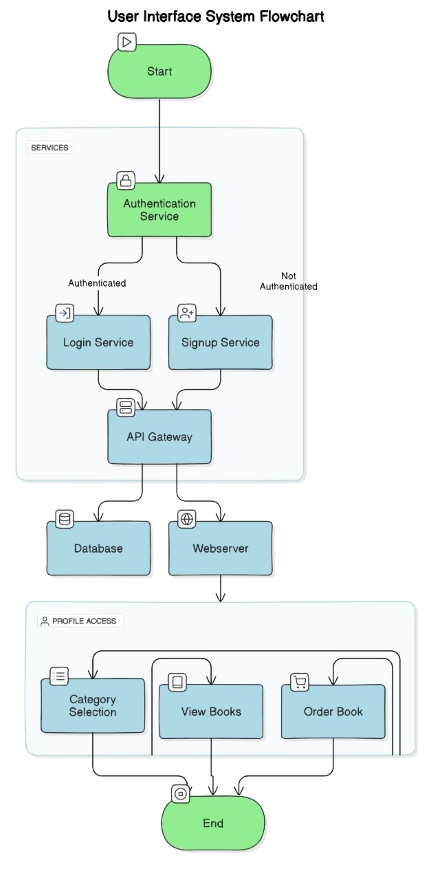
**Requirement Gathering and Analysis Phase**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 05 July 2024 |
| Team ID | SWTID1720014974 |
| Project Name | Book-Store |
| Maximum Marks | 3 |

**Technical Architecture:**

****

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Web interface for user interaction. | HTML, CSS, JavaScript, React.js |
|  | Application Logic | Backend logic for bookstore processes | Node.js, Express.js |
|  | Database | Data storage and management | MongoDB |
|  | Cloud Database | Cloud-based database service | MongoDB Atlas |
|  | File Storage | Storage for uploaded files | Local Filesystem |
|  | External API-1 | Integration for external services | Payment Gateway API |
|  | External API-2 | Integration for external services | ISBN Lookup API |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | List the open-source frameworks used | React.js, Node.js, Express.js, MongoDB |
|  | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | JWT (JSON Web Tokens), HTTPS, bcrypt for password hashing, CORS policies |
|  | Scalable Architecture | Justify the scalability of architecture (3 – tier, Micro-services) | Microservices architecture leveraging Node.js for backend services |
|  | Availability | Justify the availability of application (e.g. use of load balancers, distributed servers etc.) | Deployment on scalable cloud platforms like AWS EC2 with Elastic Load Balancing |
|  | Performance | Design considerations for application performance (e.g., requests per second, use of Cache, CDN) | Use of caching with Redis, CDN integration for static assets delivery |