

**HIGH LEVEL DESIGN (HLD) SIGN-OFF**

**Authorization Memorandum**

This form is to sign-off completion of the Design Phase for **CAMPUS MANAGEMENT SOFTWARE AT WORLD SKILL CENTER (WSC).**

World Skill Centre (WSC) acknowledges receipt of the deliverables as part of the Design Phase through the submission of this document.

|  |  |
| --- | --- |
| **MODULE NAME** | **Examination Module** |
| **MODULE OWNER** | **Mr. Suresh Kumar** |

**WSC AUTHORITY NAME AND SIGNATURE**

**SOUL AUTHORITY NAME AND SIGNATURE**

Logo

Description automatically generated Logo

Description automatically generated

Logo

Description automatically generated

**High Level Design (HLD)**

Of

**Examination module**

For Implementation of

**Campus Management Software**

at

**World Skill Center (WSC)**

**Sustainable Outreach And Universal**

**Leadership (SOUL) Limited**

**Confidentiality**

This document is a strictly confidential communication to and solely for the use of the recipient and may not be reproduced or circulated without SOUL Ltd’s prior written consent. If you are not the intended recipient, you may not disclose or use the information in this documentation in any way

**Document Control History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Version No.** | **Created By** | **Affected Sections** | **Reviewed By** |
| 28-August-2023 | 1.0 | Subhodeep Dey | All |  |
|  |  |  |  |  |

# Project Control

|  |  |
| --- | --- |
| **Package Number:** | PKG 0111 |
| **Project Name:** | Campus Management Software At World Skill Center (WSC) |
| **Location:** | Bhubaneswar |
| **Customer Name:** | World Skill Center (WSC) |
| **Customer Contact Information:** | Mancheswar, Bhubaneswar |
| **Project Manager / Email ID:** | Sharmistha Panda  sharmistha.panda@soulunileaders.com |
| **Module SME / Email ID:** | Mr. Suresh Kumar (suresh.kumar@worldskillcenter.org) |

Table Of Contents

[HIGH LEVEL DESIGN (HLD) SIGN-OFF 1](#_Toc21289)

[Authorization Memorandum 1](#_Toc26715)

[Project Control 3](#_Toc10824)

[1 Introduction 8](#_Toc25316)

[1.1 Background 8](#_Toc29972)

[1.2 Scope and Purpose of the document 8](#_Toc9450)

[1.3 Assumptions 9](#_Toc21227)

[1.4 Dependencies 9](#_Toc32334)

[1.5 Current IT Environment 10](#_Toc21023)

[1.6 Document Structure 10](#_Toc12850)

[1.7 Required System Architecture 10](#_Toc10006)

[1.8 Interoperability Framework 11](#_Toc20995)

[1.9 Technical Decision Summary 11](#_Toc6565)

[1.10 Architectural Risks & Mitigation 12](#_Toc31217)

[1.11 Reference 12](#_Toc16929)

[2 Enterprise Architecture Framework (Business Architecture) 12](#_Toc16967)

[2.1 Business Drivers 12](#_Toc15339)

[2.2 Business Concerns 13](#_Toc5164)

[2.3 Business Goals 13](#_Toc24976)

[2.4 Business Value Chain 13](#_Toc17887)

[2.5 Business Context Diagram 14](#_Toc1712)

[2.6 Business Process Flow 14](#_Toc16752)

[2.6.1 Examination Process Flow 14](#_Toc1605)

[2.6.2 System Context Diagram 15](#_Toc21110)

[2.7 Conceptual Architecture 15](#_Toc28677)

[3 Architecture Views 15](#_Toc23134)

[3.1 Exam Type 15](#_Toc13812)

[3.1.1 Use Case Diagram 16](#_Toc26689)

[3.1.2 Design of Workflow 16](#_Toc863)

[3.1.3 Validations 16](#_Toc22631)

[3.1.4 Field List: 17](#_Toc28888)

[3.1.5 Processes After Form Submission 17](#_Toc223)

[3.1.6 Process Flow: 17](#_Toc9451)

[3.1.7 Pre-requisites and Dependency 17](#_Toc23507)

[3.2 Assessment Component 17](#_Toc12196)

[3.2.1 Use Case Diagram 18](#_Toc25987)

[3.2.2 Design of Workflow 18](#_Toc7969)

[3.2.3 Validations 18](#_Toc21256)

[3.2.4 Field List: 19](#_Toc32189)

[3.2.5 Processes After Form Submission 19](#_Toc12372)

[3.2.6 Process Flow: 19](#_Toc11932)

[3.2.7 Pre-requisites and Dependency 20](#_Toc9993)

[3.3 Grading Scale 20](#_Toc934)

[3.3.1 Use Case Diagram 20](#_Toc32020)

[3.3.2 Design of Workflow 20](#_Toc41)

[3.3.3 Validations 20](#_Toc4413)

[3.3.4 Field List: 21](#_Toc5563)

[3.3.5 Processes After Form Submission 21](#_Toc3328)

[3.3.6 Process Flow: 21](#_Toc21790)

[3.3.7 Pre-requisites and Dependency 22](#_Toc32370)

[3.4 Exam Declaration 22](#_Toc6947)

[3.4.1 Use Case Diagram 23](#_Toc25080)

[3.4.2 Design of Workflow 23](#_Toc16877)

[3.4.3 Validations 23](#_Toc8993)

[3.4.4 Field List: 25](#_Toc24832)

[3.4.5 Processes After Form Submission 29](#_Toc5344)

[3.4.6 Process Flow: 29](#_Toc15096)

[3.4.7 Pre-requisites and Dependency 29](#_Toc9459)

[3.5 Module Wise Exam Group 29](#_Toc11752)

[3.5.1 Use Case Diagram 29](#_Toc27462)

[3.5.2 Design of Workflow 30](#_Toc25511)

[3.5.3 Validations 30](#_Toc32133)

[3.5.4 Field List: 32](#_Toc10649)

[3.5.5 Processes After Form Submission 34](#_Toc28741)

[3.5.6 Process Flow: 34](#_Toc21432)

[3.5.7 Pre-requisites and Dependency 35](#_Toc31336)

[3.6 Exam Evaluation Plan 35](#_Toc27880)

[3.6.1 Use Case Diagram 35](#_Toc5289)

[3.6.2 Design of Workflow 36](#_Toc18135)

[3.6.3 Validations 36](#_Toc32365)

[3.6.4 Field List: 37](#_Toc19963)

[3.6.5 Processes After Form Submission 40](#_Toc6739)

[3.6.6 Process Flow: 40](#_Toc27247)

[3.6.7 Pre-requisites and Dependency 40](#_Toc26655)

[3.7 Exam Paper Setting 41](#_Toc15544)

[3.7.1 Use Case Diagram 41](#_Toc20192)

[3.7.2 Design of Workflow 41](#_Toc15807)

[3.7.3 Validations 41](#_Toc16784)

[3.7.4 Field List: 43](#_Toc17277)

[3.7.5 Processes After Form Submission 44](#_Toc22917)

[3.7.6 Process Flow: 44](#_Toc14384)

[3.7.7 Pre-requisites and Dependency 44](#_Toc7513)

[3.8 Component Wise Evaluation 45](#_Toc18)

[3.8.1 Use Case Diagram 45](#_Toc8253)

[3.8.2 Design of Workflow 45](#_Toc10726)

[3.8.3 Validations 45](#_Toc17628)

[3.8.4 Field List: 46](#_Toc10460)

[3.8.5 Processes After Form Submission 47](#_Toc2087)

[3.8.6 Process Flow: 48](#_Toc2709)

[3.8.7 Pre-requisites and Dependency 48](#_Toc7116)

[3.9 Component Wise Evaluation Tool 48](#_Toc29267)

[3.9.1 Use Case Diagram 48](#_Toc5417)

[3.9.2 Design of Workflow 49](#_Toc10902)

[3.9.3 Validations 49](#_Toc23792)

[3.9.4 Field List: 51](#_Toc26806)

[3.9.5 Processes After Form Submission 52](#_Toc11482)

[3.9.6 Process Flow: 52](#_Toc31290)

[3.9.7 Pre-requisites and Dependency 53](#_Toc5155)

[3.10 Component Wise Reevaluation 53](#_Toc4605)

[3.10.1 Use Case Diagram 53](#_Toc159)

[3.10.2 Design of Workflow 54](#_Toc23390)

[3.10.3 Validations 54](#_Toc31136)

[3.10.4 Field List: 56](#_Toc27663)

[3.10.5 Processes After Form Submission 57](#_Toc25364)

[3.10.6 Process Flow: 57](#_Toc12607)

[3.10.7 Pre-requisites and Dependency 57](#_Toc4732)

[3.11 Component Wise Reevaluation Tool 57](#_Toc6965)

[3.11.1 Use Case Diagram 57](#_Toc3245)

[3.11.2 Design of Workflow 58](#_Toc8848)

[3.11.3 Validations 58](#_Toc14964)

[3.11.4 Field List: 59](#_Toc23758)

[3.11.5 Processes After Form Submission 60](#_Toc5021)

[3.11.6 Process Flow: 60](#_Toc16842)

[3.11.7 Pre-requisites and Dependency 61](#_Toc31465)

[3.12 Final Semester Result 61](#_Toc4677)

[3.12.1 Use Case Diagram 61](#_Toc6902)

[3.12.2 Design of Workflow 62](#_Toc4036)

[3.12.3 Validations 62](#_Toc30785)

[3.12.4 Field List: 64](#_Toc32018)

[3.12.5 Processes After Form Submission 65](#_Toc3203)

[3.12.6 Process Flow: 66](#_Toc22000)

[3.12.7 Pre-requisites and Dependency 66](#_Toc24091)

[3.13 Final Semester Result Tool 66](#_Toc30003)

[3.13.1 Use Case Diagram 66](#_Toc19027)

[3.13.2 Design of Workflow 67](#_Toc16431)

[3.13.3 Validations 67](#_Toc29077)

[3.13.4 Field List: 68](#_Toc3174)

[3.13.5 Processes After Form Submission 69](#_Toc14102)

[3.13.6 Process Flow: 69](#_Toc27915)

[3.13.7 Pre-requisites and Dependency 69](#_Toc29385)

[3.14 Cumulative Marksheet Tool 70](#_Toc19885)

[3.14.1 Use Case Diagram 70](#_Toc4061)

[3.14.2 Design of Workflow 70](#_Toc19241)

[3.14.3 Validations 70](#_Toc25695)

[3.14.4 Field List: 71](#_Toc1372)

[3.14.5 Processes After Form Submission 72](#_Toc12981)

[3.14.6 Process Flow: 72](#_Toc14121)

[3.14.7 Pre-requisites and Dependency 73](#_Toc3075)

[3.15 Final Semester Result Report 73](#_Toc17092)

[3.15.1 Use Case Diagram 73](#_Toc24432)

[3.15.2 Design of Workflow 74](#_Toc22294)

[3.15.3 Validations 74](#_Toc31503)

[3.15.4 Field List: 75](#_Toc21140)

[3.15.5 Processes After Form Submission 76](#_Toc5419)

[3.15.6 Process Flow: 76](#_Toc30628)

[3.15.7 Pre-requisites and Dependency 76](#_Toc107)

[3.16 Course Wise Performance 76](#_Toc2222)

[3.16.1 Use Case Diagram 76](#_Toc11920)

[3.16.2 Design of Workflow 77](#_Toc19863)

[3.16.3 Validations 77](#_Toc6251)

[3.16.4 Field List: 78](#_Toc29291)

[3.16.5 Processes After Form Submission 78](#_Toc24167)

[3.16.6 Process Flow: 78](#_Toc28714)

[3.16.7 Pre-requisites and Dependency 79](#_Toc11945)

[3.17 Module Wise Performance 79](#_Toc30983)

[3.17.1 Use Case Diagram 79](#_Toc9418)

[3.17.2 Design of Workflow 80](#_Toc27552)

[3.17.3 Validations 80](#_Toc14513)

[3.17.4 Field List: 81](#_Toc11511)

[3.17.5 Processes After Form Submission 82](#_Toc15657)

[3.17.6 Process Flow: 82](#_Toc1845)

[3.17.7 Pre-requisites and Dependency 82](#_Toc13338)

[3.18 Cumulative Marksheet 82](#_Toc23420)

[3.18.1 Use Case Diagram 82](#_Toc11665)

[3.18.2 Design of Workflow 83](#_Toc10069)

[3.18.3 Validations 83](#_Toc20470)

[3.18.4 Field List: 84](#_Toc8825)

[3.18.5 Processes After Form Submission 86](#_Toc23341)

[3.18.6 Process Flow: 86](#_Toc8204)

[3.18.7 Pre-requisites and Dependency 86](#_Toc17773)

[3.19 Student Backpaper Tracking 87](#_Toc30338)

[3.19.1 Use Case Diagram 87](#_Toc18445)

[3.19.2 Design of Workflow 87](#_Toc3006)

[3.19.3 Validations 87](#_Toc9124)

[3.19.4 Field List: 88](#_Toc20277)

[3.19.5 Processes After Form Submission 89](#_Toc24102)

[3.19.6 Process Flow: 89](#_Toc31256)

[3.19.7 Pre-requisites and Dependency 89](#_Toc4836)

[4 Tools And Technologies 89](#_Toc28109)

[5 Non-Functional Requirements 90](#_Toc27546)

[5.1 Performance 90](#_Toc2886)

[5.2 Reliability 90](#_Toc21240)

[5.3 Usability 90](#_Toc29172)

[5.4 Compatibility 90](#_Toc13370)

[5.5 Security 90](#_Toc19302)

[5.6 Compliance 90](#_Toc30934)

[5.7 Documentation 90](#_Toc12505)

[5.8 Security: User access and authorization 90](#_Toc15433)

[6 Database Design 95](#_Toc25149)

[Prototype 96](#_Toc12162)

[7 Prototype 97](#_Toc643)

[7.1 Exam Type 97](#_Toc12419)

[7.2 Assessment Component 97](#_Toc27433)

[7.3 Grading Scale 98](#_Toc48)

[7.4 Exam Declaration 98](#_Toc31077)

[7.5 Module Wise Exam Group 98](#_Toc3934)

[7.6 Exam Evaluation Plan 99](#_Toc2391)

[7.7 Exam Paper Setting 100](#_Toc26025)

[7.8 Component Wise Evaluation 100](#_Toc14386)

[7.9 Component Wise Evaluation Tool 101](#_Toc30608)

[7.10 Component Wise Reevaluation 101](#_Toc3126)

[7.11 Component Wise Reevaluation Tool 102](#_Toc12015)

[7.12 Final Semester Result 102](#_Toc23026)

[7.13 Final Semester Result Tool 102](#_Toc16337)

[7.14 Cumulative MarkSheet Tool 103](#_Toc14380)

[7.15 Final Semester Result Report 103](#_Toc29988)

[7.16 Course Wise Performance 104](#_Toc8278)

[7.17 Module Wise Performance 104](#_Toc15330)

[7.18 Cumulative MarkSheet 104](#_Toc1927)

[7.19 Student BackPaper Tracking 105](#_Toc10408)

[8 Definitions and Acronyms 105](#_Toc23624)

[9 Deployment Description 106](#_Toc21198)

# Introduction

The World Skill Center (WSC) is a premier advanced skill training institute established by the Government of Odisha, Skill Development and Technical Education Department, through the Odisha Skill Development Authority (OSDA). The WSC will impart advanced skill training in eight trades from engineering and service sectors. WSC caters primarily to induct the best talent from ITIs and Polytechnics and training them to become globally employable in emerging areas such as "Industry 4.0". WSC is housed in a state-of-the-art, 18-storey, air-conditioned building with nearly half a million square feet of space in the heart of the capital city of Bhubaneswar.

With the implementation of campus management software for WSC the goal is to streamline the operations and functions of the campus by integrating various processes, such as admissions, course registration, academic progress tracking, and financial management, HRMS, Procurement and Inventory management, etc into a unified system. The implementation also aims at providing a user-friendly interface for all stakeholders, making it easier for them to access the necessary information and complete their tasks with ease. The modules to be covered during the implementation of the software include:

|  |  |
| --- | --- |
| Students Management Modules | Infrastructure Management |
| Academic System | Finance and Accounting System |
| Procurement & Inventory Management | Training and Placement |
| Human Resources Management System | Application Integration |

# 

## Background

The Web Based Campus Management Application at World Skill Centre (WSC) application is required by WSC for the smooth operation of all departments / support functions with on-line delivery of services to all stakeholders.

The project aims to create a mechanism to provide the basis for evolution of an IT enabled state of the art workflow automation system in a planned manner.

## Scope and Purpose of the document

The design documents track the necessary information required to effectively define architecture and system design in order to give the development team guidance on the architecture of the system to be developed. Its intended audience is the project manager, project team, and development team. Some portions of this document, such as the user interface (UI), may be shared with the client/user, and other stakeholders whose input/approval into the UI is needed.

This document covers all the functional requirements of the **Examination module** of ERP Product. The ERP Examination module helps in organizing the entire education set-up, such as Student Database, Fee Structure, Trainer Information Assessment and Result Declaration and Cumulative Marksheet, Certificate Generation, etc.

The scope of Education module

* Courses and Subjects, Timetable preparation and management 
* Class occurrence monitoring
* Class Attendance of students (Face recognition or any other automated mode).  
* Results Analysis after declaration of results by WSC 
* Students Feedback mechanism on quality of teaching learning  Internship, Training, Apprentice 
* Re-Admission Process after discontinuing in studies 
* Students Back paper tracking etc.
* Student Profile :Personal Information, Contact Details, Academic Details 
* Issue of Identity Card. 
* Student Profile: Personal Information, Contact Details, Academic Details.
* Re-Admission Process after discontinuing in studies
* Issue of Identity Card
* Faculty Workload / Lesson Plan
* Rewards and achievements.
* Biometric Attendance (Face recognition) 
* Hostel Management (Hostel seat allotment, attendance, fees etc) 
* Scholarships. 
* Rewards and achievements. 
* Students’ Evaluation/remarks of Faculty. 
* Student Exams records of all 6 semesters 
* Parent Enquiry/Alert/Messaging Management. 
* Messaging System 
* Student Grievance System etc.

Following functionalities are covered in this document:

* **Results Analysis after declaration of results by WSC **
* **Students Back paper tracking etc.**
* **Students Exams records **
* **Alert/Messaging Management.**

## Assumptions

The assumptions are listed as follows:

* The required hardware and software requirements will be provided
* Data loading to be handled by WSC
* All external entities will provide their interface for application integration

## Dependencies

The ERP system will be dependant on external interfaces for integration services. The following are a few external interfaces:

* Payment Gateway
* Communication channels such as – Email, SMS, WhatsApp
* Microsoft 365
* Biometric System
* Website

## Current IT Environment

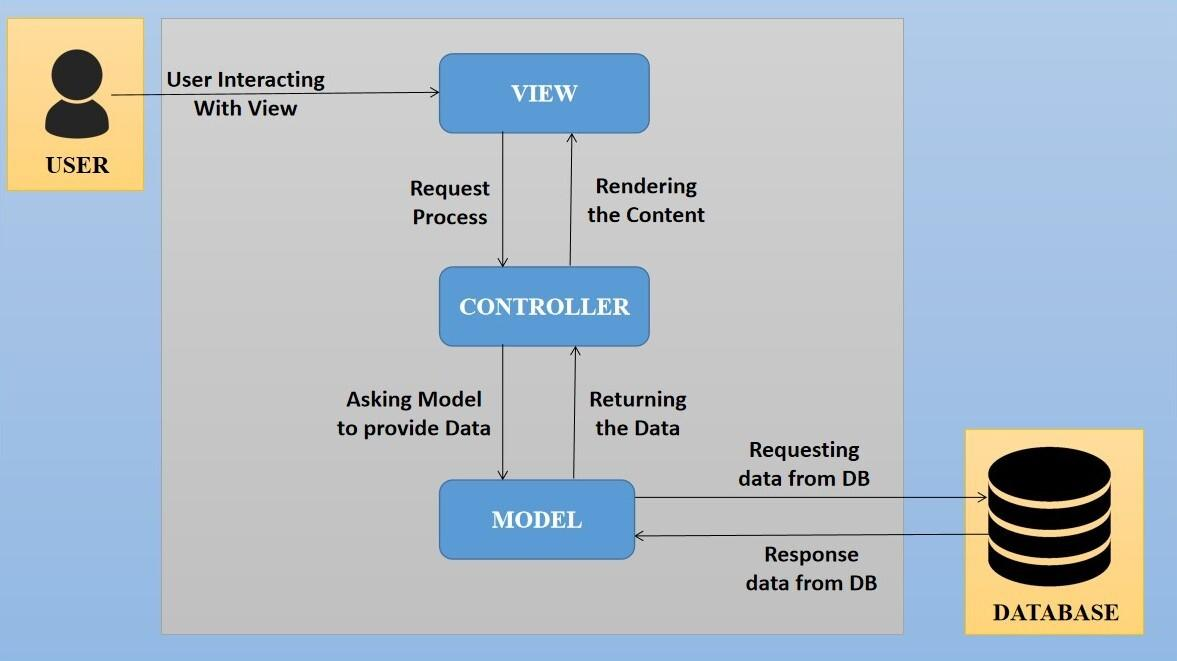
|  |  |
| --- | --- |
| **Hardware Requirements** | **Software Requirements** |
| Processor: Intel CPU with at least 8 cores | Operating System: Ubuntu 22.04 LTS |
| RAM: 16 GB to 32 GB | Web Server: Nginx |
| Storage: A minimum of 160 GB SSD | Database server: MariaDB |
| Network: 1 Gbps | Python: 3.10 or later |

## Document Structure

The following sections are part of this document :

* Business goals, objectives and requirements
* Understand business context and interactions
* Conceptual design
* Overall system context
* Understand use-case and scenarios
* Define implementation

## Required System Architecture



The proposed design is an architectural pattern that separates an application into three main logical

components: The **Model**, the **View** and the **Controller**. Each of these components are built to handle

specific development aspects of the application like:

* The model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data
* The View component is used for all the UI logic of the application. It generates a user interface for the user
* The controller is the component that enables the interconnection between the views and the model, so it acts as an intermediary

## Interoperability Framework

Frappe is a full stack, batteries-included, web framework written in Python and JavaScript with MariaDB as the database. It is pretty generic and can be used to build database-driven apps with an elegant and consistent UI.



