**Software Requirement Specification (SRS)**

**Internship Management System (IMS)**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to define the requirements for the **Internship Management System (IMS)**. The system is a web-based platform designed to streamline the management of interns and their tasks. It enables **admins** (HR/Managers) to onboard interns, assign and track tasks, and provide feedback, while allowing **interns** to apply for internships, access their tasks, submit work, and monitor progress.

This SRS will serve as a guide for stakeholders, developers, and testers to ensure the project aligns with user needs and organizational goals.

**1.2 Scope**

The IMS will provide the following key functionalities:

* **Admin Capabilities:**
  + Onboard, approve, and manage interns.
  + Create and manage internship programs across multiple domains.
  + Approve or Reject applications for an internship
  + Assign tasks with deadlines, review submissions, and provide feedback.
  + Monitor progress through performance dashboards.
* **Intern Capabilities:**
  + Apply for available internship opportunities but can have only one active internship at a time (only one domain).
  + Log in to view assigned tasks and deadlines.
  + Submit completed work in file or link format.
  + Track progress, receive feedback, and view task completion status.

The system will be a **full-stack application** with a secure backend (Node.js, Express), modern frontend (React + Tailwind CSS), and a scalable NoSQL database (MongoDB).

**1.3 Intended Users**

* **Admins (HR/Managers):** Responsible for creating internship opportunities, onboarding interns, managing tasks, and reviewing performance.
* **Interns (Students/Trainees):** Responsible for applying, completing assigned tasks, and tracking their own progress.

**1.4 Definitions & Acronyms**

* **IMS:** Internship Management System
* **CRUD:** Create, Read, Update, Delete
* **JWT:** JSON Web Token (used for authentication)

**2. Functional Requirements**

**2.1 Admin Features**

* Secure **Login/Authentication**.
* Add/Edit/Delete intern profiles.
* Create more admin
* Approve/Decline internship applications.
* Create/Update/Delete internships in multiple domains.
* Assign tasks with deadlines to interns.
* Accept/Decline submitted tasks.
* Track intern performance via dashboards.
* Provide written feedback and performance ratings.

**2.2 Intern Features**

* Secure **Login/Authentication**.
* Apply for internships in available domains but an intern can have only one active internship at a time (only one domain).
* View assigned tasks, deadlines, and requirements.
* Submit work (file upload/link submission).
* Receive feedback and task acceptance/rejection updates.
* Track progress via dashboard (percentage completed, pending tasks, etc.).

### ****3. Non-Functional Requirements****

* **Performance**
  + The system must support at least **100 concurrent interns** without performance degradation.
  + Average response time for user actions should be **under 2 seconds**.
* **Security**
* All user credentials must be stored using a **secure password hashing mechanism**.
* The system will enforce **role-based access control (Admin vs Intern)**.
* **Token-based authentication** will be used to ensure secure session management.
* **Usability**
  + Dashboards should be **intuitive and user-friendly**.
  + The application must provide a **responsive design** compatible with both desktop and mobile devices.
* **Scalability**
  + Backend architecture will be **modular**, allowing the easy addition of new roles (e.g., Mentors).
  + The database schema must support scaling to **thousands of users and tasks** without redesign.
* **Reliability**
  + The system should maintain an **uptime of at least 99.5%** under cloud deployment.
  + The database must support **automatic backups** to ensure data recovery in case of failure.

### ****4. System Design (High-Level)****

* **Frontend:** React with Tailwind CSS for building responsive dashboards.
* **Backend:** Node.js with Express, exposing RESTful APIs for all operations.
* **Database:** MongoDB with collections: **Users, Internships, Applications, Tasks, Submissions, Feedback**.
* **Authentication:** Token-based login with **role-based access** for Admins and Interns.
* **Deployment:** Cloud-based deployment supporting auto-scaling (e.g., frontend on Vercel/Netlify, backend on Render/Heroku, database on MongoDB Atlas).

