**Software Requirement Specification for**

**“Course File Automation”**

|  |  |
| --- | --- |
| **Name** | MAGASRI R |
| **Roll no** | 7376221EC213 |
| **Seat no** | 164 |
| **Project ID** | 29 |
| **Problem Statement** | COURSE FILE AUTOMATION |

1. **Introduction:** 
   1. **Purpose:**

• The purpose of this project is to develop a Course File Automation system aimed at simplifying the creation, management, and maintenance of course files for academic purposes. The system will streamline the process of handling course-related documentation, ensuring that it is consistent, comprehensive, and easily accessible.

* 1. **Scope of Project:** 
     + The Course File Automation system will serve as a comprehensive module for academic management, enabling users to create, edit, and manage course files efficiently. The system will include features such as document templates, automated document generation, version control, and role-based access.
     + Additionally, the system will incorporate role-based access control to ensure that users have appropriate permissions based on their roles, whether they are instructors, administrators, or other academic personnel. This will help maintain the integrity and security of the documents.
     + The system will also provide analytical tools to assess the completeness and quality of course files, offering insights into areas that may need improvement. By streamlining these processes, the Course File Automation system aims to reduce administrative workload, improve documentation standards, and support academic excellence.

1. **System Overview:** 
   1. **Users:** 
      1. **Instructors:** 
         * Create and manage course files.
         * Use predefined templates to ensure consistency.
         * Generate documents automatically.
         * Perform version control.
      2. **Administrators:** 
         * Oversee the course file management process. • Access analytical dashboards for monitoring.
         * Manage user access and permissions.

* 1. **Features:** 
     1. **Login and Registration:** 
        + Users can register for an account or log in with their existing account.
     2. **Course File Management:** 
        + Instructors can create and manage course files using templates.

1. **Automated Document Generation:**

• System generates documents automatically based on predefined templates.

1. **Version Control:**

• Maintain different versions of course files, allowing users to track changes.

1. **Admin Access:**

• Admins can view all course files and monitor the creation and management process.

1. **Analytics Dashboard:**

• Provides insights into course file completeness and quality metrics.

1. **System Requirements Specification 3.1. Functional Requirements**

**1. User Management:**

* + - * Users can register and log in.
      * Admins have access control with an analytical dashboard and dedicated features.

**2. Course File Management:**

* + - * Instructors can create and manage course files using templates.
      * Data entry forms include fields for course details and related documents.

**3. Automated Document Generation:**

• System generates documents automatically based on predefined templates.

1. **Version Control:**

• Maintain different versions of course files, allowing users to track changes.

1. **Admin Dashboard:** 
   * + - Admins can view and manage all course files.
       - Includes tools for monitoring the course file management process.

**3.2. Non-Functional Requirements**

**1. Performance:**

• The system must respond to user actions within 2 seconds to ensure efficient usability and handle a concurrent user load of at least 100 users without significant performance degradation.

**2. Security:**

• User data must be encrypted during transmission and storage, and access to sensitive functionalities should be restricted to authorized admin users through secure authentication mechanisms.

**3. Usability:**

• The user interface should be intuitive and user-friendly, with clear and concise error messages to guide users in case of input errors or system failures.

**4. Reliability:**

• The system should be available 24/7 with minimal downtime and should have a backup and recovery mechanism to prevent data loss in case of system failures or crashes.

**5. Scalability:**

• The system should be designed to accommodate an increasing number of users and data volume over time and should be scalable to support additional features and functionalities as per future requirements.

1. **System Architecture:** 
   1. **Backend Entities:** 
      1. **User Entity:**

|  |  |
| --- | --- |
| Name | String |
| Email | String |
| Password | Hash code |
| Role | String (Instructor/Admin) |

* + 1. **Marks Entity:**

|  |  |
| --- | --- |
| StudentId | String |
| CourseId | String |
| TestId | String |
| Date | Date |
| Marks | Array of Objects  (CO-wise marks) |