## Technical Decision Summary



## Architectural Risks & Mitigation

|  |  |
| --- | --- |
| **Risks** | **Recommendation** |
| Inability to provide the required hardware resources (Server) for installation of the ERP product | The issue should be brought up during cadence meetings, and the OCAC/WSC IT team should assure timely availability of the required server |
| Unavailability of API for third party integration | The project timeline schedule needs to be shared with third parties to avoid the delay in integration. Also, inclusion of external teams to meet timelines |
| The number of users accessing the application exceeds its limit (as specified in the proposal document), leading to performance issues | WSC leadership team to ensure the scalability of the hardware / software resources |
| End Users of the application unavailable for the training to be provided by SOUL | SOUL to inform WSC Stakeholders on training plan in advance and engage early for training |

## Reference

|  |  |
| --- | --- |
| ***Sl No.*** | ***Document Name*** |
| 1 | SRS for Examination Module |
|  |  |

# Enterprise Architecture Framework (Business Architecture)

## Business Drivers

The following points motivate the business efficacy of Campus Management Software at World Skill Center (WSC) system:

* Integrate the various functions such as Admission, Academics, Examination, Training & Placement, HRMS, Procurement, Finance into a single platform which can be customized as per user requirements
* Implement a common framework for the system and sub systems
* Provide a web based application for the users to interact with the system with role-based access and a consistent look and feel

## Business Concerns

Campus Management Software At World Skill Center application is designed to address the following concerns:

Student Management: To help the students / trainers manage the students admission activities, starting from initial communication to course enrollment

Financial Management: Managing finances is a fundamental concern for businesses. This includes budgeting, cash flow management, ledger maintenance, balance sheet,etc

Employee Engagement: Attracting, retaining, and engaging skilled and motivated employees is critical. Human resources concerns also include training, performance management

Procurement management: The source-to-settle process. It encompasses the evaluation, selection, and creation of formal contractual agreements as well as managing the company's ongoing supplier relationships

## Business Goals

The goal is to implement a web based Campus Management Application System for efficient internal functioning of the World Skill Center (WSC) with on-line delivery of services to each stakeholder of WSC supported by a suitable, robust, secure and reliable system

## Business Value Chain

The ERP system's business value chain typically encompasses the following key components:

Procurement Management: This includes the processes related to sourcing and acquiring raw materials, goods, or services required for the organization's operations. The ERP system streamlines procurement activities, such as supplier management, purchase requisitions, purchase orders, and inventory management, leading to cost savings and better supply chain management.

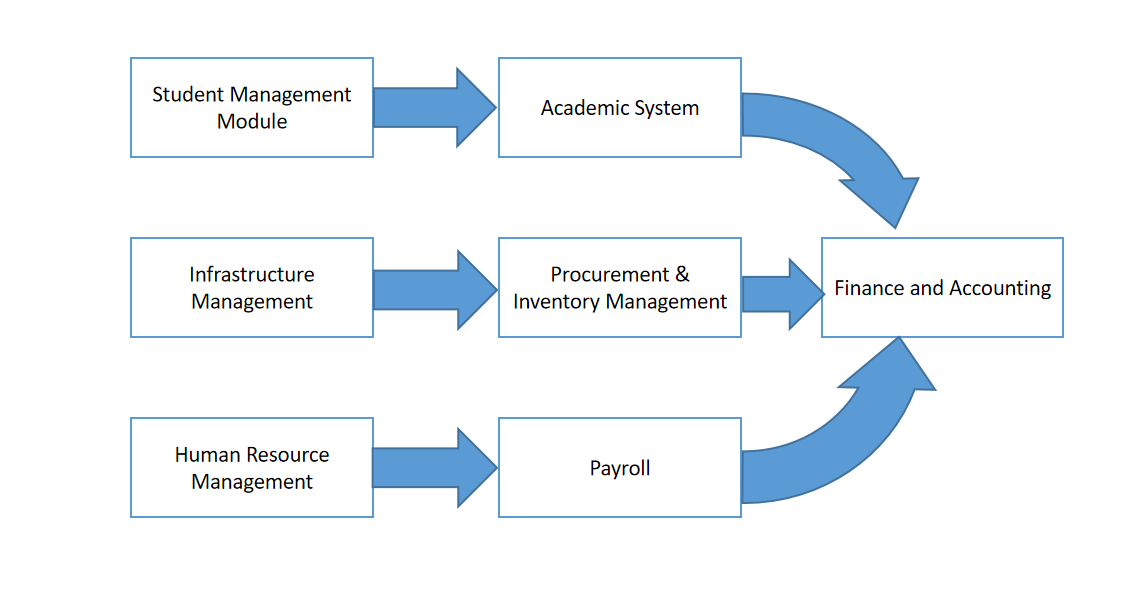
Production/Operations Management: The ERP system facilitates the planning, scheduling, and execution of production processes. It optimizes resource allocation, tracks work progress, manages bills of materials, and monitors production costs to enhance overall operational efficiency and quality.

Inventory Management: The ERP system enables real-time tracking of inventory levels, stock movements, and stock outs. It helps in maintaining optimal inventory levels, reducing carrying costs, and ensuring timely availability of products.

Financial Management: ERP systems centralize financial data and automate accounting, financial reporting, budgeting, and financial analysis. This streamlines financial processes, enhances accuracy, and provides management with a clear financial overview for better decision-making.

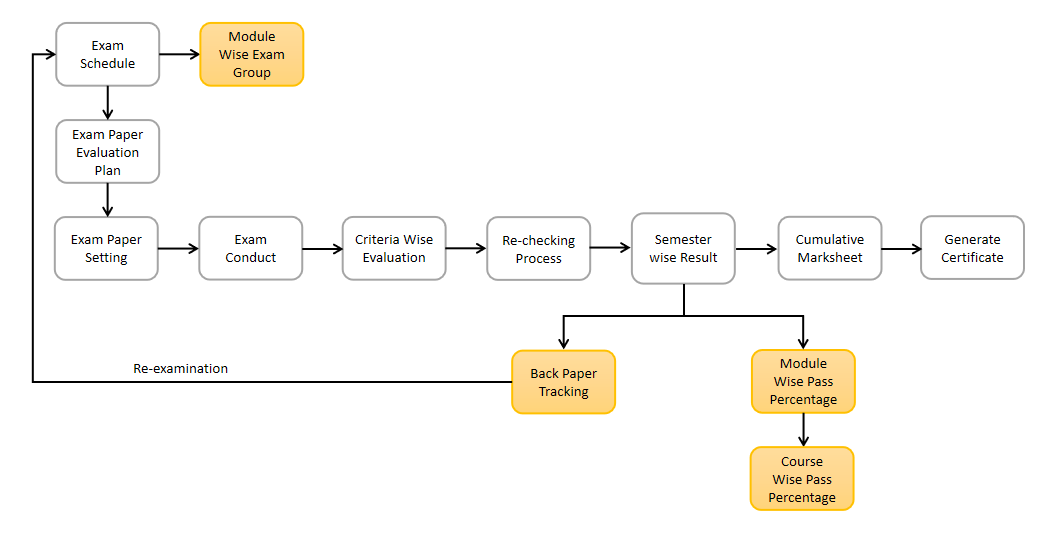
Human Resources Management: ERP systems handle various HR functions, including payroll, employee records, recruitment, performance management, and training. This leads to streamlined HR processes, improved workforce management, and enhanced employee satisfaction.

## Business Context Diagram

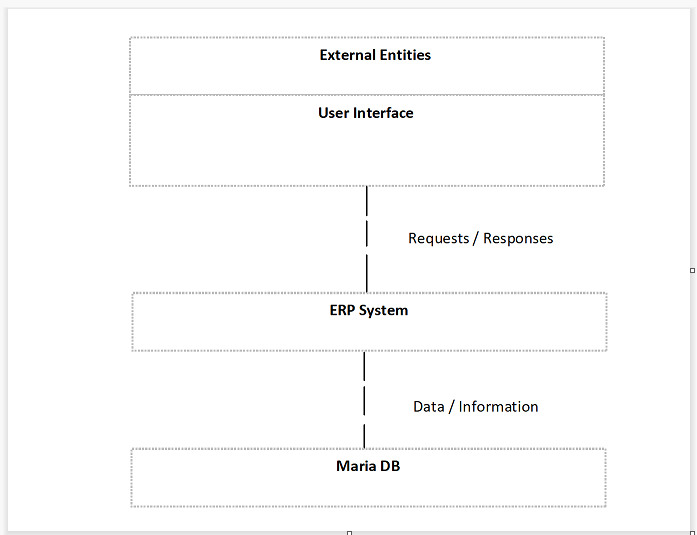


## Business Process Flow

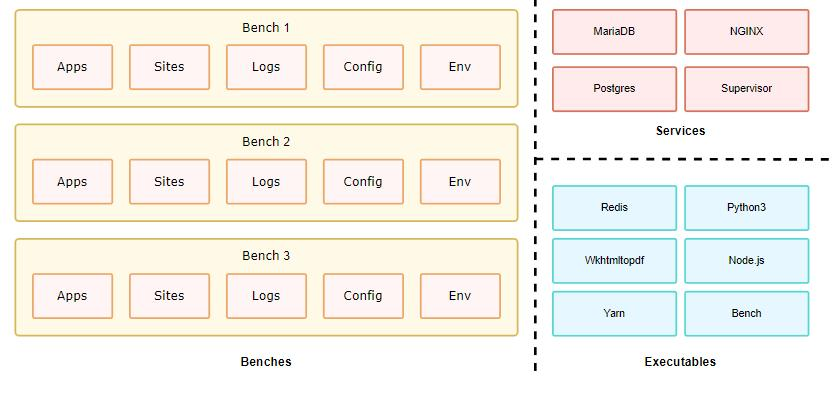
### Examination Process Flow



### System Context Diagram



## Conceptual Architecture



A single bench can host multiple combinations of sites and apps. For the most part, having a single bench works for hosting hundreds of sites that depend on the same versions of said app, given you've scaled up the workers. You can host multiple versions of the application on the same server by creating multiple benches parallelly. The following diagram hints to the system dependencies and how they are used.

# Architecture Views

## Exam Type

The Exam Type allows user to create and set values for a particular Exam Declaration which shall be beneficial to determine the type of examinations that are being conducted and can be evaluated on the basis of it. For example, an Exam Declaration can be classified as Mid Term Examination or End Term Examination.

### Use Case Diagram

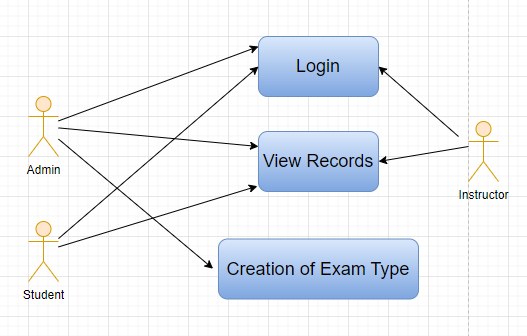


Figure : Exam Type Use case diagram

### 

### Design of Workflow

* Design workflow is not applicable to this screen as records in this are created by only one role and are not sent to users with other roles for approval or modification

### Validations

The data entered on this page will undergo a 2 step validation (i.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Exam Type screen.

The following validation is done on client side:

* Mandatory checks for fields : None
* Linked Fields : None

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

There are no server side validations for the Exam Type screen.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Type | Text Field | User Input |  |  |  |
| 2 | Description | Long Text Field | User Input |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form is not submittable.

### Process Flow:

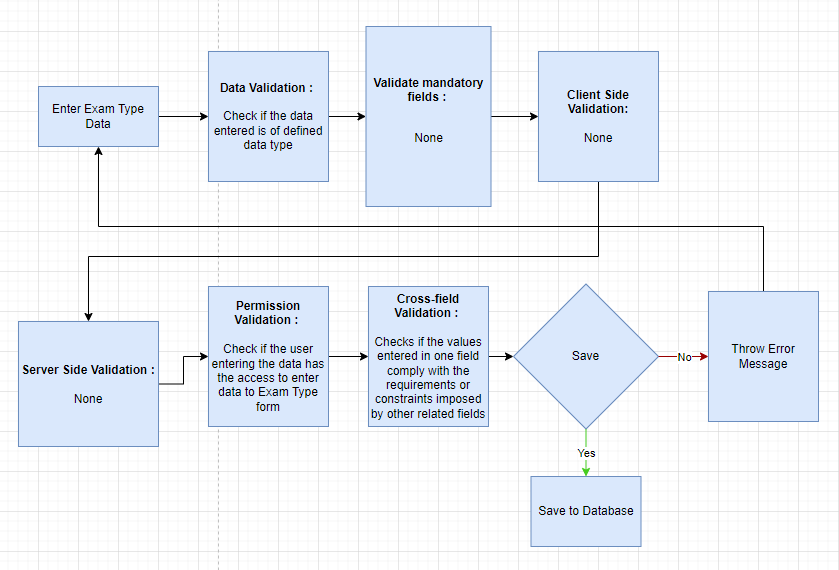


Figure : Exam Type Process Flow

### Pre-requisites and Dependency

None

## Assessment Component

The Assessment Component is various components of Continuous Evaluation on the basis of which evaluation shall be conducted. Also, multiple Assessment Criteria can be grouped together under Assessment Criteria Group as well as Maximum Credit can be allotted to them. It can also be linked with Exam Declaration if required. For example, Assessment Criteria can be specified as Mid Term, Practical, End Term, Quiz, Viva, etc.

### Use Case Diagram

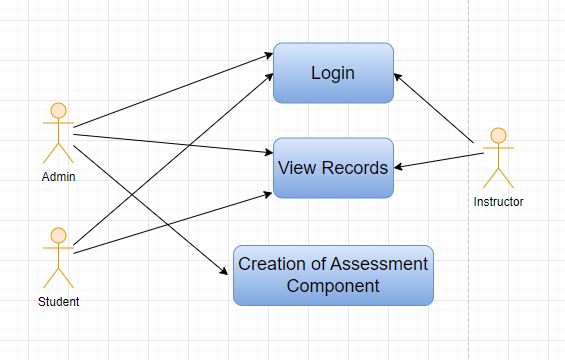


Figure : Assessment Component Use case diagram

### Design of Workflow

**Design of Workflow**

* Design workflow is not applicable to this screen as records are not sent to users with other roles for approval or modification

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Class Scheduling Tool screen.

The following validation is done on client side:

Mandatory checks for fields : Assessment Criteria, Exam Type

Linked Fields : The Component Type field is linked with Exam Type screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Assessment Criteria | Text Field | User Input | Yes |  | (R)Assessment Component |
| 2 | Assessment Criteria Group | Link Field | Fetched from Assessment Criteria Group master screen |  |  | Hidden |
| 3 | Maximum Credit | Float | User Input |  |  | (D) |
| 4 | Depends on Exam Declaration | Check Box | User Input |  |  | (D) |
| 5 | Component Type | Link Field | Fetched from Exam Type master screen | Yes |  |  |

### Processes After Form Submission

* This section is not applicable since the form is not submittable.

### Process Flow:

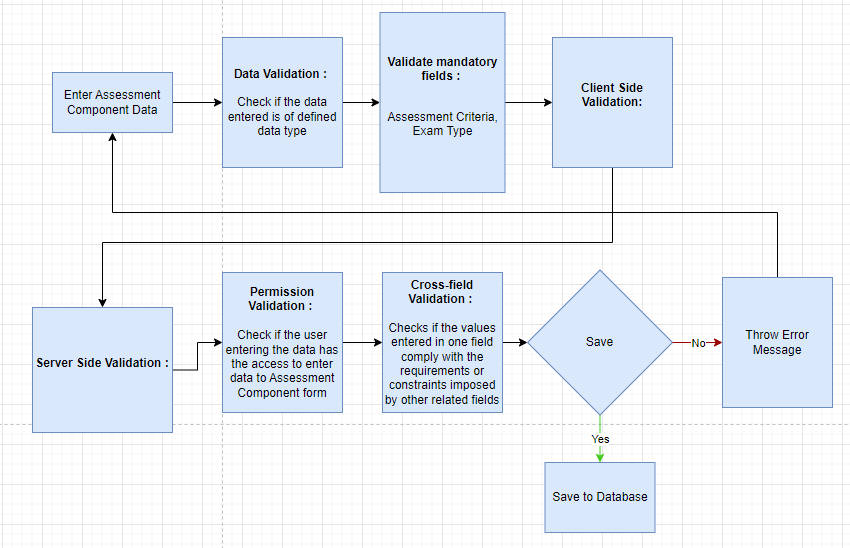


Figure : Assessment Component Process Flow

### Pre-requisites and Dependency

* Exam Type

## Grading Scale

Under Grading Scale, user can define the threshold for the different grades obtained by the students, based on their scores in the assessment. For example, students obtaining a score of 90% and above would be graded as A+, students obtaining a score of 80% and above would be graded A- and so on.

### Use Case Diagram

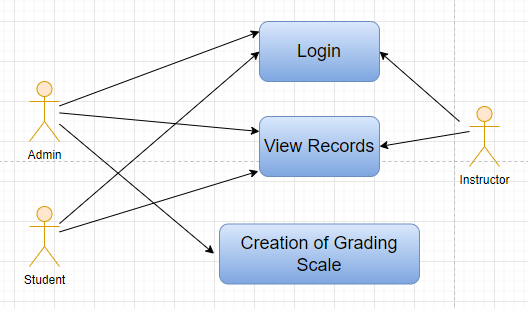


Figure : Grading Scale Use case diagram

### Design of Workflow

* Design workflow is not applicable to this screen as records in this screen are not sent to users with other roles for approval or modification

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Grading Scale screen.

The following validation is done on client side:

* Mandatory checks for fields : Grading Scale Name, Grade Code, Grade Point, Threshold
* Linked Fields : None

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* def validate(self): The purpose of this code is to validate a collection of intervals and their associated threshold percentages, ensuring that duplicate threshold percentages are not allowed and that a grade is defined for the 0% threshold if it exists.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Grading Scale Name | Text Field | User Input | Yes |  |  |
| 2 | **Grading Scale Interval** | Table | User Input |  | These table is describe below |  |
| 3 | Description | Long Text Field | User Input |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Grading Scale Interval** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Grade Code | Text Field | User Input | Yes |  |  |
| 2 | Grade Point | Text Field | User Input | Yes |  |  |
| 3 | Threshold | Float | User Input | Yes |  |  |
| 4 | Result | Drop Down | PASS  FAIL |  |  |  |
| 5 | Grade Description | Long Text Field | User Input |  |  |  |

### 

### Processes After Form Submission

* This section is not applicable there is no workflow on submission.

### Process Flow:

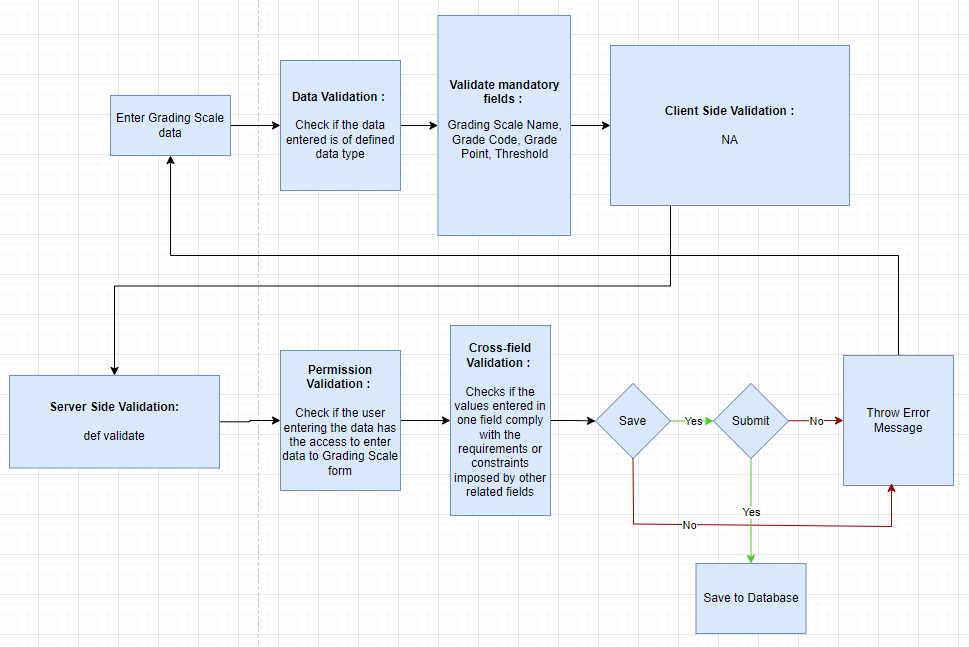


Figure : Grading Scale Process Flow

### Pre-requisites and Dependency

* None

## Exam Declaration

The Exam Declaration will allow user to declare an exam for various programs in accordance with the ‘Academic Calendar’ template which was used during the start of each session.

### Use Case Diagram

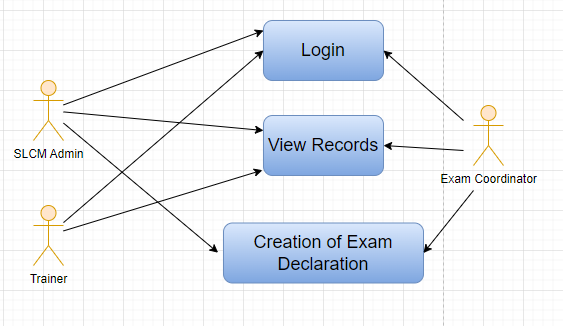


Figure : Exam Declaration Use case diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Exam Declaration screen.

