| **Term** | **Definition and Purpose** |
| --- | --- |
| **Authorized User** | An admin-level user with permission to manage students, instructors, and system settings. Ensures proper access control. |
| **Student** | A registered learner whose attendance is monitored and recorded by the system. |
| **Instructor** | A faculty member responsible for marking student attendance and managing course data. |
| **Registration** | The process of entering a user (student or instructor) into the system, including details and ID photo. |
| **Unregistration** | Removing a user from the system, deleting their data and revoking access. |
| **Registered Device** | A phone or laptop assigned to an instructor, approved for taking attendance. |
| **Barcode Scanning** | Using a device’s camera to read student ID barcodes for attendance tracking. |
| **Attendance Report** | A document or summary showing student attendance, viewable by authorized users. |
| **Role-Based Access Control (RBAC)** | A system that restricts access based on user roles, ensuring only the right users perform sensitive actions. |
| **Multi-Factor Authentication (MFA)** | A login process requiring multiple verification steps (e.g., password + code) for extra security. |
| **HTTPS / SSL Security** | Internet protocols that encrypt data to protect it during online transmission. |
| **Cloud-Based System** | A system hosted on the internet instead of local servers, allowing remote access and easier scaling. |
| **MongoDB Atlas / MySQL** | Cloud databases used to store system data. MongoDB is document-based, MySQL is table-based. |
| **Device Location Feature** | Allows authorized users to find lost registered devices using location tracking tools. |
| **Scalability** | The system’s ability to grow in capacity (users, data) without reducing performance. |
| **Deployment Timeline** | The planned dates for system testing and going live (e.g., May 15: check; May 22: full launch). |
| **Flask** | A lightweight Python-based web framework used to build the backend of the system. Purpose: Handling server logic and APIs. |
| **Backend** | The server-side logic of the application that handles data storage, security, and application logic. Often built using Flask or similar frameworks. |
| **Frontend** | The part of the system that users interact with (UI). It communicates with the backend to display or send data. |
| **React Native** | A mobile app framework by Meta used to build native apps for iOS and Android. Purpose: Delivering a cross-platform user interface for instructors or students. |