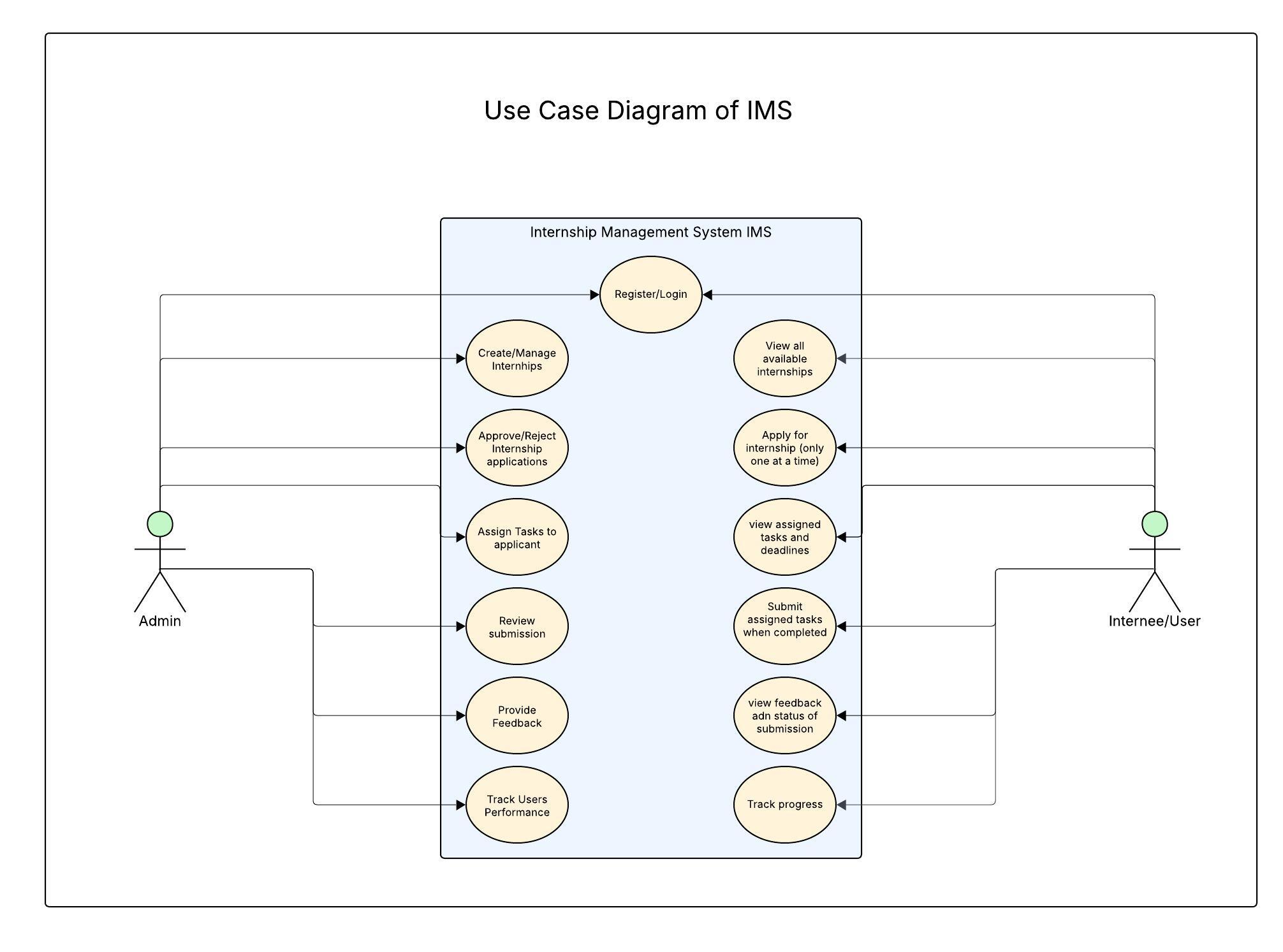
**5. Use Case Diagram (High-Level)**

**Actors:**

* Admin
* Intern

**Interactions:**

* Admin: Login → Manage Internships → Assign Tasks → Review Submissions → Give Feedback
* Intern: Apply → Login → View Tasks → Submit Work → Track Progress



**6. Future Enhancements**

* **Email Notifications:** For task assignment, deadlines, and feedback.
* **Analytics Dashboard:** For performance reports and intern evaluation.
* **Integration with GitHub/Google Drive:** For direct project submission.
* **Certificate Generation Module:** Automatic certificate upon internship completion.

**7. Expected Outcome**

The successful implementation of IMS will:

* **Streamline intern onboarding and tracking.**
* **Increase efficiency** in task assignment and review.
* Provide **real-time transparency** between interns and admins.
* Enhance professional development for interns through structured tracking and feedback.