The following validation is done on client side:

* Mandatory checks for fields : Program Grade, Exam Program, Semesters, Exam Type, Assessment Component, Academic Year, Academic Term, Courses, Examination Start Date, Examination End Date, From Time, To Time
* Linked Fields : The Program Grade field is linked with the Program Grade Screen. The Exam Program field is linked with the Program Screen. The Student field is linked with the Student Screen. The Academic Year field is linked with the Academic Year Screen. The Academic Term field is linked with the Academic Term Screen. The Semester field is linked with the Semester Screen. The Courses field is linked with the Courses Screen. The Marker field is linked with the Instructor Screen. The Checker field is linked with the Instructor Screen. The Student field is linked with the Student Screen. The Student Category field is linked with the Student Category Screen. The Fee Structure field is linked with the Fee Structure Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* date\_validation: This method validates various date-related conditions associated with the exam declaration, such as ensuring that certain dates are in the correct order and meet specific criteria.
* calculate\_total\_hours: This method calculates the total duration in hours for courses offered within the exam declaration. It calculates the difference between "to\_time" and "from\_time" for each course.
* validate\_courses: This method validates the courses offered within the exam declaration, checking if the specified courses belong to the associated semesters and ensuring that time intervals are valid.
* validate\_fee\_structure: This method validates the fee structure associated with the exam declaration, confirming that the provided fee structure belongs to the exam program and is of the "Exam Fees" type.
* sort\_by\_date: This method sorts the courses offered within the exam declaration by their examination dates.
* get\_courses: This method retrieves a list of courses based on the academic year and other filters, ensuring they belong to the specified program and are relevant for the exam declaration.
* validate: This method calls various validation methods to ensure the integrity and correctness of the exam declaration data. It also checks for duplicate rows and sorts the courses by date.
* set\_user\_permission: This method sets user permissions for instructors based on the exam declaration.
* on\_update\_after\_submit: This method is triggered after the document is updated and submitted. It sorts the courses by date and updates module disabled status.
* on\_trash: This method is triggered when the document is deleted and is used to delete associated user permissions.
* on\_submit: This method is triggered when the document is submitted. It sets user permissions for instructors, triggers notifications, and potentially creates assessment results.
* on\_cancel: This method is triggered when the document is canceled and could perform actions such as canceling fees associated with the exam declaration.
* validate\_assessment\_plan: This method validates the selected course assessment plan against program and academic year.
* module\_disabled\_update: This method updates the disabled status of module-wise exam groups associated with the exam declaration.
* get\_fee\_structure: This function retrieves valid fee structures based on the exam program.
* filter\_courses: This function filters courses based on the program.
* make\_student\_admit\_card: This function creates student admit cards based on the exam declaration data.
* make\_exam\_assessment\_result and create\_conduct\_certificate: These functions work together to create fee records for students based on the exam declaration, essentially generating assessment results for the exam.
* cancel\_fees: This function cancels fees associated with the exam declaration.
* valid\_exam\_declaration\_no: This function searches for valid exam declaration numbers based on provided filters and conditions.

Overall, the Exam Declaration class encapsulates functionality for managing exam declarations, ensuring the correctness of associated data, generating student admit cards, and handling assessment result creation.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
|  | Program Grade | Link Field | Link Field to Program Grades master screen | Yes |  | 1. Course Grade |
|  | Exam Program | Link Field | Program will be fetched based from selected “Program Grade” | Yes | Link Field to Program Screen | (R) Course Name |
|  | **Semesters** | Table | Only those semesters list will come which belongs to the entered Exam Program | Yes | Description of this table is given below |  |
|  | Class | Link Field | Linked Field to Class Master Screen |  | Options:  Class A, Class B, Class C, Class D |  |
|  | Exam Type | Link Field | Link to Exam Type master screen | Yes | Eg. In-Module, End Module |  |
|  | Assessment Component | Link Field | 1. Link to Assessment Criteria Screen 2. Data will fetch based upon the exam type | Yes | Eg. Phase Test 1  Phase Test 2  Practical Assignment 1,  Practical Assignment 2, etc. |  |
|  | Exam Name | Text Field | User Input | Yes |  |  |
|  | Disabled | Checkbox | User Input |  |  |  |
|  | Is Application Required | Checkbox | User Input |  |  | (D) |
|  | Academic Year | Link Field | Fetched from Academic Year master screen | Yes |  |  |
|  | Academic Term | Link Field | Fetched from Academic Term master screen | Yes |  |  |
|  | Exam Category | Drop down | Regular/Re-Test |  |  |  |
|  | Application Form Start Date | Date Picker | Display and Mandatory if “Is Application Required” is checked |  |  | (D) |
|  | Application Form End Date | Date Picker | 1. Display and Mandatory if “Is Application Required” is checked 2. Should be greater than Start Date |  |  | (D) |
|  | Exam Start Date | Date Picker | User Input | Yes |  |  |
|  | Exam End Date | Date Picker | Should be greater than Start Date | Yes |  |  |
|  | Admit Card Issue Date | Date Picker | User Input |  |  |  |
|  | Block List Display Date | Date Picker | Date should be before “Exam Start Date” |  |  |  |
|  | Get Courses | Button | Courses will be fetched based on Selected “Exam Program” and “Semester” |  |  | (R) Get Modules |
|  | **Courses Offered** | Table | Only those courses will come which belongs to the entered semester | Yes | Description of this table is given below | (R) Modules Offered |
|  | Minimum Attendance Criteria | Float | User Input | Yes |  |  |
|  | Get Students | Button |  |  |  |  |
|  | Total Enrolled Student | Number | Calculated total number of Students fetched based on “Get Students” Button |  |  |  |
|  | **Students** | Table | Students will be fetched based on selected Course, Semester, Class, Academic Year, Academic Term |  | Description of this table is given below |  |
|  | Fees Status | Drop down | Full Paid/  More than 50%/  Allow all | Yes |  |  |
|  | Exam Fees Applicable | Drop down | By default It will be NO, if Fees Status is **Allow All** |  | YES/NO |  |
|  | **Fee Structure** | Table | If “Exam Fees Applicable” is “Yes” then the table will show |  | This table is described below |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Semesters** | | | | | | |
| ID | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Semester | Link Field | Fetched from Semester master screen and filtered based on Selected “Exam Program” | Yes |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Courses Offered** | | | | | |
| ID | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Courses | Link Field | Auto fetch based on “Get Courses” Button | Yes |  | (R) Modules |
| 2 | Course Name | Text field | Auto fetch based on “Get Courses” Button |  |  | (R) Modules Name |
| 3 | Course Code | Text field | Auto fetch based on “Get Courses” Button |  |  | (R) Modules Code |
| 4 | Semester | Link Field | Auto fetch based on “Get Courses” Button |  |  |  |
| 5 | Examination Start Date | Date Picker | Individual Module Examination Start date need to be between the exam Start Date and End Date | Yes | User Input |  |
| 6 | Examination End Date | Date Picker | Individual Module Examination End date need to be between the exam Start Date and End Date | Yes | User Input |  |
| 7 | From Time | Time | User Input | Yes |  |  |
| 8 | To Time | Time | User Input | Yes |  |  |
| 9 | Total Duration (in Hours) | Time | Calculated based on “From Time” & “To Time” |  |  |  |
| 10 | Marker | Link Field | Link Field to Instructor Screen |  |  |  |
| 11 | Checker | Link Field | Link Field to Instructor Screen |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Students** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Text Field | Auto fetch based on “Get student” Button |  |  |  |
| 2 | Student Name | Text Field | Auto fetch based on “Get student” Button |  |  |  |
| 3 | Roll No | Text Field | Auto fetch based on “Get student” Button |  |  |  |
| 4 | Registration Number | Text Field | Auto fetch based on “Get student” Button |  |  | (D) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Fee Structure** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student Category | Link Field | Fetched from Student Category master screen | Yes |  |  |
| 2 | Fee Structure | Link Field | Fetched from Fee Structure master screen |  |  |  |
| 3 | Amount | Currency Field | Auto fetch based on “Fee Structure” |  |  |  |
| 4 | Due Date | Date Picker |  | Yes |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no workflow after submission.

### Process Flow:



Figure : Exam Declaration Process Flow

### Pre-requisites and Dependency

* Academic Calendar
* Program
* Academic Year

## Module Wise Exam Group

With the help of Exam Process Screen, a user can grouped students to giving the exams on the basis of batch, room capacity, lab examination, etc.

### Use Case Diagram

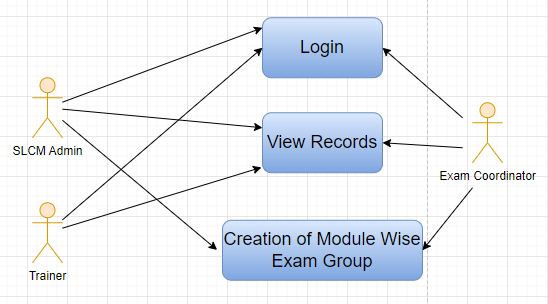


Figure : Module Wise Exam Group Use case diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Module Wise Exam Group screen.

The following validation is done on client side:

* Mandatory checks for fields : Exam Declaration Id, Modules Id, Marker, Checker, Course Manager, Invigilator Details Table, Group Name
* Linked Fields : The Exam Declaration Id field is linked with the Exam Declaration Screen. The Academic Year field is linked with the Academic Year Screen. The Academic Term field is linked with the Academic Term Screen. The Department field is linked with the Department Screen. The Exam Schedule Id field is linked with the Exam Declaration Screen. The Semester field is linked with the Semester Screen. The Modules Id field is linked with the Course Screen. The Checker field is linked with the Trainer Screen. The Checker field is linked with the Trainer Screen. The Course Manager field is linked with the Trainer Screen. The Student field is linked with the Student Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* validate: This function is called during the validation process of the ModuleWiseExamGroup document. It performs several checks and validations to ensure the integrity and correctness of the data in the document. It calls other functions like duplicate\_validation, over\_lapping\_of\_scheduling, and group\_validation to perform specific validations.
* on\_submit: This function is triggered when the ModuleWiseExamGroup document is submitted. It performs validations and operations specific to the submission process. It calls the functions group\_validation and date\_time\_mandatory to ensure that required fields are filled before submission and then sends emails to students and trainers based on certain conditions.
* calculate\_total\_hours: This method calculates the total duration in hours for each scheduling entry in the scheduling\_group\_exam field of the ModuleWiseExamGroup document. It calculates the difference between the to\_time and from\_time fields of each entry.
* over\_lapping\_of\_scheduling: This function checks for any overlap in scheduling of exams for students within the same ModuleWiseExamGroup. It retrieves data related to student exam schedules and compares their dates and times to identify overlapping exams, and if found, it raises an error with the relevant details.
* duplicate\_validation: This function checks for duplicate entries of the ModuleWiseExamGroup document based on the exam\_declaration\_id and modules\_id fields. If duplicate entries are found, it raises an error indicating that the module-wise exam group has already been declared.
* time\_mandatory: This function ensures that the from\_time and to\_time fields are filled for each scheduling entry in the scheduling\_group\_exam field of the ModuleWiseExamGroup document. If these fields are not filled, it raises an error.
* validate\_time: This function validates the from\_time and to\_time fields for each scheduling entry in the scheduling\_group\_exam field of the ModuleWiseExamGroup document. It ensures that the from\_time is not greater than the to\_time and raises an error if this condition is violated.
* date\_validation: This function validates the examination date for each scheduling entry in the scheduling\_group\_exam field of the ModuleWiseExamGroup document. It checks whether the examination date is within the range of the module\_exam\_start\_date and module\_exam\_end\_date fields and raises an error if it's not.
* date\_time\_mandatory: This function validates the presence of examination date, from\_time, and to\_time for each scheduling entry in the scheduling\_group\_exam field of the ModuleWiseExamGroup document. If any of these fields are missing, it raises an error.
* group\_validation: This function checks the presence of a group\_name for each student in the student\_list field of the ModuleWiseExamGroup document. It is called both during validation and submission processes, and if a group name is missing, it raises a validation message or an error, respectively.
* filter\_group: This function populates the scheduling\_group\_exam field of the ModuleWiseExamGroup document with group names. It checks for existing group names and either appends them or creates new entries in the scheduling groups field.
* valid\_module\_as\_exam\_declation: This function provides a list of exam courses based on the given parent exam declaration ID. It is used to populate the list of valid modules that can be selected as part of the exam declaration.
* get\_semester: This function retrieves semester information related to a given exam declaration ID. It is used to set the semester value based on the selected exam declaration.
* module\_start\_date: This function retrieves module-specific start dates, end dates, attendance criteria, and minimum attendance criteria based on the selected modules, exam declaration, and academic term. These values are used for validation and calculations.
* get\_student: This function retrieves a list of students based on various filters, such as academic term, programs, class, attendance criteria, start and end dates, module ID, semester, exam category, exam schedule ID, course type, and assessment component. It fetches student data and calculates attendance percentages for further processing.
* get\_program\_enrollment: This function retrieves the list of enrolled students for a given academic term and program, with optional filtering by class. It is used to populate the student list for regular exams.
* get\_program\_enrollment\_fail: This function retrieves the list of students who failed a specific assessment component in a course. It fetches students who qualify as re-exam candidates based on various criteria, including the assessment component and program grade.

Each of these functions and methods plays a role in validating data, ensuring proper scheduling, calculating relevant duration, and preparing data for email notifications and exam qualification assessments within the context of the Module Wise Exam Group document.

**Notification**

Mail will be triggered to the Trainers and Students who belongs to that exam declaration.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Exam Declaration Id | Link | Links to Exam Declaration Screen | Yes |  |  |
| 2 | Disabled | Check |  |  |  |  |
| 3 | Exam Name | Data |  |  |  |  |
| 4 | Academic Year | Link | Links to Academic Year Screen |  |  |  |
| 5 | Academic Term | Link | Links to Academic Term Screen |  |  |  |
| 6 | Exam Category | Data |  |  |  |  |
| 7 | Exam End Date | Data |  |  |  |  |
| 8 | Assessment Component | Data |  |  |  |  |
| 9 | Exam Type | Data |  |  |  |  |
| 10 | Department | Link | Links to Department Screen |  |  |  |
| 11 | Course Type | Data |  |  |  |  |
| 12 | Exam Schedule Id | Link | Links to Exam Declaration Screen |  |  |  |
| 13 | Course Name | Data |  |  |  |  |
| 14 | Class | Data |  |  |  |  |
| 15 | Semester | Link | Links to Semester Screen |  |  |  |
| 16 | Exam Start Date | Data |  |  |  |  |
| 17 | Modules Id | Link | Links to Course Screen | Yes |  |  |
| 18 | Modules Name | Data |  |  |  |  |
| 19 | Module Code | Data |  |  |  |  |
| 20 | Module Exam Start Date | Data |  |  |  |  |
| 21 | Module Exam End Date | Data |  |  |  |  |
| 22 | Attendance Criteria | Data |  |  |  |  |
| 23 | Qualification Attendance Percentage | Float |  |  |  |  |
| 24 | Start date of Attendence Duration | Date |  |  |  |  |
| 25 | End date of Attendence Duration | Date |  |  |  |  |
| 26 | Marker | Link | Links to Trainer Screen | Yes |  |  |
| 27 | Checker | Link | Links to Trainer Screen | Yes |  |  |
| 28 | Course Manager | Link | Links to Trainer Screen | Yes |  |  |
| 29 | Marker Name | Data |  |  |  |  |
| 30 | Checker Name | Data |  |  |  |  |
| 31 | Course Manager Name | Data |  |  |  |  |
| 32 | Invigilator Details Table | Table |  | Yes |  |  |
| 33 | Get Student | Button |  |  |  |  |
| 34 | Total Enrolled Student | Data |  |  |  |  |
| 35 | Student List | Table |  | Yes |  |  |
| 36 | Scheduling Group Exam | Table |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Invigilator Details** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Trainer | Link | Links to Trainer Screen | Yes |  |  |
| 2 | Trainer Name | Data |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Module Wise Exam Student** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student No | Link | Links to Student Screen |  |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll No | Data |  |  |  |  |
| 4 | Permanent Registration No | Data |  |  |  |  |
| 5 | Total no of Classes Scheduled | Float |  |  |  |  |
| 6 | Total No of Class Attended By the student | Float |  |  |  |  |
| 7 | Attendance Percentage | Percent |  |  |  |  |
| 8 | Elegibility Status | Select |  |  |  |  |
| 9 | Examination Qualification Approval | Check |  |  |  |  |
| 10 | Group Name | Select |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student Group Exam Scheduling** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Group Name | Data |  | Yes |  |  |
| 2 | Examination Date | Date |  |  |  |  |
| 3 | From Time | Time |  |  |  |  |
| 4 | To Time | Time |  |  |  |  |
| 5 | Total Duration (in Hours) | Time |  |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

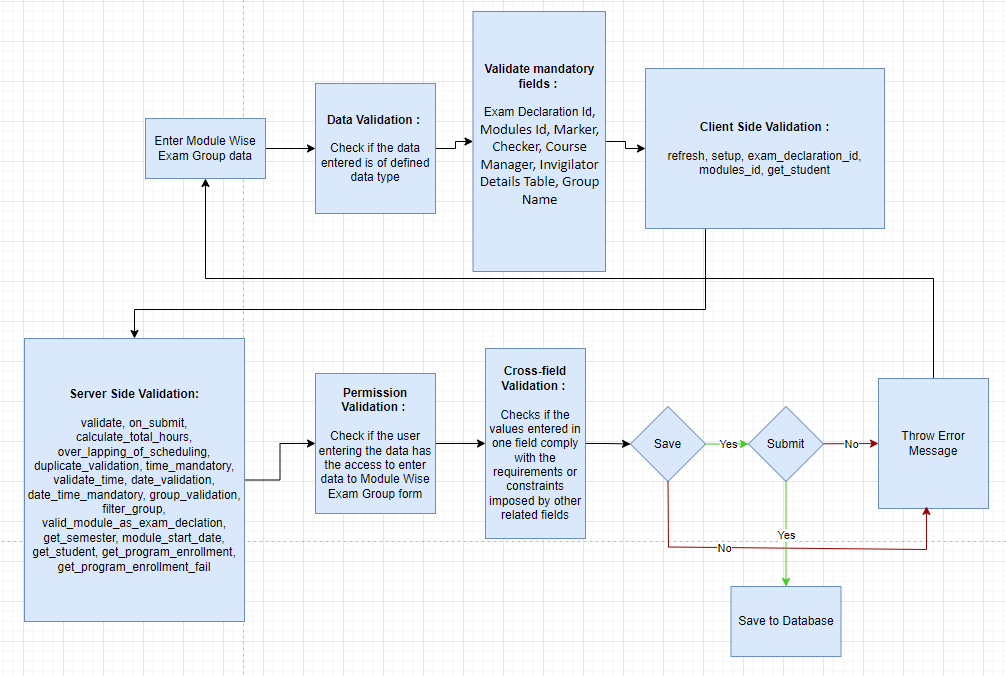


Figure : Module Wise Exam Group Process Flow

### Pre-requisites and Dependency

* Exam Declaration

## Exam Evaluation Plan

An Exam Evaluation Plan is a schedule to conduct the examination/assessment of a particular course for a group of students studying that course in an on-going academic term.

### Use Case Diagram

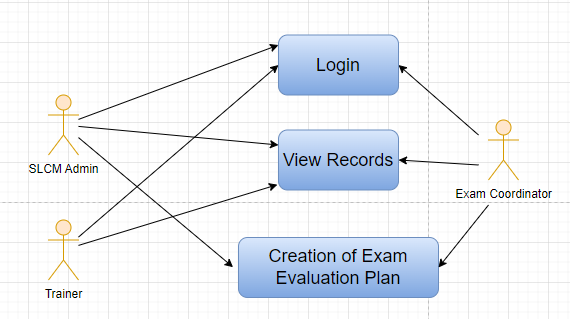


Figure : Exam Evaluation Plan Use Case Diagram

### Design of Workflow

* None

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Exam Evaluation Plan screen.

The following validation is done on client side:

* Mandatory checks for fields : Exam Declaration, Grading Scale, Exam Coordinator, Assessment Criteria, Programs, Semester, Academic Year, Academic Term, Exam Coordinator Name, Exam Assessment Plan Item, Paper Setting Start Date, Paper Setting End Date
* Linked Fields : The Exam Declaration field is linked with the Exam Declaration master screen.The Grading Scale field is linked with the Grading Scale master screen.The Exam Coordinator field is linked with the Trainer Screen.The Assessment Criteria field is linked with the Assessment Criteria master screen.The Programs field is auto-fetched based on the Program Screen.The Academic Year field is auto-fetched based on the Academic Year Screen.The Academic Term field is auto-fetched based on the Academic Term screen.The Exam Coordinator Name field is linked with the Trainer Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* validate: This method is called during document validation and is used to perform various validation checks on the Exam Assessment Plan document, including checking for duplicate rows in the "examiners\_list" and "moderator\_list" tables.
* on\_submit: This method is triggered when the Exam Assessment Plan document is submitted. It calls specific functions to handle notifications and actions related to paper setters and moderators.
* on\_update\_after\_submit: This method is called after the document is updated and submitted. It updates the start and end dates of pending Exam Paper Setting documents associated with this assessment plan.
* create\_exam\_paper\_setter: This function is called through a whitelist and initiates the creation of Exam Paper Setting documents based on the examiners\_list.
* validate\_programs: Validates whether the selected programs belong to the chosen exam declaration.
* validate\_semester: Validates whether the selected semester belongs to the chosen exam declaration.
* validate\_exam\_declaration: Validates the selected exam declaration against a list of valid exam declarations.
* validate\_dates: Performs date validations, checking if the paper setting dates are appropriate and ensuring they are not smaller than exam declaration dates.
* validate\_course: Validates selected courses against valid courses from the exam declaration and checks for duplicate courses in the assessment plan items.
* get\_sem: A whitelisted function that returns a list of semesters based on the chosen exam declaration.
* filter\_paper\_setter: A whitelisted function that filters and returns instructor names based on the chosen course for paper setters.
* course\_assessment\_credit: A whitelisted function that retrieves course assessment credits for the specified student group.
* get\_courses: A whitelisted function that returns a list of courses based on the chosen program, exam declaration, and assessment criteria.
* get\_assessment\_criteria\_detail: A whitelisted function that retrieves assessment criteria details for a specified course and criteria.
* get\_courses\_by\_paper\_setter: A whitelisted function that returns a list of courses assigned to a specific paper setter.
* make\_exam\_paper\_setting\_by\_paper\_setting\_date: Initiates the creation of Exam Paper Setting documents based on the paper\_setting\_start\_date.
* make\_exam\_paper\_setting: Creates Exam Paper Setting documents for examiners based on the specified document's data, including academic details and examiner assignments.