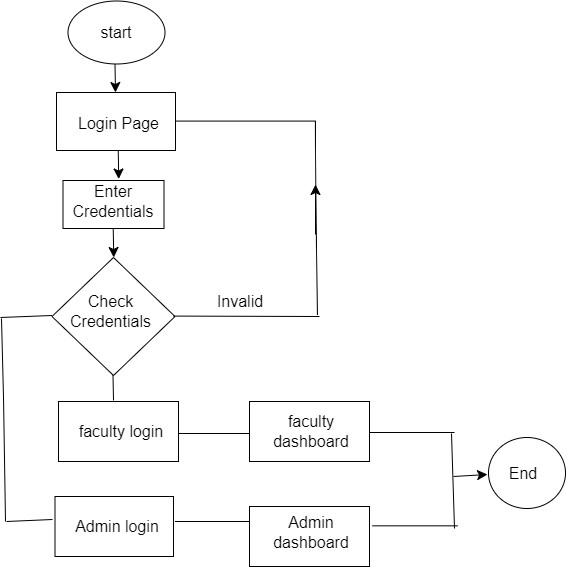
* + 1. **CO-PO Mapping Entity:**

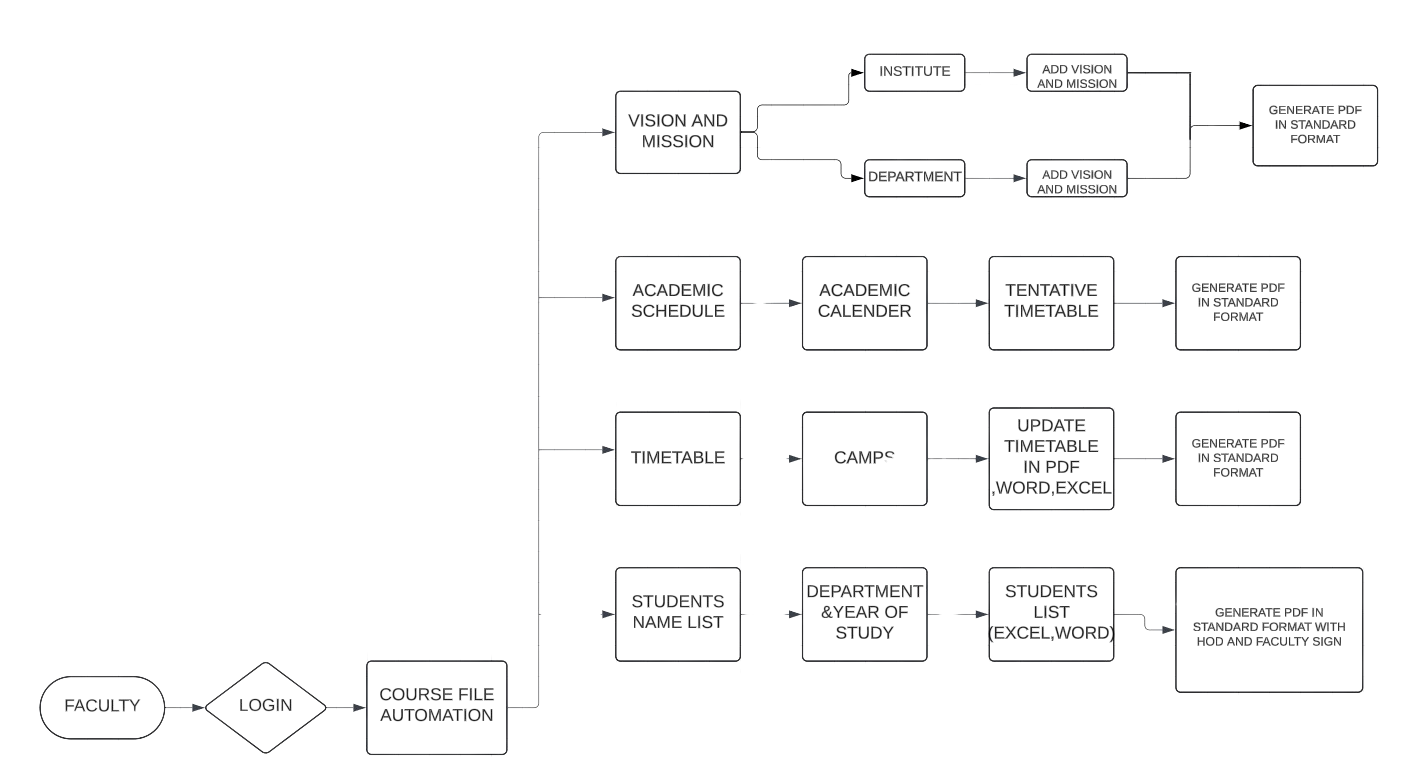
|  |  |
| --- | --- |
| CourseId | String |
| COs | Array of Objects |
| POs | Array of Objects |
| Mapping | Array of Objects (CO-PO relations) |

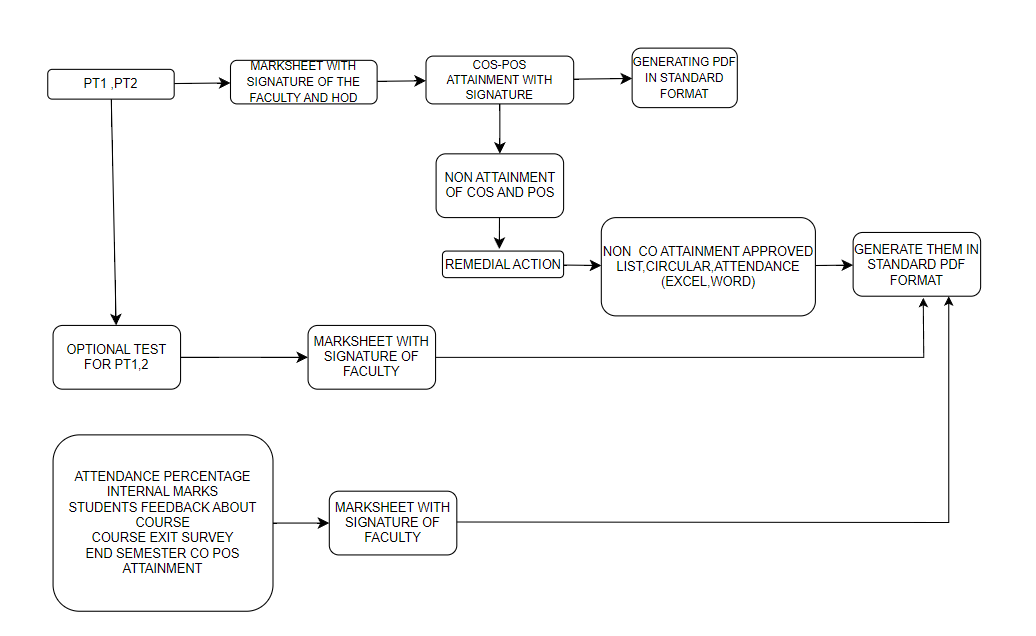
* 1. **Technology Stack:**

|  |  |
| --- | --- |
| **Front-End** | React Js |
| **Back-End** | Node Js |
| **Database** | MongoDB |
| **API** | Restful API |

1. **Work Flow Diagram:**





****