These functions and methods collectively handle various aspects of the Exam Assessment Plan document, including validations, notifications, and the creation of related Exam Paper Setting documents.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Exam Declaration | Link Field | Fetched from Exam Declaration master screen | Yes |  | (R)Exam Schedule |
| 2 | Grading Scale | Link Field | Fetched from Grading Scale master screen | Yes |  |  |
| 3 | Exam Coordinator | Link Field | Fetched from Trainer Screen | Yes |  |  |
| 4 | Assessment Criteria | Link Field | Fetched from Assessment Criteria master screen | Yes |  |  |
| 5 | Programs | Link Field | Auto fetch based on “Exam Declaration” | Yes |  | (R) Course |
| 6 | Semester | Link Field | Filtered will be applied based on selected “Programs” and existing “Exam Declaration” | Yes |  |  |
| 7 | Academic Year | Link Field | Auto fetch based on “Exam Declaration” | Yes |  |  |
| 8 | Academic Term | Link Field | Auto fetch based on “Exam Declaration” | Yes |  |  |
| 9 | Exam Coordinator Name | Link Field | Fetched from Trainer Screen | Yes |  |  |
| 10 | **Exam Assessment Plan Item** | Table | Child Table: Course Assessment Plan Item | Yes | This table is described below |  |
| 11 | Paper Setting Start Date | Date Picker | User Input | Yes |  |  |
| 12 | Paper Setting End Date | Date Picker | User Input | Yes | On the basis of the end date, the paper setter needs to submit the paper sets, that he/she prepared |  |
| 13 | **Paper Setter List** | Table | Child Table: Moderator List |  | This tables is described below |  |
| 14 | **Moderator List** | Table | Child Table: Exam Assessment Plan |  | This table is described below |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Exam Assessment Plan Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link Field | Auto fetch based on selected “Programs” & “Semester” & “Assessment Criteria” | Yes |  | (R) Module |
| 2 | Course Name | Text Field | Auto fetch based on Course |  |  | (R) Module Name |
| 3 | Course Code | Text Field | Auto fetch based on Course |  |  | (R) Module Code |
| 4 | Passing Marks | Number | Auto fetch based on Course |  |  |  |
| 5 | Total Marks | Number | Auto fetch based on Course | Yes |  |  |
| 6 | Total Credit | Number | Auto fetch based on Course | Yes |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Paper Setter List** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link Field | Only fetch those courses which belongs to the selected semester | Yes |  | (R) Module |
| 2 | Course Code | Text Field | Auto fetch based on Course |  |  | (R) Module Code |
| 3 | Course Name | Text Field | Auto fetch based on Course |  |  | (R) Module Name |
| 4 | Paper Setter | Link Field | Fetched from Instructor master screen | Yes |  |  |
| 5 | Full Name | Text Field | Auto fetch based on Paper Setter |  |  |  |
| 6 | No of Sets | Number |  | Yes |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Moderator List** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link Field | Fetched those course which are present in Paper setter list | Yes |  | (R) Module |
| 2 | Course Code | Text Field | Auto fetch based on Course |  |  | (R) Module Code |
| 3 | Course Name | Text Field | Auto fetch based on Course |  |  | (R) Module Name |
| 4 | Moderator | Link Field | Fetched from Instructor master screen | Yes |  |  |
| 5 | Moderator Name | Text Field | Auto fetch based on Moderator |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

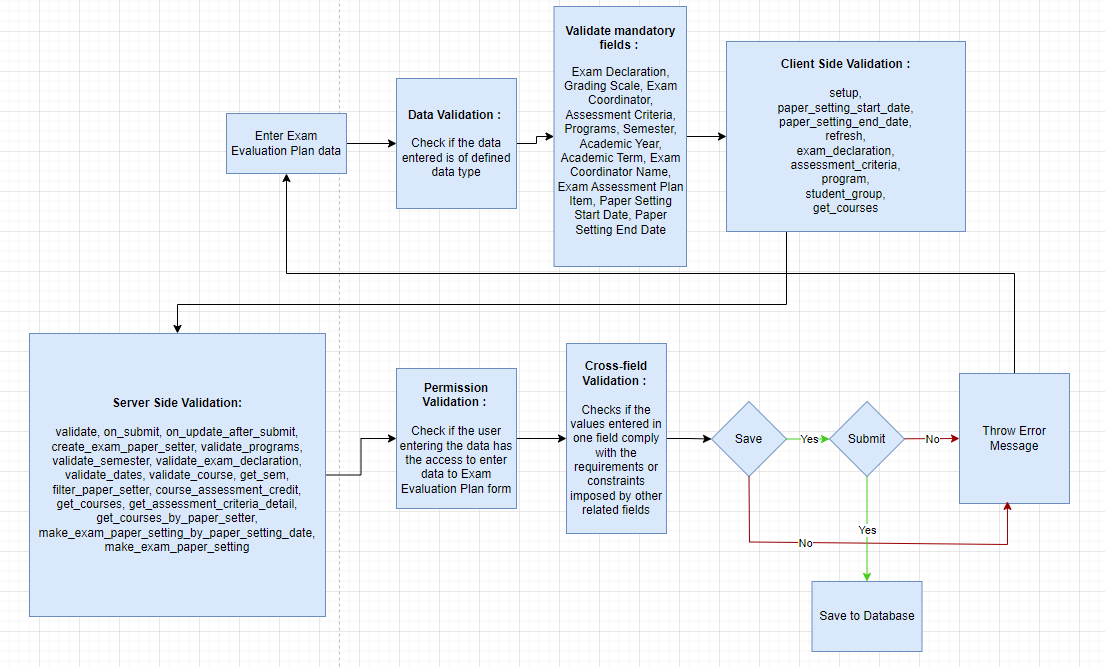


Figure : Exam Evaluation Plan Process Flow

### Pre-requisites and Dependency

* Exam Declaration
* Instructor & Instructor Log
* Program
* Semester
* Course
* Student Group

## Exam Paper Setting

The Exam Paper Setting is used to prepare paper setting schedules and plan for paper setters and moderators for a particular exam declaration.

### Use Case Diagram

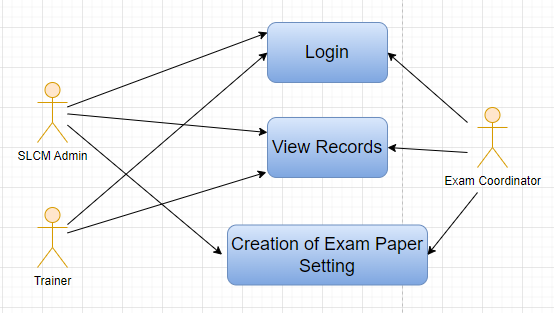


Figure : Exam Paper Setting Use Case Diagram

### Design of Workflow

* Not Applicable (Not yet finalized)

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Assignment Upload screen.

The following validation is done on client side:

* Mandatory checks for fields : Exam Declaration, Assessment Criteria, Grading Scale, Exam Coordinator, Programs, Semester, Academic Year, Academic Term, Paper Setting Start Date, Paper Setting End Date
* Linked Fields : The Exam Declaration field is linked with the Exam Declaration screen.The Assessment Criteria field is linked with the Assessment Criteria screen.The Grading Scale field is linked with the Grading Scale screen.The Exam Coordinator field is linked with the Trainer screen.The Programs field is linked with the Programs screen.The Semester field is linked with the Semester screen.The Academic Year field is linked with the Academic Year screen.The Academic Term field is linked with the Academic Term screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* date\_validation(doc): This function performs date validation for an Exam Paper Setting document. It checks if the schedule\_date of the given document falls within the range of paper\_setting\_start\_date and paper\_setting\_end\_date specified in the related Exam Assessment Plan documents. If the validation fails, a "Exam Assessment Plan paper setting dates Not Exists" error is thrown.
* ExamPaperSetting(Document): This class represents the document type "Exam Paper Setting". It defines several methods that control the behavior of instances of this document type. The validate method calls the date\_validation function for validating the schedule date. The on\_change method throws an error if the paper copy is not attached when the workflow state is set to "Approved." The after\_insert method sets user permissions after the document is inserted. The set\_user\_permission method sets user permissions for exam coordinators, examiners, and moderators. The on\_trash method is triggered when the document is deleted and calls the delete\_permission method to remove associated user permissions.
* set\_user\_permission(self): This method within the ExamPaperSetting class sets user permissions for instructors (exam coordinators, examiners, and moderators) based on their names. It calls the set\_instructor\_permission method for each instructor name to grant them access to the current document.
* delete\_permission(self): This method within the ExamPaperSetting class deletes user permissions associated with the current document. It retrieves all user permissions with a reference to the current document and deletes them.
* set\_instructor\_permission(self, instructor): This method within the ExamPaperSetting class grants user permissions to an instructor. It queries the database to find the instructor's associated employee record and extracts the user\_id. If the user\_id is available, the add\_user\_permission function is called to add a user permission for the instructor on the current document.
* get\_userid(examiner): This function queries the database to retrieve the user\_id associated with a given examiner's name from the Employee and Instructor tables. It returns the user\_id if found.
* is\_verified\_user(docname): This function checks whether the current user is a verified user for the given Exam Paper Setting document. It retrieves information about the document, including the exam coordinator, examiner, and moderator names. It then checks the roles of the current user and compares them to determine if they have the necessary permissions to access the document in its various workflow states.
* filter\_examiner(doctype, txt, searchfield, start, page\_len, filters): This function is a whitelisted function used for filtering examiners based on a search term (txt). It queries the database for "Paper Setter Item" records related to a specific assessment plan and course. It returns a list of examiners and their full names that match the search term.
* filter\_moderator(doctype, txt, searchfield, start, page\_len, filters): Similar to the previous function, this one filters moderators based on a search term. It queries the database for "Moderator List" records related to a specific assessment plan and course, returning a list of moderators and their names that match the search term.
* filter\_course(doctype, txt, searchfield, start, page\_len, filters): This function filters courses based on a search term. It queries the database for "Course Assessment Plan Item" records related to a specific assessment plan, returning a list of course names, course names, and course codes that match the search term.
* get\_assessment\_plan\_details(assessment\_plan): This function retrieves details of an assessment plan based on its name. It queries the database for "Exam Assessment Plan" records and returns information about the programs, program, academic year, and academic term associated with the assessment plan.
* get\_examiner\_moderator(assessment\_plan, course): This function retrieves the examiner and moderator names for a given assessment plan and course. It queries the database for "Paper Setter Item" and "Moderator List" records related to the specified assessment plan and course, returning the examiner and moderator names.

These functions and methods are designed to facilitate the management of Exam Paper Setting documents and related permissions.

**Notification**

* After the paper setter moderator list is published, the paper setter will take three weeks to set the paper, and the moderator will take three days to check the paper sets
* Once the paper setter created the paper sets,
* Then reminder sent to the director and Examination Head for review through mail
* Paper sets needs to be approved either by director or Examination Head
* Mail will be triggered to the paper setter and director when approved by Examination Head.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Exam Declaration | Link | Links to Exam Declaration Screen | Yes |  |  |
| 2 | Exam Type | Link | Links to Exam Type Screen |  |  |  |
| 3 | Assessment Criteria | Link | Links to Assessment Criteria | Yes |  |  |
| 4 | Grading Scale | Link | Links to Grading Scale screen | Yes |  |  |
| 5 | Exam Coordinator | Link | Links to Trainer Screen | Yes |  |  |
| 7 | Department | Link | Links to Department Screen |  |  |  |
| 8 | Programs | Link | Links to Programs Screen | Yes |  |  |
| 9 | Semester | Link | Links to Semester Screen | Yes |  |  |
| 10 | Academic Year | Link | Links to Academic Year Screen | Yes |  |  |
| 11 | Academic Term | Link | Links to Academic Term Screen | Yes |  |  |
| 12 | Class Room | Link |  |  |  |  |
| 13 | Exam Coordinator Name | Data |  | Yes |  |  |
| 15 | Exam Assessment Plan Item | Table |  | Yes |  |  |
| 17 | Paper Setting Start Date | Date |  | Yes |  |  |
| 18 | Paper Setting End Date | Date |  | Yes |  |  |
| 19 | Paper Setter List | Table |  |  |  |  |
| 20 | Moderator List | Table |  |  |  |  |

### Processes After Form Submission

* The workflow will get triggered on submission

### Process Flow:

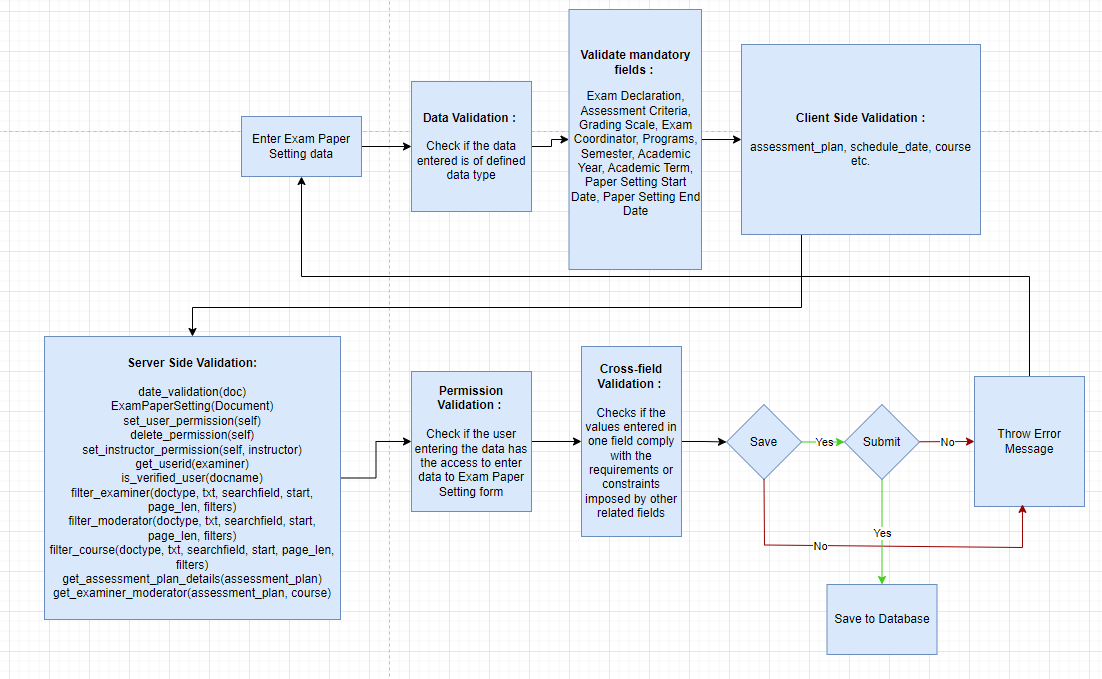


Figure : Exam Paper Setting Process Flow

### Pre-requisites and Dependency

* Exam Evaluation Plan

## Component Wise Evaluation

Component Wise Evaluation allows us to create an evaluation for a particular assessment criterion in a course for a student enrolled in that course.

### Use Case Diagram

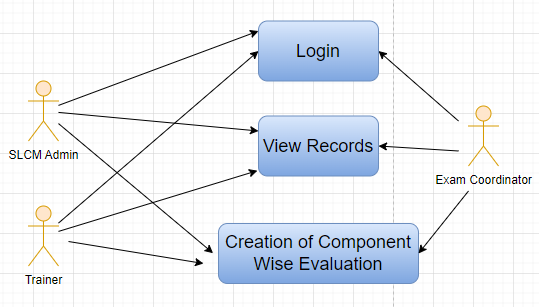


Figure : Component Wise Evaluation Use Case Diagram

### Design of Workflow

* None

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Component Wise Evaluation screen.

The following validation is done on client side:

* Mandatory checks for fields : Student, Academic Year, Academic Term, Course, Assessment Criteria, Exam Declaration, Attendance Status
* Linked Fields : The Student field is linked with the Student screen.The Academic Year field is linked with the Academic Year Screen.The Academic Term field is linked with the Academic Term Screen.The Course field is linked with the Course Screen.The Assessment Criteria field is linked with the Assessment Criteria Screen.The Exam Declaration field is linked with the Exam Declaration Screen.The Department field is linked with the Department Screen.The Course Type field is linked with the Program Grades Screen.The Programs field is linked with the Programs Screen.The Semester field is linked with the Semester Screen.The Module Wise Exam Group field is linked with the Module Wise Exam Group Screen.The Exam Assessment Plan field is linked with the Exam Evaluation Plan Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the definitions in the server side:

* CourseAssessment class (Document): This class represents a document type named "CourseAssessment". It contains several methods for validation and data retrieval related to course assessments. The methods include validate, which performs various validation checks; get\_module\_wise\_exam\_group, which retrieves an exam group based on exam declaration and course; on\_update\_after\_submit, which updates the qualifying status after the document is submitted; exam\_type, which retrieves the exam category based on an exam declaration; qualifying\_status, which determines whether the assessment passes or fails based on earned and passing marks; validate\_marks, which checks if earned marks exceed total marks; and validate\_attendance, which ensures consistency between attendance status and earned marks.
* get\_courses function: This function is used to retrieve a list of courses for a given exam declaration. It queries the database to fetch course information from the "Exam Courses" table based on the provided exam declaration.
* get\_assessment\_criteria function: This function retrieves assessment criteria associated with a given exam declaration. It queries the "Exam Declaration" table to fetch assessment criteria based on the provided exam declaration.
* get\_details function: This function retrieves program enrollment details for a student and a specific course. It uses the "Course Enrollment" and "Program Enrollment" tables to gather information such as programs, academic years, and terms.
* get\_exam\_declaration function: This function retrieves exam declaration names based on filters like student, program, academic year, and academic term. It first obtains a list of module-wise exam groups associated with the student and then maps these groups to exam declarations.
* get\_assessment\_criteria\_detail function: This function retrieves detailed information about assessment criteria for a specific course and criteria. It queries the "Credit distribution List" table to fetch credits, total marks, and passing marks associated with the given course and criteria.
* get\_exam\_assessment\_plan function: This function retrieves exam assessment plans based on various filters like program, academic year, and exam declaration. It queries the "Exam Assessment Plan" and "Course Assessment Plan Item" tables to retrieve assessment plans associated with the provided filters.

**Notification**

None

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link | Links to Student Screen | 1 |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll No | Data |  |  |  |  |
| 4 | Registration Number | Data |  |  |  |  |
| 5 | Academic Year | Link | Links to Academic Year Screen | 1 |  |  |
| 6 | Academic Term | Link | Links to Academic Term Screen | 1 |  |  |
| 7 | Course | Link | Links to Course Screen | 1 |  |  |
| 8 | Module Name | Data |  |  |  |  |
| 9 | Assessment Criteria | Link | Links to Assessment Criteria | 1 |  |  |
| 10 | Exam Declaration | Link | Links to Exam Declaration Screen | 1 |  |  |
| 11 | Department | Link | Links to Department Screen |  |  |  |
| 12 | Course Type | Link | Links to Program Grades Screen |  |  |  |
| 13 | Programs | Link | Links to Programs Screen |  |  |  |
| 14 | Semester | Link | Links to Semester Screen |  |  |  |
| 15 | Module Wise Exam Group | Link | Links to Module Wise Exam Group Screen |  |  |  |
| 16 | Exam Category | Data |  |  |  |  |
| 17 | Exam Assessment Plan | Link | Links to Exam Evaluation Plan Screen |  |  |  |
| 18 | Qualifying Status | Data |  |  |  |  |
| 19 | Passing Marks | Data |  |  |  |  |
| 20 | Attendence Status | Select |  | 1 |  |  |
| 21 | Earned Marks | Float |  |  |  |  |
| 22 | Total Marks | Float |  |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:



Figure : Component Wise Evaluation Process Flow

### Pre-requisites and Dependency

* Exam Declaration
* Course
* Assessment Criteria
* Grading Scale

## Component Wise Evaluation Tool

Component Wise Evaluation Tool helps in entering marks earned for the students in a semester for a particular course and particular assessment criteria.

### Use Case Diagram

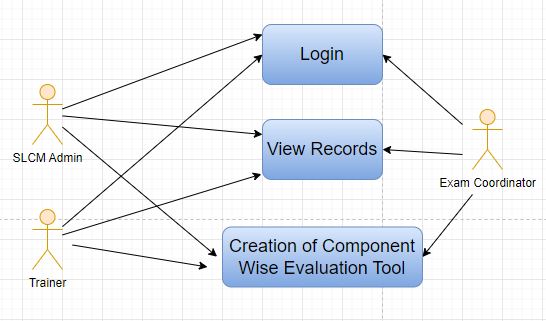


Figure : Component Wise Evaluation Tool Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Component Wise Evaluation Tool screen.

The following validation is done on client side:

* Mandatory checks for fields : Academic Year, Academic Term, Course, Assessment Criteria
* Linked Fields : The Academic Year field is linked with the Academic Year master screen.The Academic Term field is linked with the Academic Term master screen, and a filter is applied based on the Academic Year.The Exam Declaration field is linked with the Exam Declaration master screen, and a filter is applied based on the Semester and Academic Term.The Exam Assessment Plan field is linked with the Exam Assessment Plan master screen.The Program Grade field is linked with the Program Grades master screen.The Programs field is linked with the Programs master screen, and a filter is applied based on the Program Grade.The Semester field is linked with the Program master screen, and a filter is applied based on the Programs.The Course field is linked with the Course master screen, and a filter is applied based on the Semester.The Assessment Criteria field is linked with the Assessment Criteria master screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The server side validations for Component Wise Evaluation Tool are:

* get\_enroll\_students(course, criteria, exam\_declaration): This function retrieves a list of enrolled students for a specific course and criteria based on an exam declaration. It fetches student details and assigns them unique IDs while also including credit and total marks information. It returns the student list along with an empty course\_assessment dictionary.
* get\_course(program): Retrieves a list of courses associated with a particular program from the Program Course doctype.
* enroll\_student(source\_name): Maps a student applicant to a student record and returns a new program enrollment document linked to that student.
* check\_attendance\_records\_exist(course\_schedule=None, student\_group=None, date=None): Checks if attendance records exist for a given course schedule, student group, and date. Returns the list of attendance records if found.
* mark\_attendance(students\_present, students\_absent, course\_schedule=None, student\_group=None, date=None): Creates attendance records for students marked as present and absent in a given course schedule, student group, and date.
* get\_student\_guardians(student): Retrieves a list of guardians associated with a specific student.
* get\_student\_group\_students(student\_group, include\_inactive=0): Returns a list of students within a student group, optionally including inactive students.
* get\_fee\_structure(program, academic\_term=None): Retrieves the fee structure associated with a program and an optional academic term.
* get\_fee\_components(fee\_structure): Retrieves fee components within a specific fee structure.
* get\_fee\_schedule(program, student\_category=None): Fetches fee schedules for a program and an optional student category.
* collect\_fees(fees, amt): Records the collection of fees, updates paid and outstanding amounts, and returns the new paid amount.
* get\_course\_schedule\_events(start, end, filters=None): Fetches events for rendering in a course schedule calendar view between specified start and end dates, filtered by optional parameters.
* get\_assessment\_criteria(course): Retrieves assessment criteria and their weightage for a given course.
* get\_assessment\_details(course\_assessment\_plan): Fetches assessment details including theory, practical scores, credits, and maximum scores for a given assessment plan.
* get\_result(student, course\_assessment\_plan): Retrieves the submitted result of a student for a specific assessment plan.
* get\_grade(grading\_scale, percentage): Calculates and returns the grade based on a grading scale and percentage.
* mark\_assessment\_result(course\_assessment\_plan, scores): Records assessment results for students based on the given assessment plan and scores.
* submit\_assessment\_results(course\_assessment\_plan, student\_group): Submits assessment results for all students within a specific student group for a given assessment plan.
* update\_email\_group(doctype, name): Updates an email group with student or guardian email addresses.
* get\_current\_enrollment(student, academic\_year=None): Retrieves the current program enrollment details for a student within a specified academic year.
* get\_total\_marks(course, criteria): Retrieves total marks and credits for a specific course and criteria.
* make\_course\_assessment(course\_assessment): Creates course assessment records for students based on the provided data.
* get\_courses(doctype, txt, searchfield, start, page\_len, filters): Retrieves a list of courses based on search criteria and semester.
* get\_assessment\_criteria\_list(doctype, txt, searchfield, start, page\_len, filters): Retrieves a list of assessment criteria based on the selected course.
* get\_semester\_and\_exam\_assessment\_plan(declaration\_id=None): Retrieves the semester and exam assessment plan based on an exam declaration.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Academic Year | Link Field | Fetched from Academic Year master screen | Yes |  |  |
| 2 | Academic Term | Link Field | Fetched from Academic Term master screen and filter applied on the basis of Academic Year | Yes |  |  |
| 3 | Exam Declaration | Link Field | Fetched from Exam Declaration master screen and filtered applied on the basis of Semester and Academic Term |  |  |  |
| 4 | Exam Assessment Plan | Link Field | Fetched from Exam Assessment Plan master screen |  |  |  |
| 5 | Total Students | Number |  |  |  |  |
| 6 | Program Grade | Link Field | Fetched from Program Grades master screen |  |  | (R)  Course Grade |
| 7 | Programs | Link Field | Fetched from Programs master screen, filtered applied on the basis of Program Grade |  |  | (R)  Courses |
| 8 | Semester | Link Field | Fetched from Program master screen and filtered applied on the basis of Programs |  |  |  |
| 9 | Course | Link Field | Fetched from Course master screen and filtered applied on the basis of Semester | Yes |  | (R)  Module |
| 10 | Course Code | Text Field | Auto fetch based on Course |  |  | (R)  Module Code |
| 11 | Course Name | Text Field | Auto fetch based on Course |  |  | (R)  Module Name |
| 12 | Assessment Criteria | Link Field | Fetched from Assessment Criteria master screen | Yes |  | (R)Assessment Component |
| 13 | **Student Details Data** | Table | HTML |  | This table is described below |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student Details Data** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | HTML Field | Auto fetched when all the above relevant inputs given |  |  | (R) Student Id |
| 2 | Student Name | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 3 | Roll No | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 4 | Earned Marks | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 5 | Total Marks | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

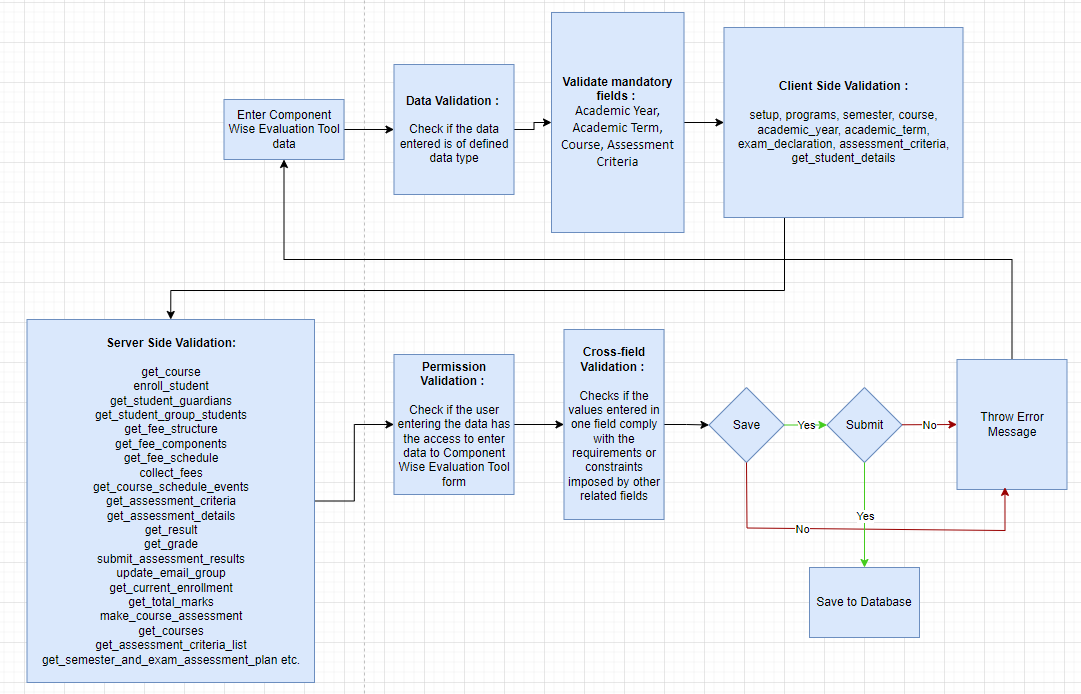


Figure : Component Wise Evaluation Tool Process Flow

### Pre-requisites and Dependency

* Exam Declaration
* Course
* Assessment Criteria
* Grading Scale

## Component Wise Reevaluation

Component Wise Reevaluation allows us to create a reevaluation for a particular assessment criterion in a course for a student enrolled in that course.

### Use Case Diagram

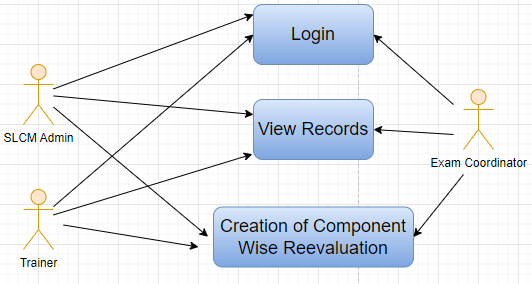


Figure : Component Wise Reevaluation Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Component Wise Reevaluation screen.

The following validation is done on client side:

* Mandatory checks for fields : Student, Academic Year, Academic Term, Attendance Status, Course, Assessment Criteria, Final Marking Item
* Linked Fields : The Student field is linked to the Student Screen.The Academic Year field is linked with the Academic Year Screen.The Academic Term field is linked with the Academic Term Screen.The Department field is linked to the Department Screen.The Course Type field is linked to the Program Grade Screen.The Programs field is linked to the Programs Screen.The Semester field is linked to the Semester Screen.The Course field is linked to the Course Screen.The Assessment Criteria field is linked to the Assessment Criteria.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Component Wise Reevaluation:

* AssessmentCreditsAllocation(Document):

This class represents a Document type called "Assessment Credits Allocation". It contains validation logic and methods for various calculations related to allocating credits based on assessments. The validate method triggers a series of validations and calculations when the document is being saved. Methods like qualifying\_status\_child, final\_earned\_marks\_calculation\_child, passing\_marks\_calculation are used to calculate and update qualifying status, final earned marks, and passing marks based on the given assessment criteria. The class also has methods to validate duplicate records, student eligibility, and assessment criteria selection.

* qualifying\_status\_child(self):

This method iterates through the "final\_credit\_item" child table of the Assessment Credits Allocation document. It compares the final earned marks of each credit item with its associated passing marks. If the final earned marks are less than or equal to the passing marks, the "qualifying\_status" of that credit item is set to "Fail"; otherwise, it's set to "Pass".

* final\_earned\_marks\_calculation\_child(self):

This method calculates the final earned marks for each credit item in the "final\_credit\_item" child table. It adds the earned marks and grace marks of the credit item to calculate the final earned marks. Additionally, it retrieves the "exam\_declaration" field from the "Course Assessment" document related to the credit item.

* passing\_marks\_calculation(self):

This method calculates and sets the passing marks for each credit item based on the associated credit distribution list (cdl\_list). If passing marks are not explicitly defined for a credit item, it fetches the passing marks from the cdl\_list.

* validate\_duplicate\_record(self):

This method checks for duplicate Assessment Credits Allocation records with the same student, course, assessment criteria, academic year, and academic term. If a duplicate record is found, it raises an error.

* validate\_student(self):

This method validates whether the selected student is enabled. If the student is not enabled, it raises an error indicating that the student must be enabled.

* validate\_assessment\_criteria(self):

This method validates the selected assessment criteria for the course against the assessment criteria associated with the student's ongoing course enrollments. If the selected assessment criteria is not among the valid options, it raises an error.

* validate\_marks(self):

This method validates various mark-related fields in the document, including grace marks, final marks, earned credits, total credits, etc. It ensures that marks and credits are within valid ranges and relationships, and raises errors if conditions are not met.

* get\_course\_details(self):

This method fetches details of the selected course from the "Course" document and searches for the credit distribution entry that matches the assessment criteria. It returns the relevant credit distribution entry.

* get\_courses(doctype, txt, searchfield, start, page\_len, filters):

This function retrieves a list of courses based on the provided filters (primarily the student's name) and search text. It performs a SQL query to fetch courses associated with the student from the "Program Enrollment" and "Program Enrollment Course" documents.

* get\_assessment\_criteria(doctype, txt, searchfield, start, page\_len, filters):

This function retrieves a list of assessment criteria based on the provided filters (student, course) and search text. It queries the "Course Enrollment" and "Credit distribution List" documents to find assessment criteria associated with the student and course.

* get\_course\_assessment(student,course,assessment\_criteria):

This function retrieves and calculates various details related to course assessments for a given student, course, and assessment criteria. It calculates total earned marks, total total marks, and weightage marks based on existing "Course Assessment" records.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link | Links to Student Screen | Yes |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll No | Data |  |  |  |  |
| 4 | Registration Number | Data |  |  |  |  |
| 5 | Academic Year | Link | Links to Academic Year Screen | Yes |  |  |
| 6 | Academic Term | Link | Links to Academic Term Screen | Yes |  |  |
| 7 | Attendence Status | Select |  | Yes |  |  |
| 8 | Department | Link | Links to Department Screen |  |  |  |
| 9 | Course Type | Link | Links to Program Grade Screen |  |  |  |
| 10 | Programs | Link | Links to Programs Screen |  |  |  |
| 11 | Semester | Link | Links to Semester Screen |  |  |  |
| 12 | Course | Link | Links to Course Screen | Yes |  |  |
| 13 | Module Name | Data |  |  |  |  |
| 14 | Module Code | Data |  |  |  |  |
| 15 | Assessment Criteria | Link | Links to Assessment Criteria | Yes |  |  |
| 16 | Get Assessments | Button |  |  |  |  |
| 17 | Final Marking Item | Table | Links to Final Credit Item description of which is given below | Yes |  |  |
| 19 | Weightage Marks | Float |  |  |  |  |
| 20 | Total Marks | Float |  |  |  |  |
| 21 | Total Credits | Float |  |  |  |  |
| 22 | Qualifying Status | Data |  |  |  |  |
| 23 | Grace Marks | Float |  |  |  |  |
| 24 | Final Earned Marks | Float |  |  |  |  |
| 25 | Passing Marks | Data |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Final Credit Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course Assessment | Link | Links to Course Assessment Screen |  |  |  |
| 2 | Wightage Marks | Float |  |  |  |  |
| 3 | Grace Marks | Float |  |  |  |  |
| 4 | Final Earned Marks | Data |  |  |  |  |
| 5 | Passing Marks | Float |  |  |  |  |
| 6 | Total Marks | Float |  |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

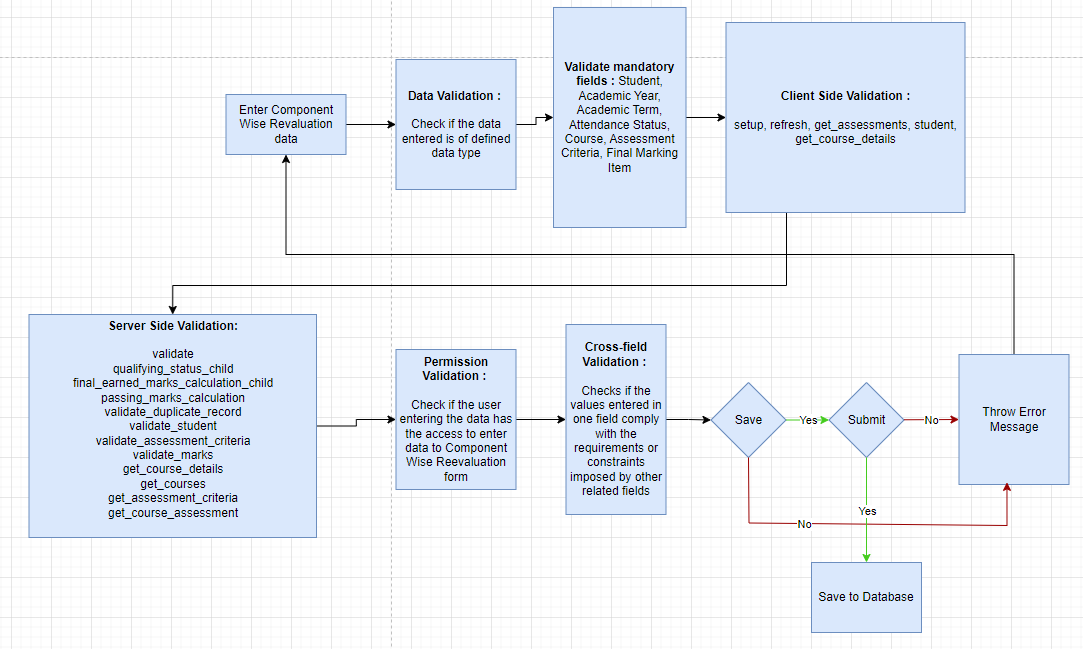


Figure : Component Wise Reevaluation Process Flow

### Pre-requisites and Dependency

* Exam Declaration
* Course
* Assessment Criteria
* Grading Scale

## Component Wise Reevaluation Tool

Component Wise Reevaluation Tool helps in entering marks earned for the students on reevaluation in a semester for a particular course and particular assessment criteria.

### Use Case Diagram

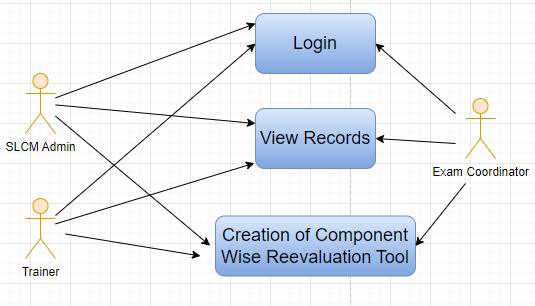


Figure : Component Wise Reevaluation Tool Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Component Wise Reevaluation Tool screen.

The following validation is done on client side:

* Mandatory checks for fields : Academic Year, Academic Term, Program Grade, Course, Assessment Criteria
* Linked Fields : The Academic Year field is linked with the Academic Year screen.The Academic Term field is linked with the Academic Term screen, and filtering is applied based on the selected Academic Year.The Program Grade field is linked with the Program Grades master screen.The Programs field is linked with the Programs master screen, and filtering is applied based on the selected Program Grade.The Semester field is linked with the Program master screen, and filtering is applied based on the selected Programs.The Course field is linked with the Course master screen, and filtering is applied based on the selected Semester.The Assessment Criteria field is linked with the Assessment Criteria master screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

There following are the server side validations for Component Wise Reevaluation Tool:

* get\_course\_details(self): This method is used to retrieve details of a course's credit distribution based on the assessment criteria specified.
* get\_student\_allocations(self): This method retrieves student allocation data for continuous evaluation. It gathers data about students, their earned marks, attendance status, and other related information based on different exam categories like "Regular" and "Re-Exam".
* make\_continuous\_evaluation(continuous\_evaluation): This function processes and creates continuous evaluation records based on the provided data. It handles creating records for both regular and re-exam categories, ensuring that relevant data is saved according to the assessment criteria.
* get\_student\_allocations\_dict(doc): This function fetches student allocation data and returns it in a dictionary format, with student IDs as keys and their related assessment data as values. It is used for the "Regular" exam category.
* get\_student\_allocations\_dict\_fail(doc, exam\_declaration): Similar to the above function, this one fetches student allocation data for the "Re-Exam" category. It takes an additional parameter, exam\_declaration, to distinguish the specific re-exam being referred to.
* validate\_duplicate\_record(self): This function checks for duplicate records of assessment credits allocation. It ensures that the same student, course, assessment criteria, academic year, and term don't have duplicate records in the system.
* get\_course\_details(doc): A helper function to retrieve specific credit distribution details for a course and assessment criteria combination.
* get\_courses(doctype, txt, searchfield, start, page\_len, filters): This function is used for autocompletion when searching for courses. It fetches a list of courses based on the semester filter and the entered search text, returning a list of course names, course names, and course codes that match the search.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Academic Year | Link Field | Fetched from Academic Year master screen | Yes |  |  |
| 2 | Academic Term | Link Field | Fetched from Academic Term master screen and filtered applied on the basis of Academic Year | Yes |  |  |
| 3 | Program Grade | Link Field | Fetched from Program Grades master screen | Yes |  | (R) Course Grade |
| 4 | Programs | Link Field | Fetched from Programs master screen and filtered applied on the basis of Program Grades |  |  | (R) Courses |
| 5 | Semester | Link Field | Fetched from Program master screen and filtered applied on the basis of Programs |  |  |  |
| 6 | Course | Link Field | Fetched from Course master screen and filtered applied on the basis of Semester | Yes |  | (R) Module |
| 7 | Course Code | Text Field | Auto fetch based on Course |  |  | (R) Module Code |
| 8 | Course Name | Text Field | Auto fetch based on Course |  |  | (R) Module Name |
| 9 | Assessment Criteria | Link Field | Fetched from Assessment Criteria master screen | Yes |  | (R)Assessment Component |
| 10 | Exam Category | Select | Options: Regular, Re-Exam |  |  |  |
| 11 | Student Inputs | HTML |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student Details Data** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | HTML Field | Auto fetched when all the above relevant inputs given |  |  | (R)Student Id |
| 2 | Student Name | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 3 | Roll No | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 4 | Earned Marks | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |
| 5 | Total Marks | HTML Field | Auto fetched when all the above relevant inputs given |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

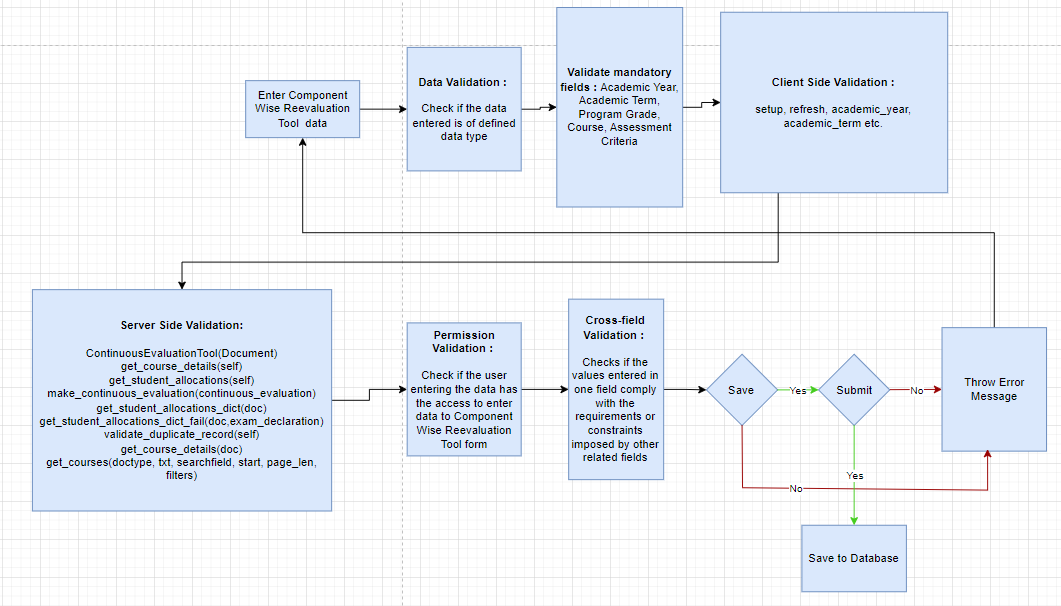


Figure : Component Wise Reevaluation Tool Process Flow

### Pre-requisites and Dependency

* Exam Declaration
* Course
* Assessment Criteria
* Grading Scale

## Final Semester Result

Final Semester Result is used to evaluate the final assessment of a student for a particular semester.

### Use Case Diagram

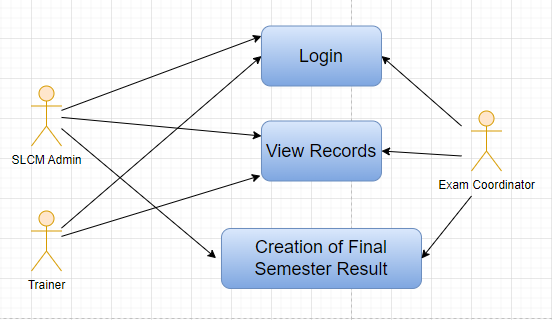


Figure : Final Semester Result Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Final Semester Result screen.

The following validation is done on client side:

* Mandatory checks for fields : Student, Grading Scale, Course
* Linked Fields : The Student field is linked with the Student Screen.The Department field is linked with the Department Screen.The Programs field is linked with the Programs Screen.The Semester field is linked with the Semester Screen.The Course field is linked with the Course Screen.The Academic Year field is linked with the Academic Year Screen.The Academic Term field is linked with the Academic Term Screen.The Grading Scale field is linked with the Grading Scale screen.The Student Group field is linked with the Student Group Screen.The Assessment Group field is linked with the Assessment Group.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Final Semester Result:

* Class ExamAssessmentResult(Document): This class represents a custom document type named "Exam Assessment Result" that inherits from the base class "Document". It defines various methods that are triggered during different life cycle events of the document, such as on submission, on cancellation, and on validation. These methods perform actions like setting assessment result items, calculating grades and percentages, validating duplicate entries, and completing/canceling course enrollments based on assessment results.
* def on\_submit(self): This method is triggered when the document is submitted. It performs a series of actions, including validating for duplicate rows in the assessment result, setting evaluation result items, calculating grades, percentages, and mapping fields, and ensuring there are no duplicate records before saving.
* def on\_cancel(self): This method is triggered when the document is canceled. It reverses the action of completing course enrollments that were previously completed when the document was submitted.
* def validate(self): This method performs validation checks on the document, such as checking for provisional admission status, validating duplicates, setting evaluation result items, and calculating percentages and grades.
* def on\_change(self): This method is triggered when there is a change in the assessment result items. If there are assessment result items, it recalculates the percentage and grade results.
* def validate\_provisional\_admission(self): This method checks if a student with provisional admission status is attempting to submit an assessment result, and if so, it raises an exception suggesting changing the status to admitted.
* def set\_assessment\_result\_items(self): This method sets assessment result items by querying the "Assessment Credits Allocation" document and creating rows in the "assessment\_result\_item" field of the document.
* def set\_grade(self): This method calculates grades based on earned marks, calculates total scores, and determines the result (pass/fail) for each assessment result item.
* def validate\_duplicate\_for\_save(self): This method validates whether there are duplicate assessment result records for saving by checking the student, document status, program, and academic term.
* def validate\_duplicate\_for\_submit(self): Similar to the above method, this method validates whether there are duplicate assessment result records for submission.
* def calculate\_sgpa(self): This method calculates the Semester Grade Point Average (SGPA) for a student based on earned credits and grades.
* def calculate\_percentage\_grade\_result(self): This method calculates the percentage, grade, and result (pass/fail) for each assessment result item based on earned and total marks.
* def map\_fields(self): This method maps fields from other documents and programs to the current document, specifically related to the order of semesters.
* def calculate\_sgpa\_cgpa(self): This method calculates both Semester Grade Point Average (SGPA) and Cumulative Grade Point Average (CGPA) for a student based on earned credits and grades.
* def complete\_course\_enrollment(self): This method marks course enrollments as completed based on assessment results.
* def cancel\_complete\_course\_enrollment(self): This method reverses the action of marking course enrollments as completed.
* def set\_evaluation\_result\_item(self): This method sets evaluation result items based on assessment result items, calculating earned credits, and earned marks.
* def get\_grade\_result(...): This method calculates the grade and result (pass/fail) based on a grading scale, earned marks, and total marks.
* def get\_assessment\_status(...): This method checks whether a student has completed assessments for a given semester, academic year, and term.
* def get\_student\_details(...): This method retrieves details of a student enrolled in a program.
* def filter\_courses(...): This method filters courses based on a search text and student context, returning a list of matching courses for a student.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link | Links to Student Screen | Yes |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll No | Data |  |  |  |  |
| 4 | Registration Number | Data |  |  |  |  |
| 5 | Department | Link | Links to Department Screen |  |  |  |
| 6 | Programs | Link | Links to Programs Screen |  |  |  |
| 7 | Semester | Link | Links to Semester Screen |  |  |  |
| 8 | Course | Link | Links to Course Screen |  |  |  |
| 9 | Academic Year | Link | Links to Academic Year Screen |  |  |  |
| 10 | Academic Term | Link | Links to Academic Term Screen |  |  |  |
| 11 | Grading Scale | Link | Links to Grading Scale screen | Yes |  |  |
| 12 | Status | Select |  |  |  |  |
| 13 | Student Group | Link | Links to Student Group Screen |  |  |  |
| 14 | Assessment Group | Link | Links to Assessment Group |  |  |  |
| 15 | Assessment Status | Data |  |  |  |  |
| 16 | Get Result | Button |  |  |  |  |
| 17 | Assessment Result Item | Table | Links to Assessment Result Item table |  |  |  |
| 18 | Evaluation Result Item | Table | Links to Evaluation Result Item table |  |  |  |
| 19 | Semesters SGPA | Table | Links to Semesters SGPA table |  | Hidden |  |
| 20 | Secured Marks | Float |  |  |  |  |
| 21 | Total Marks | Float |  |  |  |  |
| 22 | SGPA | Float |  |  |  |  |
| 23 | Percentage | Percent |  |  |  |  |
| 24 | Grade | Data |  |  |  |  |
| 25 | Result | Data |  |  |  |  |
| 26 | Overall CGPA | Float |  |  |  |  |
| 27 | Comment | Small Text |  |  |  |  |
| 28 | Credit Point | Float |  |  |  |  |
| 29 | SGPA in to Credit Point | Data |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Assessment Result Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link | Links to Course Screen | Yes |  |  |
| 2 | Module Code | Data |  |  |  |  |
| 3 | Module Name | Data |  |  |  |  |
| 4 | Assessment Criteria | Link |  |  |  |  |
| 5 | Earned CR | Float |  |  |  |  |
| 6 | Total CR | Float |  |  |  |  |
| 7 | Earned Marks | Float |  |  |  |  |
| 8 | Total Marks | Float |  |  |  |  |
| 9 | Grade | Data |  |  |  |  |
| 10 | Result | Select |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Evaluation Result Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link | Links to Course Screen | Yes |  |  |
| 2 | Module Code | Data |  |  |  |  |
| 3 | Module Name | Data |  |  |  |  |
| 4 | Earned CR | Float |  |  |  |  |
| 5 | Total CR | Data |  |  |  |  |
| 6 | Earned Marks | Data |  |  |  |  |
| 7 | Total Marks | Data |  |  |  |  |
| 8 | Grade | Data |  |  |  |  |
| 9 | Result | Select |  |  |  |  |
| 10 | Module Total Mark | Data |  |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

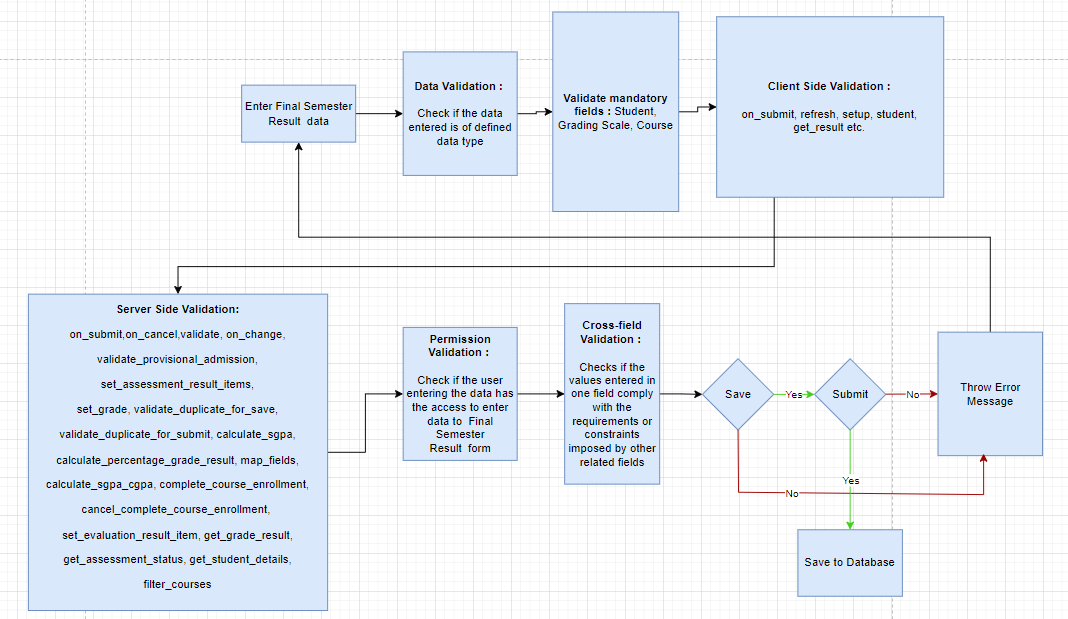


Figure : Final Semester Result Process Flow

### Pre-requisites and Dependency

* Component Wise Evaluation

## Final Semester Result Tool

Final Semester Result Tool is used to declare the final semester results for a particular group/batch of students in a semester.

### Use Case Diagram

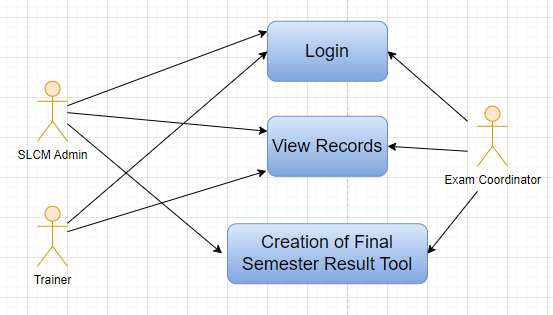


Figure : Final Semester Result Tool Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Final Semester Result Tool screen.

The following validation is done on client side:

* Mandatory checks for fields : Program Grade, Programs, Semester, Grading Scale, Academic Year, Academic Term
* Linked Fields : The Program Grade field is linked with the Program Grades master screen.The Programs field is linked with the Programs master screen, and the linkage is established based on the selected Program Grade.The Semester field is linked with the Program master screen, and the linkage is established based on the chosen Programs.The Grading scale field is linked with the Grading Scale master screen.The Academic Year field is linked with the Academic Year master screen.The Academic Term field is linked with the Academic Term master screen, and the linkage is established based on the selected Academic Year.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Final Semester Result Tool:

* Class Definition - FinalResultDeclarationTool(Document): This class represents a custom Frappe document named "FinalResultDeclarationTool". It inherits from the base class "Document". The class includes methods to process and calculate final assessment results, including grade calculations based on grading scales, attendance percentages, and other relevant data. The document's fields and methods are used to manage and store assessment results for participants.
* validate(self): This method is invoked during the validation process of the "FinalResultDeclarationTool" document. It contains the logic to calculate and store final assessment results, including grading, attendance percentages, and related information. It iterates through the list of participants, retrieves their assignment evaluations, calculates grades and percentages, and then calculates the overall result, grade, and attendance percentage. Finally, it saves these calculated values in a new "Final Assignment Result" document.
* get\_details(participant\_group\_id): This is a whitelisted function accessible via API. It retrieves details about a participant group identified by the provided participant\_group\_id. It queries and returns information about the academic year, academic term, program, course, participants' details, course name, course code, and total participant count for the specified participant group.
* get\_participants(participant\_group\_id): This is another whitelisted function accessible via API. It retrieves a list of participants and their names for a given participant group specified by participant\_group\_id.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Program Grade | Link Field | Fetched from Program Grades master screen | Yes |  |  |
| 2 | Programs | Link Field | Fetched from Programs master screen and filtered applied on the basis of Program Grade | Yes |  |  |
| 3 | Semester | Link Field | Fetched from Program master screen and filtered applied on the basis of Programs | Yes |  |  |
| 4 | Grading scale | Link Field | Fetched from Grading Scale master screen | Yes |  |  |
| 5 | Academic Year | Link Field | Fetched from Academic Year master screen | Yes |  |  |
| 6 | Academic Term | Link Field | Fetched from Academic Term master screen and filtered applied on the basis of Academic Year | Yes |  |  |
| 7 | Result Creation Status | Drop Down | In Process  Failed  Successful |  |  |  |
| 8 | Get Students | Button | User Input |  |  |  |
| 9 | Total Enrolled Student | Number | Auto created when clicked on Get Student Button |  |  |  |
| 10 | **Result Declaration Student** | Table | Child Table: Result Declaration Student |  | This table is described below |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Result Declaration Student** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field | Fetched from Student master screen |  |  |  |
| 2 | Student Name | Text Field | Auto fetch based on Student |  |  |  |
| 3 | Roll No | Text Field | Auto fetch based on Student |  |  |  |

### Processes After Form Submission

* This screen has no workflow as the form is not submittable.

### Process Flow:

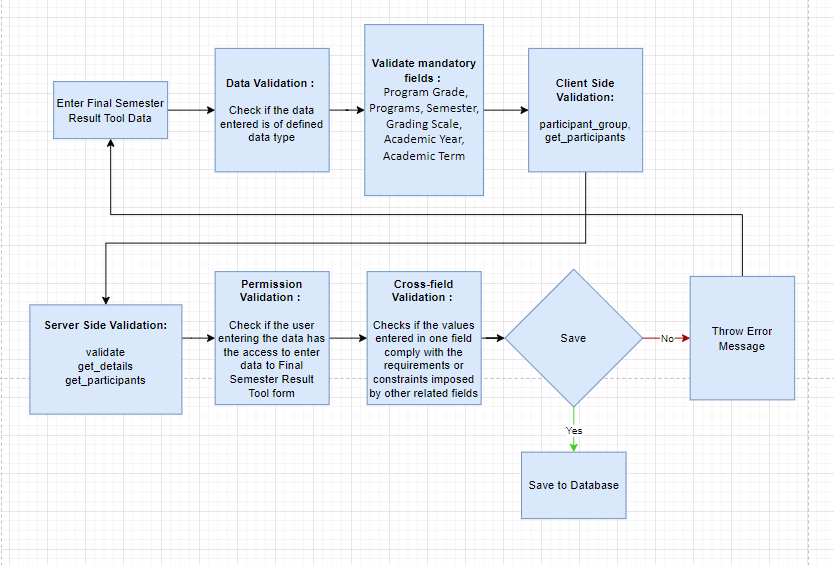


Figure : Final Semester Result Tool Process Flow

### Pre-requisites and Dependency

* Component Wise Evaluation

## Cumulative Marksheet Tool

Cumulative Marksheet Tool is a compilation of all the semesters with their grades in a single certificate. The Cumulative marksheet will have a list of subjects, grades obtained and CGPA (Cumulative Grade Point Average). And also details of students, such as roll number, name, age and date of birth, program enrollment.

### Use Case Diagram

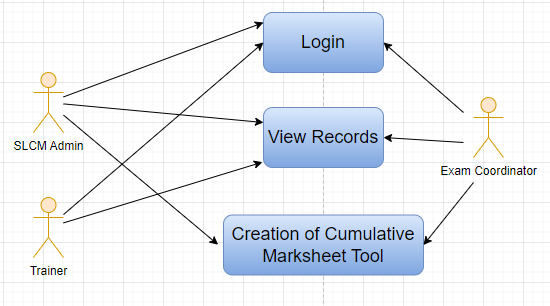


Figure : Cumulative Marksheet Tool Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Cumulative Marksheet Tool screen.

The following validation is done on client side:

* Mandatory checks for fields : Programs, Current Academic Term
* Linked Fields : The Programs field is linked with the Programs master screen.The Current Academic Term field is linked with the Academic Term master screen.The Year of Admission field is linked with the Academic Year master screen.The Year of Completion field is linked with the Academic Year master screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Cumulative Marksheet Tool:

* Class Definition - CumulativeMarksheetTool(Document):

This class represents a custom Frappe document named "CumulativeMarksheetTool". It inherits from the base class "Document". The class includes methods related to creating and managing cumulative mark sheets. It defines a validation function, and the make\_exam\_assessment\_result function, which is accessible via API. The latter function sets the "result\_creation\_status" of the document to "In Process" and either performs immediate processing for fewer records or enqueues a background job for creating cumulative mark sheets for a larger number of records.

* create\_cummulative\_marksheet(cumulative\_marksheet\_tool):

This function is responsible for actually creating cumulative mark sheets. It retrieves data from the "CumulativeMarksheetTool" document and processes each student's data. It retrieves information about program enrollment, assessment results, courses, and grades to construct the cumulative mark sheets. It calculates percentages, grades, and aggregates various data points. After constructing the mark sheet, it saves the data and updates progress.

* get\_students(academic\_term=None, programs=None):

This is a whitelisted function that returns a list of enrolled students based on the provided academic term and program (optional). It retrieves active enrolled students and returns a list containing their details.

* get\_program\_enrollment(academic\_term, programs=None):

This function retrieves program enrollment details for students based on the provided academic term and optional programs. It constructs an SQL query to fetch student details including student ID and name.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | | **Field Label** | | **Field Type** | | **Validation/Action** | | **Mandatory** | | **Remarks** | | **R/N/D** | |
| 1 | | BRANCH | | Text Field | |  | |  | |  | |  | |
| 2 | | School of | | Text Field | |  | |  | |  | |  | |
| 3 | | Specialization | | Text Field | |  | |  | |  | |  | |
| 4 | | Programs | | Link Field | | Fetched from Programs master screen | | Yes | |  | |  | |
| 5 | | Current Academic Term | | Link Field | | Fetched from Academic Term master screen | | Yes | |  | |  | |
| 6 | | Year of Admission | | Link Field | | Fetched from Academic Year master screen | |  | |  | |  | |
| 7 | | COMPLETED ON | | Text Field | |  | |  | |  | |  | |
| 8 | | Year of Completion | | Link Field | | Fetched from Academic Year master screen | |  | |  | |  | |
| 9 | | Result Creation Status | | Dropdown | | In Process/Failed/  Successful | |  | |  | |  | |
| 10 | | Get Students | | Button | | On click this button Student will fetch based on Programs & Academic Term in the Cumulative Marksheet Student | |  | |  | |  | |
| 11 | | Total Students for Marksheet | | Number | |  | |  | |  | |  | |
| 12 | | **Cumulative Marksheet Student** | | Table | | Child Table: Cumulative Marksheet Student | |  | | These table is describe below | |  | |
| 13 | | Signature of Examiner | | Attach image | | 1. On attachment, a dialogue box will be popup where the user can attach the files, take the picture with the help camera or can link to any URL.   If photos or images attached then the size will be maximum 200 kb | |  | |  | |  | |
| **Cumulative Marksheet Student** | | | | | | | | | | | | |
| **ID** | **Field Label** | | **Field Type** | | **Validation/Action** | | **Mandatory** | | **Remarks** | | **R/N/D** | |
| 1 | Student | | Link Field | | Fetched from “Get Students” Button | |  | |  | |  | |
| 2 | Student Name | | Text Field | | Auto fetch based on Student | |  | |  | |  | |
| 3 | Roll No | | Text Field | | Auto fetch based on Student | |  | |  | |  | |
| 4 | Registration Number | | Text Field | | Auto fetch based on Student | |  | |  | | (D) | |

### Processes After Form Submission

* This screen has no workflow as the form is not submittable.

### Process Flow:

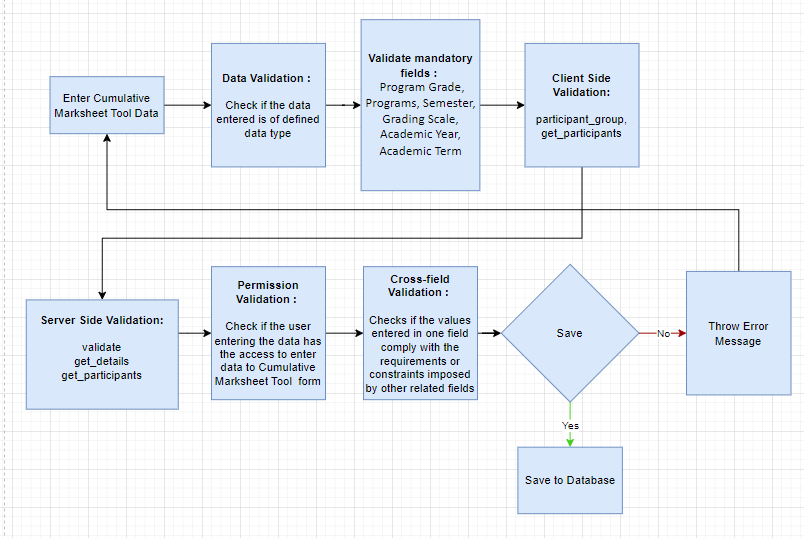


Figure : Cumulative Marksheet Tool Process Flow

### Pre-requisites and Dependency

* Final Semester Result

## Final Semester Result Report

This is the Report for Final Semester Result.

### Use Case Diagram

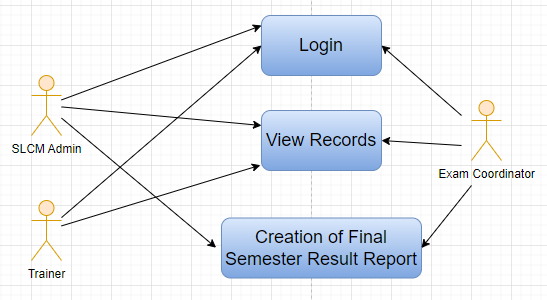


Figure : Final Semester Result Report Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Final Semester Result Report screen.

The following validation is done on client side:

* Mandatory checks for fields : Program Grades
* Linked Fields : The Program Grade field is linked with Program Grade Screen. The Program field is linked with Program Screen. The Semester field is linked with Semester Screen. The Academic Year field is linked with Academic Year Screen. The Academic Term field is linked with Academic Term Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Final Semester Result Report:

* get\_students(programs, semester, academic\_year, academic\_term):

This function retrieves a list of students based on the provided program, semester, academic year, and academic term. It queries the database for Exam Assessment Result records that match the given criteria and returns specific fields like student information, grading scale, percentage, result, etc.

* get\_course(student):

This function takes a list of students as input and retrieves the associated course details for those students. It fetches Assessment Result Item records linked to the provided list of student names and extracts information such as course code, earned marks, total marks, and assessment criteria. It then organizes and processes this data to provide a list of unique courses, assessment criteria, and their details.

* get\_assessment\_result(students):

Given a list of students, this function retrieves their assessment results. It queries the database for Assessment Result Item records linked to the provided list of student names and extracts information like course, assessment criteria, earned marks, total marks, grade, and result.

* get\_evaluation\_result(students, course):

This function retrieves evaluation results for the specified students and courses. It queries the database for Evaluation Result Item records linked to the provided student names and extracts information including course details, earned credits, earned marks, grade, and result. It also calculates the credit points (Cr\_P) based on the student's grading scale.

* total\_credit\_credit\_point(students, evaluation\_result):

This function calculates the total credit points and total credits for each student based on the provided evaluation results. It takes the student list and evaluation results as input, then iterates through the students and their associated evaluation results to calculate and assign credit points and total credits.

* FinalExamResultReport(Document):

This class represents a Frappe Document type called "FinalExamResultReport". It's a custom document type. It contains a method called "get\_student\_allocations" that coordinates the execution of the previously defined functions to gather and organize data related to student allocations, courses, assessment criteria, results, and credits. The method returns a comprehensive dictionary containing this information for further processing.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Program Grades | Link Field | Fetched from Program Grade Screen | Yes |  |  |
| 2 | Programs | Link Field | Fetched from Program Screen |  |  |  |
| 3 | Semester | Link Field | Fetched from Semester Screen |  |  |  |
| 4 | Academic Year | Link Field | Fetched from Academic Year master screen |  |  |  |
| 5 | Academic Term | Link Field | Fetched from Academic Term master screen |  |  |  |
| 6 | Report | HTML |  |  |  |  |

### Processes After Form Submission

* This screen has no workflow as the form is not submittable.

### Process Flow:

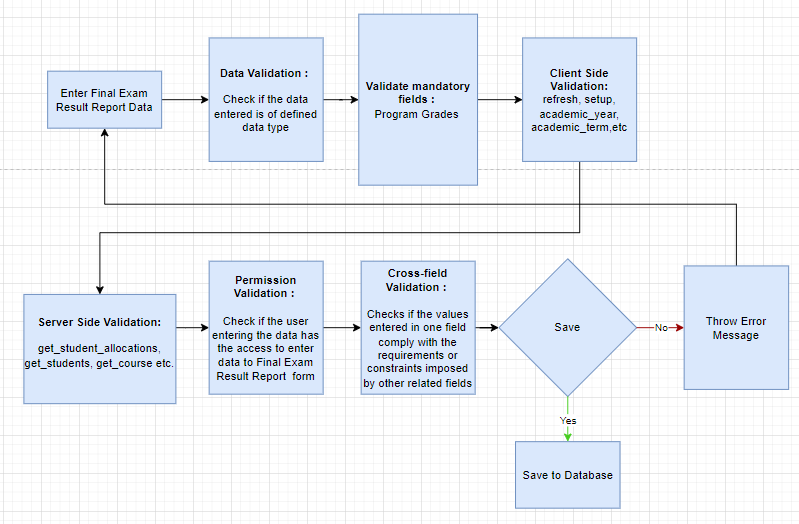


Figure : Final Semester Result Report Process Flow

### Pre-requisites and Dependency

* Final Semester Result

## Course Wise Performance

This is the Report to measure the course wise performance of students.

### Use Case Diagram

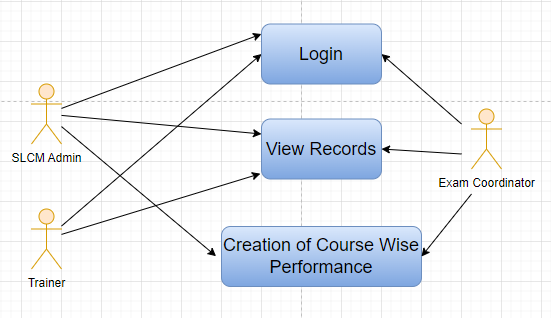


Figure : Course Wise Performance Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Course Wise Performance screen.

The following validation is done on client side:

* Mandatory checks for fields : Academic Year, Program Grades, Academic Term
* Linked Fields : The Academic Year field is linked with Academic Year Screen. The Program Grades field is linked with Program Grade Screen. The Academic Term field is linked with Academic Term Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Course Wise Performance:

* PassPercentage class (inherits from Document): This class represents a document type and is used to store and manage information related to pass percentages of various programs. It inherits from the base Document class, which provides common functionalities for managing documents.
* get\_details method within the PassPercentage class: This method is decorated with frappe.whitelist() to allow it to be accessed from the frontend. It calculates and gathers pass percentage details for different programs based on specified criteria, such as program grade and academic term. Inside a loop, it queries the database for program-related data, like enrollment, assessments, and results.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Academic Year | Link | Links to Academic Year Screen | Yes |  |  |
| 3 | Program Grades | Link | Links to Program Grade Screen | Yes |  |  |
| 5 | Academic Term | Link | Links to Academic Term Screen | Yes |  |  |
| 7 | Get Result | Button |  |  |  |  |
| 8 | Report | Table |  |  |  |  |
| 9 | Remarks | Small Text |  |  |  |  |

### Processes After Form Submission

* This screen has no workflow as the form is not submittable.

### Process Flow:

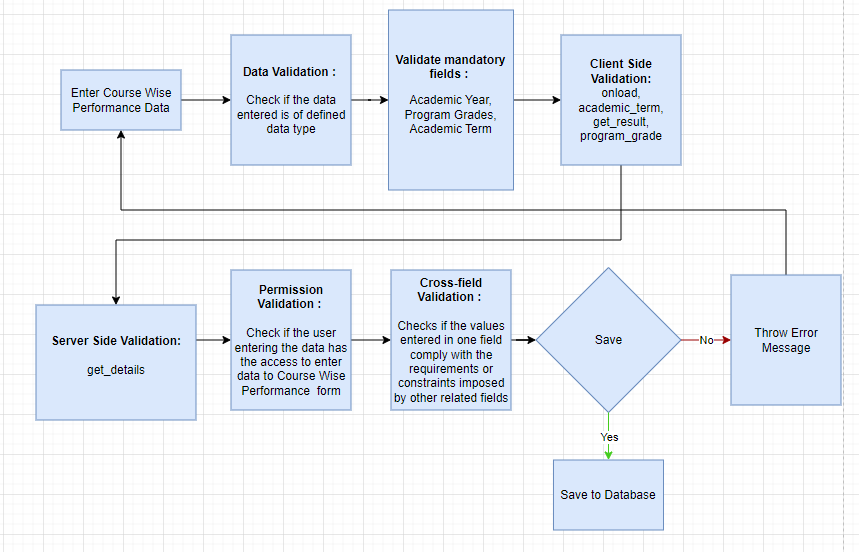


Figure : Course Wise Performance Process Flow

### Pre-requisites and Dependency

* Final Semester Result

## Module Wise Performance

This is the Report to measure the module wise performance of students.

### Use Case Diagram

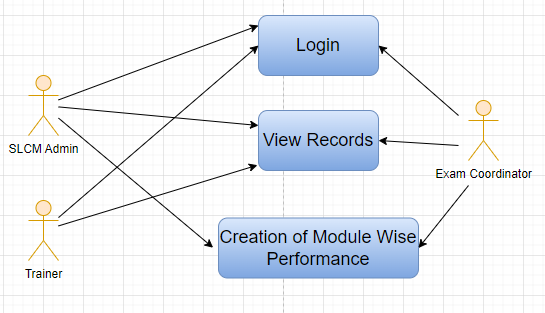


Figure : Module Wise Performance Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Module Wise Performance screen.

The following validation is done on client side:

* Mandatory checks for fields : Course Type, Programs, Academic Year, Academic Term, Course, Module Name
* Linked Fields : The Academic Year field is linked with Academic Year Screen. The Program field is linked with Program Screen. The Academic Term field is linked with Academic Term Screen. The Course Type field is linked with Course Type Screen.The Department field is linked with Department Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Module Wise Performance:

* SubjectWisePerformance class (inherits from Document): This class represents a document type and is intended to store subject-wise performance details. It inherits from the base Document class and defines functionalities to retrieve and analyze assessment results for specific academic terms and programs.
* get\_details method within the SubjectWisePerformance class: This method is decorated with frappe.whitelist() to allow it to be accessed from the frontend. It retrieves assessment results from the database based on specified criteria such as programs and academic term. It calculates statistics such as the number of students who appeared for assessments, the number who passed, and the pass percentage for each course. It handles the scenario where the denominator is zero (ZeroDivisionError) by assigning a pass percentage of 0. If no results are found for the specified criteria, it generates a message indicating the absence of results for a particular course and academic term.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Department | Link | Links to Department Screen |  |  |  |
| 2 | Course Type | Link | Links to Program Grade Screen | Yes |  |  |
| 3 | Programs | Link | Links to Programs Screen | Yes |  |  |
| 4 | Academic Year | Link | Links to Academic Year table | Yes |  |  |
| 5 | Academic Term | Link | Links to Academic Term Screen | Yes |  |  |
| 6 | Get Result | Button |  |  |  |  |
| 7 | Module Pass Percentage | Table | Links to Course Pass Table |  |  |  |
| 8 | Status | Data |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Course Pass Percentage** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Course | Link | Links to Course Screen | Yes |  |  |
| 2 | Module Name | Data |  | Yes |  |  |
| 3 | Module Code | Data |  | Yes |  |  |
| 4 | Enrolled | Int |  |  |  |  |
| 5 | Absent | Int |  |  |  |  |
| 6 | Appeared | Int |  |  |  |  |
| 7 | Failed | Int |  |  |  |  |
| 8 | Passed | Int |  |  |  |  |
| 9 | Pass % | Percent |  |  |  |  |

### Processes After Form Submission

* This screen has no workflow as the form is not submittable.

### Process Flow:

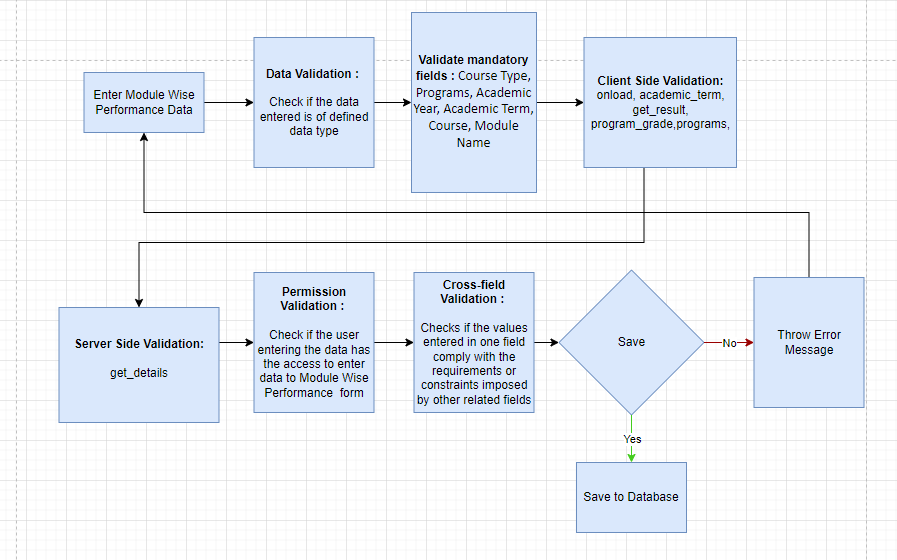


Figure : Module Wise Performance Process Flow

### Pre-requisites and Dependency

* Final Semester Result

## Cumulative Marksheet

Cumulative Marksheet is a compilation of all the semesters with their grades in a single certificate. The Cumulative marksheet will have a list of subjects, grades obtained and CGPA (Cumulative Grade Point Average). And also details of students, such as roll number, name, age and date of birth, program enrollment.

### Use Case Diagram

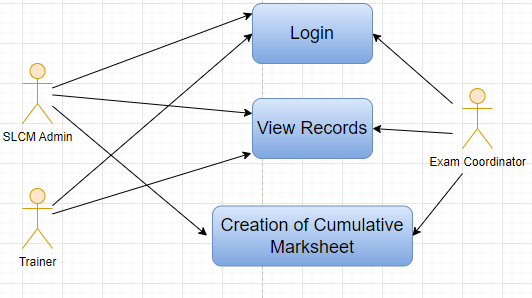


Figure : Cumulative Marksheet Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Cumulative Marksheet screen.

The following validation is done on client side:

* Mandatory checks for fields : None
* Linked Fields : The Student field is linked with Student Screen. The Year of Admission field is linked with Academic Year Screen. The Year of Completion field is linked with Academic Year Screen. The PROGRAMME field is linked with Programs Screen. The Programs field is linked with Programs Screen. The Semester field is linked with Semester Screen. The Course field is linked with Course Screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Cumulative Marksheet:

* CumulativeMarksheet class (inherits from Document): This class represents a document type in the Frappe framework intended to store and manage cumulative marksheet data. It inherits from the base Document class and defines methods to validate, map, set results, and retrieve student details.
* validate method within the CumulativeMarksheet class: This method serves as a validation step for the cumulative marksheet document. It ensures there are no duplicate submissions or saves, maps semester order fields, validates missing fields, and sets the result status (pass/fail) based on the percentage obtained.
* on\_submit method within the CumulativeMarksheet class: This method is triggered when the document is submitted. It reuses certain validation logic from the validate method to ensure there are no duplicate submissions.
* get\_grade method within the CumulativeMarksheet class: This method queries the database for a student's grading scale and calculates the corresponding grade based on the student's secured marks and total marks.
* map\_fields method within the CumulativeMarksheet class: This method populates the semester\_order field within the cumulatice\_grades\_item table by querying the database for semester order information from the "Program" table.
* validate\_duplicate\_for\_save method within the CumulativeMarksheet class: This method checks for duplicate marksheet entries for a student that are in a draft (unsaved) state and share the same student, program, and year of completion.
* validate\_duplicate\_for\_submit method within the CumulativeMarksheet class: Similar to the previous method, this one checks for duplicate entries but for marksheet documents that are submitted (in a saved state).
* validate\_missing\_fields method within the CumulativeMarksheet class: This method ensures that the semester\_order field is not empty within the cumulatice\_grades\_item table and throws an error message if any such instance is found.
* set\_result method within the CumulativeMarksheet class: This method calculates the pass or fail status for each course entry based on the secured percentage and sets the result\_p\_f field accordingly.
* get\_student\_details method within the CumulativeMarksheet class: This method retrieves student enrollment and assessment result details from the database and populates the cummulative\_courses\_item and cumulatice\_grades\_item tables within the document. It calculates cumulative percentages, totals, and grades for the student's courses.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field | Fetched from Student master screen |  |  |  |
| 2 | Student Name | Text Field | Auto fetch based on Student |  |  |  |
| 3 | Roll No | Text Field | Auto fetch based on Student |  |  |  |
| 4 | BRANCH | Text Field |  |  |  |  |
| 5 | Registration Number | Text Field | Auto fetch based on Student |  |  | (D) |
| 6 | COMPLETED ON | Text Field |  |  |  |  |
| 7 | Year of Admission | Link Field | Fetched from Academic Year master screen |  |  |  |
| 8 | Year of Completion | Link Field | Linked with Academic Year master screen |  |  |  |
| 9 | PROGRAMME | Link Field | Fetched from Programs master screen |  |  | (R)COURSE NAME |
| 10 | School of | Text Field |  |  |  |  |
| 11 | Year of Admission | Text Field |  |  |  |  |
| 12 | **Cumulative Courses Item** | Table | Child Table: Cumulative Courses Item |  | These table is describe below |  |
| 13 | **Cumulative Grades Item** | Table | Child Table: Cumulative Grades Item |  | These table is describe below |  |
| 14 | Overall Grade | Float |  |  |  |  |
| 15 | RESULT | Text Field | User Input |  |  |  |
| 16 | Signature of Examiner | Attach image | 1. On attachment, a dialogue box will popup where the user can attach the files, take the picture with the help camera or can link to any URL. 2. If photos or images attached then the size will be maximum 200 kb |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cumulative Courses Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Programs | Link Field | Fetched from Programs master screen |  |  |  |
| 2 | Semester | Link Field | Fetched from Program master screen |  |  |  |
| 3 | Course | Link Field | Fetched from Course master screen |  |  | (R) Module |
| 4 | Course Code | Text Field | Auto fetch based on Course |  |  | (R) Module Code |
| 5 | Course Name | Text Field | Auto fetch based on Course |  |  | (R) Module Name |
| 6 | CR | Text Field |  |  |  | ( D ) |
| 7 | GR | Text Field |  |  |  | ( D ) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cumulative Grades Item** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Semester | Link Field | Fetched from Program master screen |  |  |  |
| 2 | Semester Order | Drop down | 1ST SEM  2ND SEM  3RD SEM  4TH SEM  5TH SEM  6TH SEM  7TH SEM  8TH SEM  9TH SEM  10TH SEM |  |  |  |
| 3 | Grade | Text Field |  |  |  | (D) These field will be deleted |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

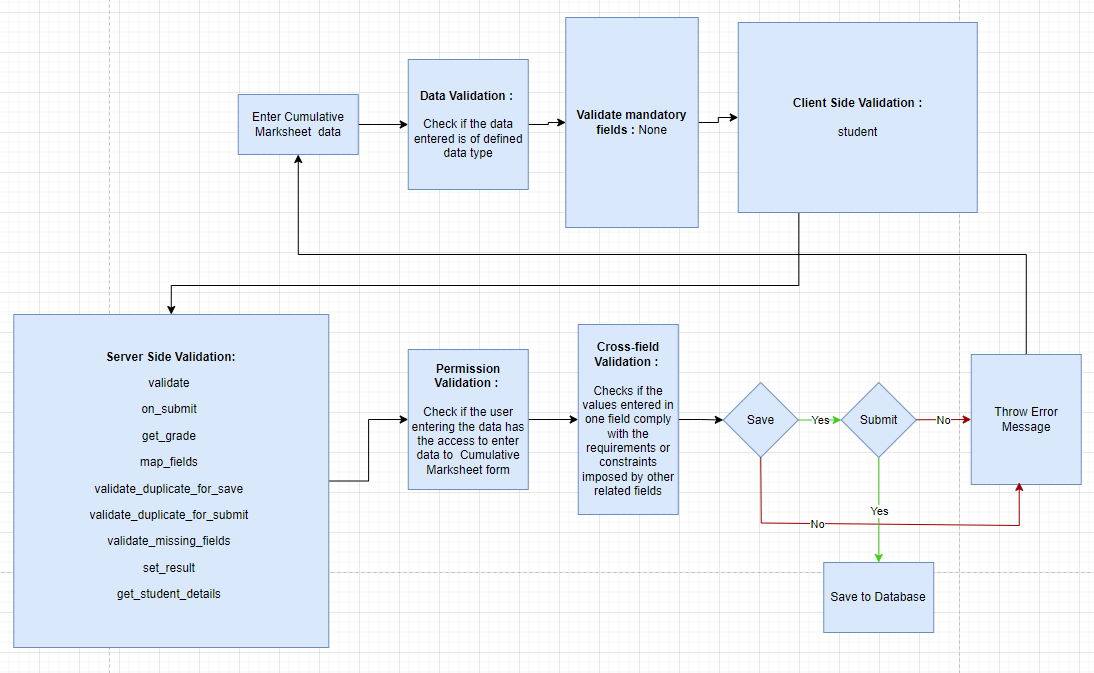


Figure : Cumulative Marksheet Process Flow

### Pre-requisites and Dependency

* Final Exam Result

## Student Backpaper Tracking

Back Paper Tracking is used to actively track and maintain the history of the whole Examination process of a student who has failed in one or more modules.

### Use Case Diagram

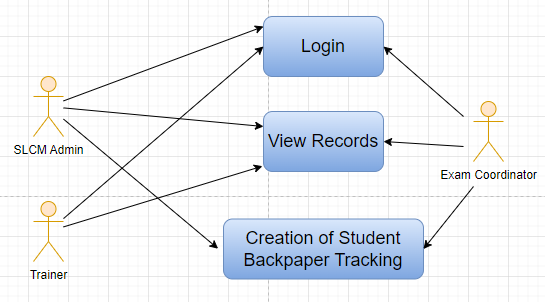


Figure : Student Backpaper Tracking Use Case Diagram

### Design of Workflow

* Not Applicable

### Validations

The data entered on this page will undergo a 2 step validation (I.e. client side and server side validation).

**Client Side Validation**

Client-side validation refers to the process of validating user input directly on the client's web browser before saving it to the server for further processing. Following are some client side validations done for Student Backpaper Tracking screen.

The following validation is done on client side:

* Mandatory checks for fields : Program Grade, Academic Year, Academic Term, Programs, Semester, Course
* Linked Fields : The Program Grade field is linked with the Program Grade master screen.The Academic Year field is linked with the Academic Year master screen.The Academic Term field is linked with the Academic Term master screen, and a filter will be applied based on the selected Academic Year.The Programs field is linked with the Programs master screen.The Semester field is linked with the Semester master screen, and filters will be applied based on the selected programs.The Course field is linked with the Course master screen.The Department field is linked with the Department screen.

**Server Side Validation :**

Server-side validation refers to the process of validating user input on the server to ensure data integrity, security, and adherence to business rules before accepting or processing the information.

The following are the server side validations for Student Backpaper Tracking:

* BackPaperTracker class (inherits from Document): This class represents a document type in the Frappe framework and is used to track instances of back papers or failed courses. It inherits from the base Document class, allowing the creation and management of back paper tracking records.
* get\_course function: This function is decorated with frappe.whitelist() to allow it to be accessed from the frontend. It takes several arguments including doctype, txt, searchfield, start, page\_len, and filters. It queries the database for a list of course names associated with a specific semester (filters.get("semester")) and returns this list to be displayed as autocomplete suggestions.
* get\_student function: This function is also decorated with frappe.whitelist() for frontend access. It takes course, academic\_year, academic\_term, and program as arguments and queries the database to retrieve evaluation result items where the result is marked as "F" (failed) for the specified course. For each such item, it searches for associated exam assessment results matching the provided academic year, academic term, and program. It then compiles a list of student records who failed the course and returns it as output.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Program Grade | Link Field | Fetched from Program Grade master screen | Yes |  | ( R ) Course Grade |
| 2 | Academic Year | Text Field | Fetched from Academic Year master screen | Yes |  |  |
| 3 | Academic Term | Link Field | Fetched from the Academic Term master screen,filter will be applied based on Academic Year | Yes |  |  |
| 4 | Programs | Text Field | Fetched from Programs master screen | Yes |  | ( R ) Course |
| 5 | Semester | Link Field | Fetched from Semester master screen,filtered will be applied based on selected programs | Yes |  |  |
| 6 | Course | Text Field | Fetched from Course master screen | Yes |  | ( R ) Module |
| 7 | Get Students | Button |  |  |  |  |
| 8 | **Backlog Details** | Table |  |  | This table is described below |  |
| 9 | Department | Link Field | Fetched from Department Screen |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Students** | | | | | | |
| **ID** | **Field Label** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field | Failed student will be fetched clicking on Get Student Button |  |  |  |
| 2 | Student Name | Text Field | Auto fetch based on Student |  |  |  |
| 3 | Result | Text Field | Fail |  |  |  |

### Processes After Form Submission

* This section is not applicable since the form has no process after submission.

### Process Flow:

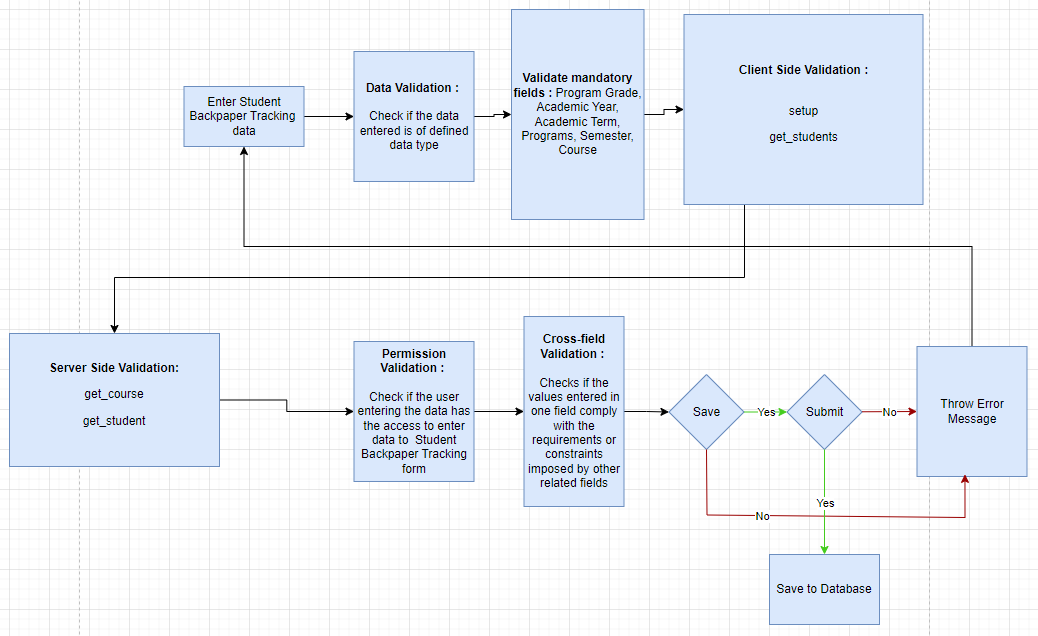


Figure : Student Backpaper Tracking Process Flow

### Pre-requisites and Dependency

* Final Exam Result

# Tools And Technologies

This section explains the tools that are used to build this design document

* Draw.io - it is an online tool used to develop the flowcharts used in this document.
* WPS office - it is a text editor that has been used to write and assemble this document

# Non-Functional Requirements

## Performance

Specify performance requirements, including response times, transaction throughput, and system scalability

## Reliability

Define the reliability requirements, such as availability, fault tolerance, and disaster recovery

## Usability

Describe usability requirements, including user-friendly interfaces, clear error messages, and intuitive workflows

## Compatibility

Specify compatibility requirements with different operating systems, web browsers, and devices

## Security

Security requirements, including authentication, access control, data encryption, and secure communication protocols

* Authentication: The system will provide robust and secure authentication mechanisms to ensure that only authorized users can access the system and its resources
* Authorization: The system will enforce access controls to restrict users' actions based on their roles and privileges, ensuring that users can only access the resources they are authorized to use
* Data Encryption: Sensitive data, both at rest and in transit, will be encrypted to protect it from unauthorized access or interception
* Secure Communication: Secure communication protocols (e.g., HTTPS, TLS) should be used to protect data exchanged between clients and servers

## Compliance

Specify any legal or regulatory compliance requirements, such as GDPR, PCI DSS, or other industry-specific standards

## Documentation

Describe the documentation requirements, including user manuals, developer guides, and API documentation

## Security: User access and authorization

The screens and the data in them is made secure by creating roles with different permissions and then assigning those roles the appropriate users. For this screen, following are the roles and their corresponding permissions:

**Exam Type**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Assessment Component**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Grading Scale**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Exam Declaration**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Module Wise Exam Group**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Exam Evaluation Plan**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Exam Paper Setting**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Component Wise Evaluation**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Component Wise Evaluation Tool**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Component Wise Reevaluation**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Component Wise Reevaluation Tool**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Final Semester Result**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Final Semester Result Tool**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Cumulative Marksheet Tool**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Final Semester Result Report**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Course Wise Performance**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Module Wise Performance**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

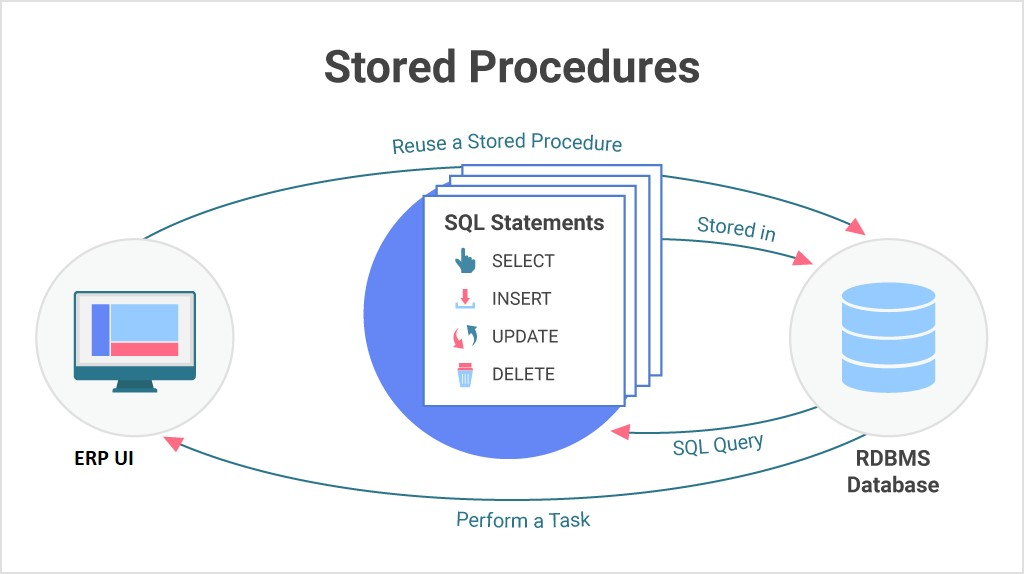
**Cumulative Marksheet**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Student BackPaper Tracking**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | SLCM Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Trainer | Yes | No | No | No | NA | NA | NA |
| 3 | Exam Coordinator | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

# Database Design



**Prototype**

# Prototype

The screen samples that have been derived based on the requirements gathered from the users / SME’s of the **Examination** module

## Exam Type

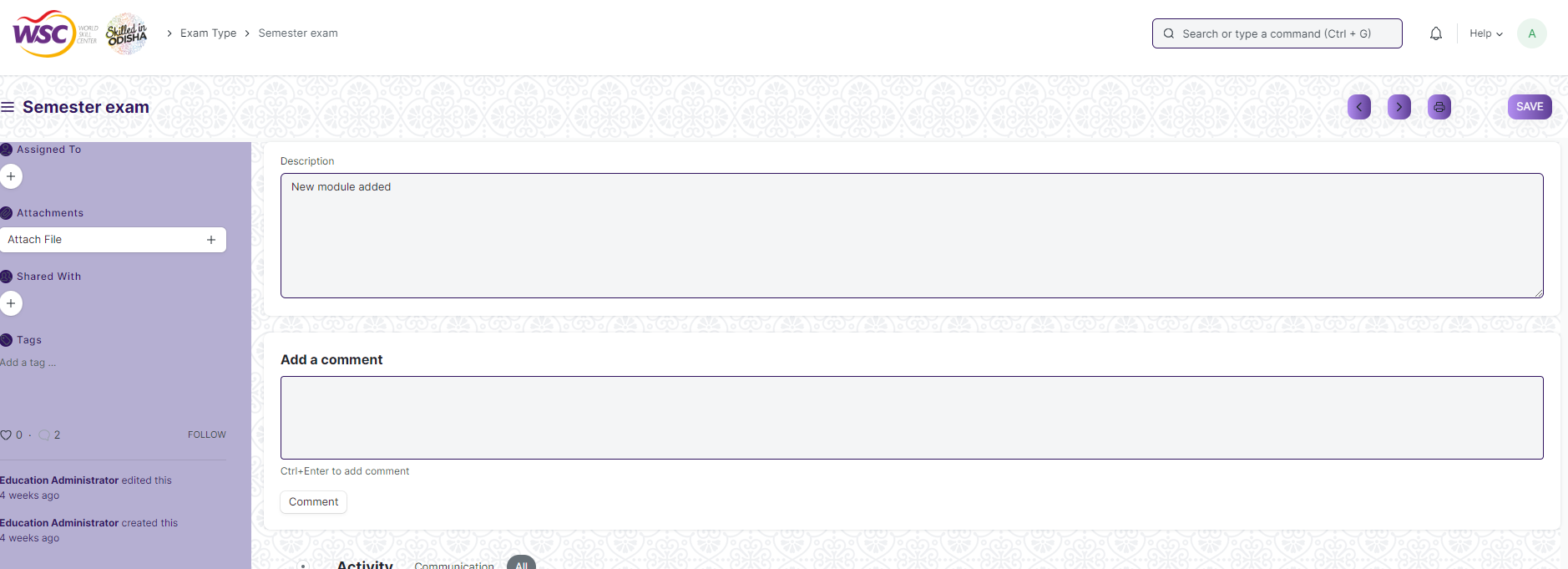


Figure : Exam Type

## Assessment Component

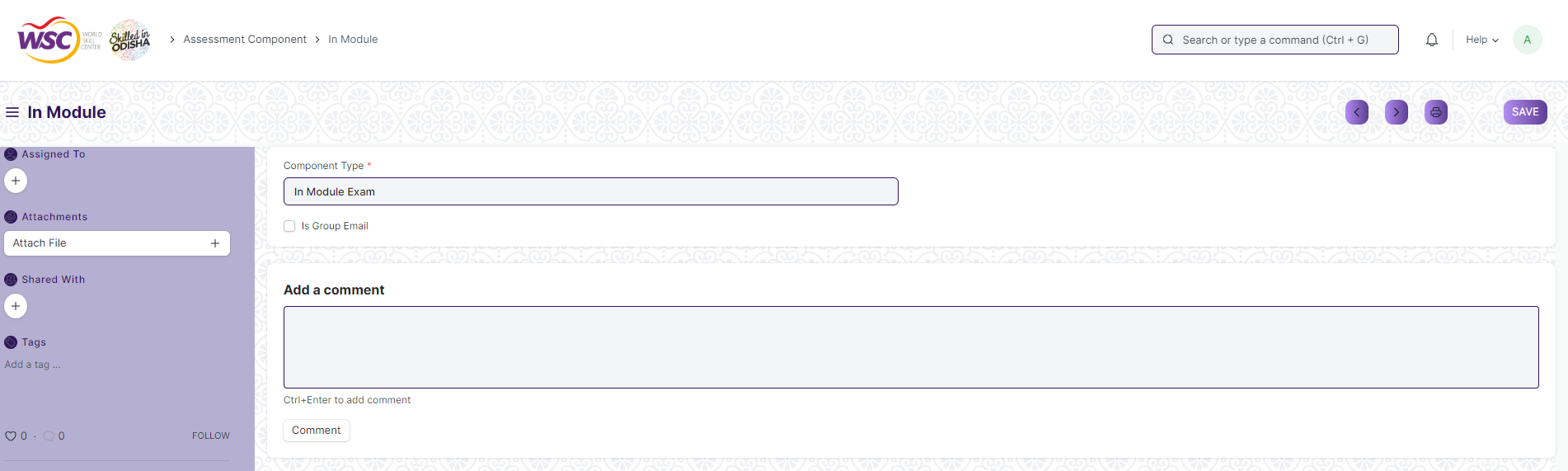


Figure : Assessment Component

## Grading Scale

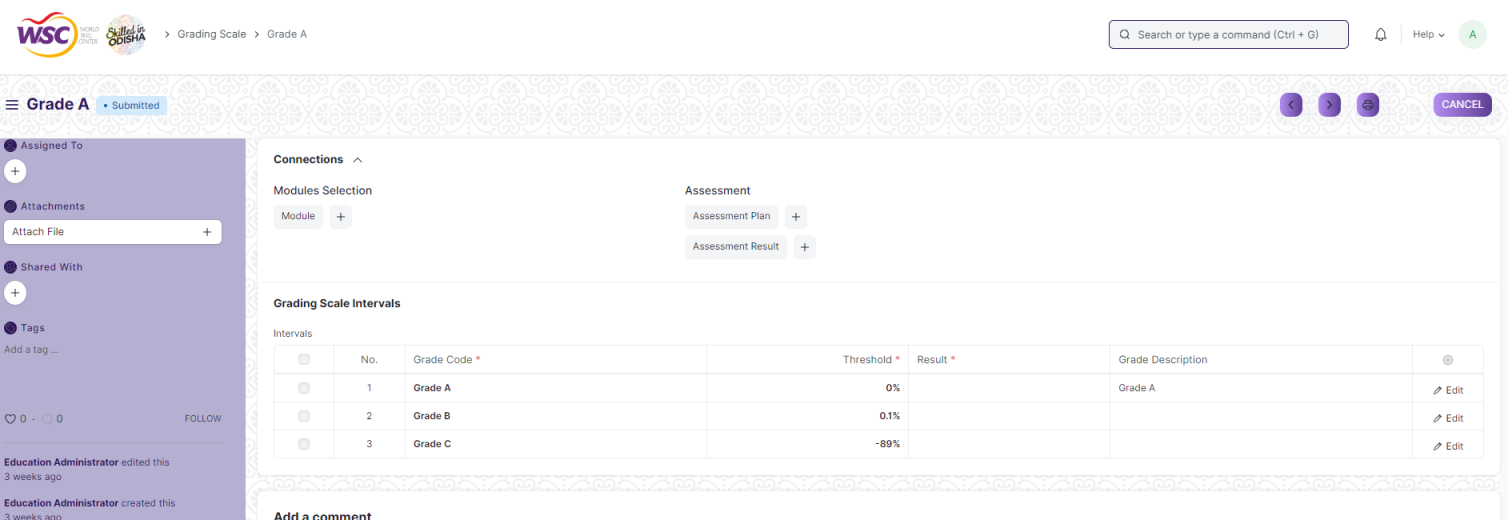


Figure : Grading Scale

## Exam Declaration

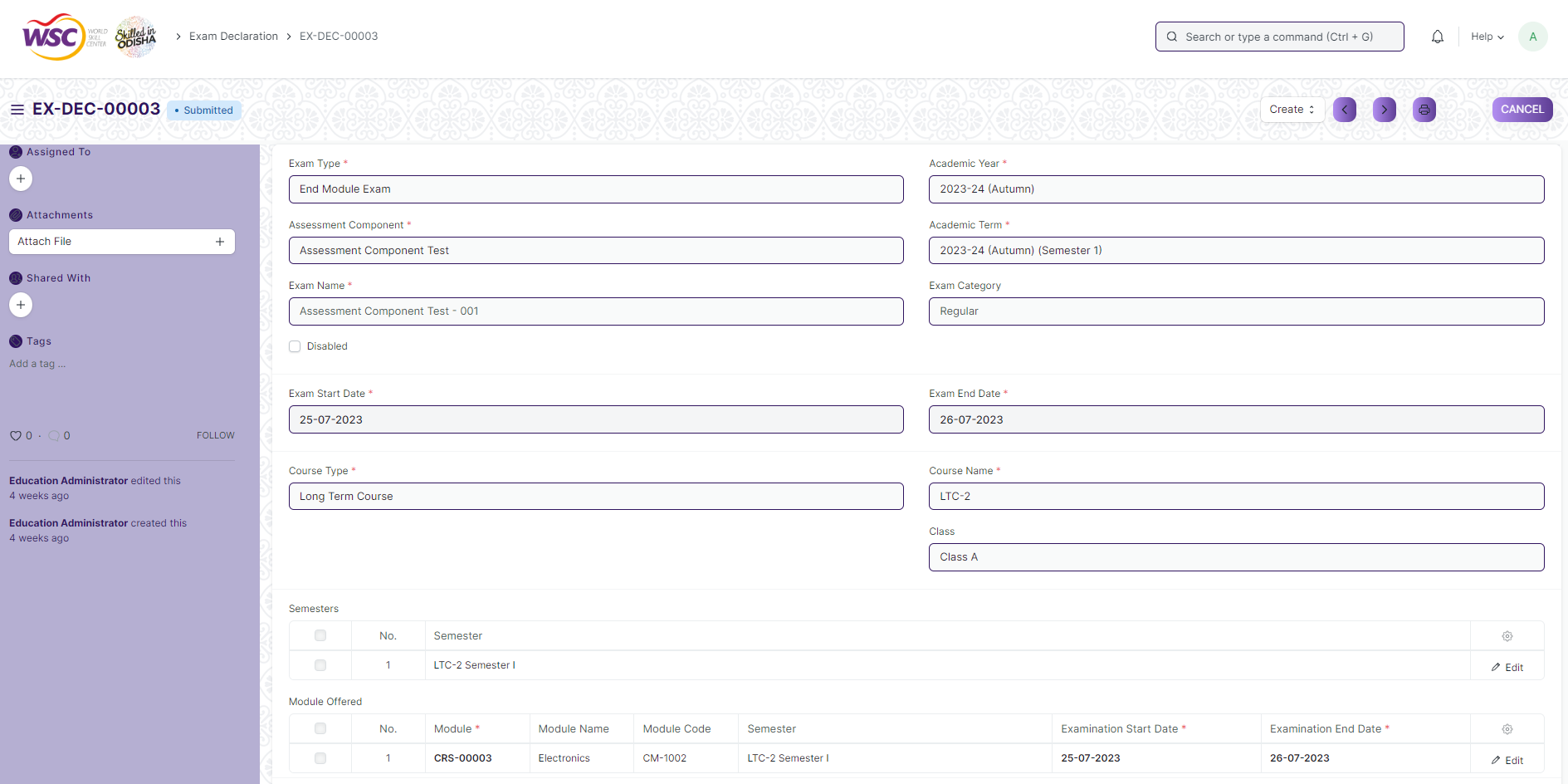


Figure : Exam Declaration

## Module Wise Exam Group



Figure : Module Wise Exam Group

## Exam Evaluation Plan

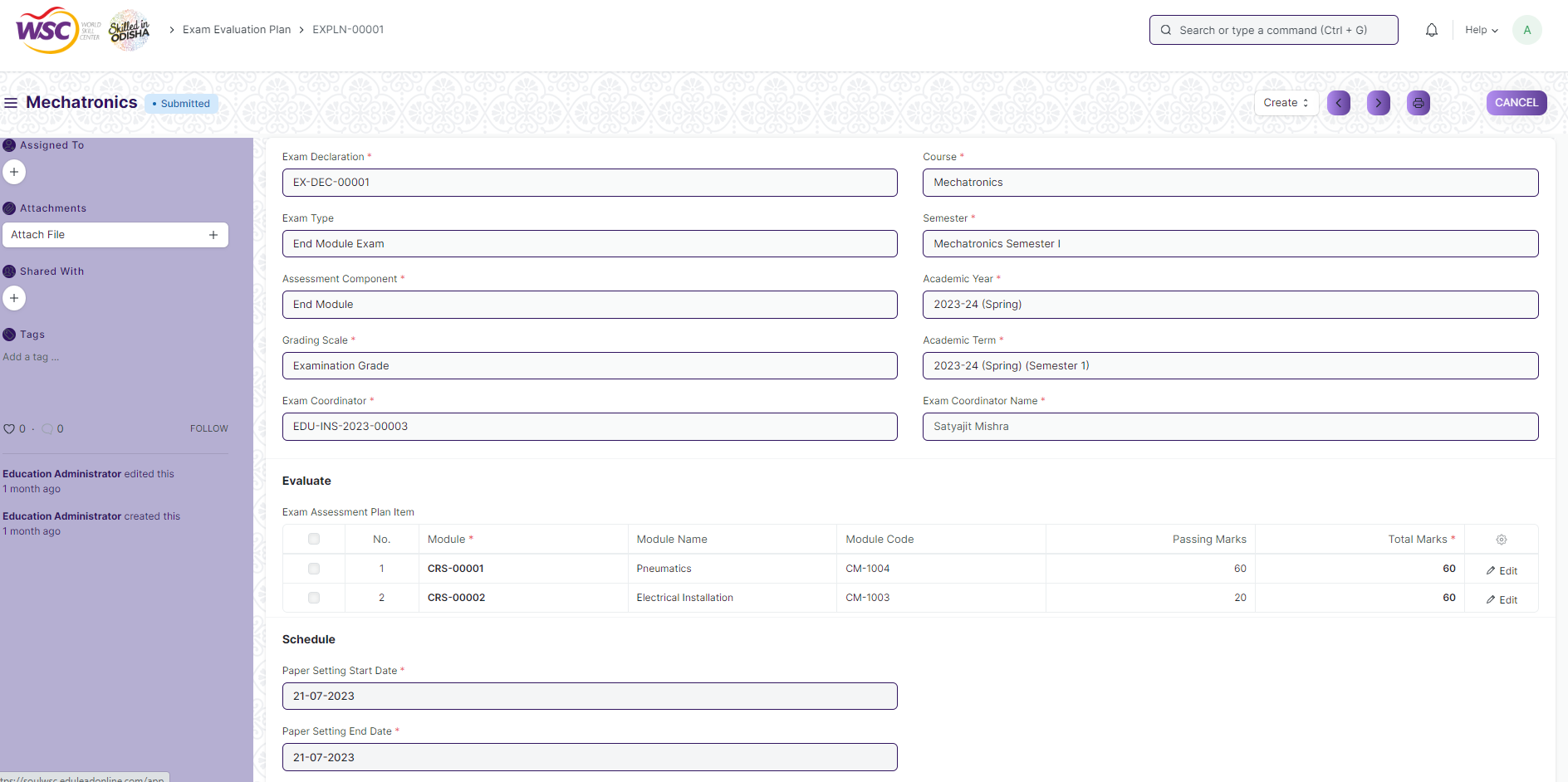


Figure : Exam Evaluation Plan

## Exam Paper Setting

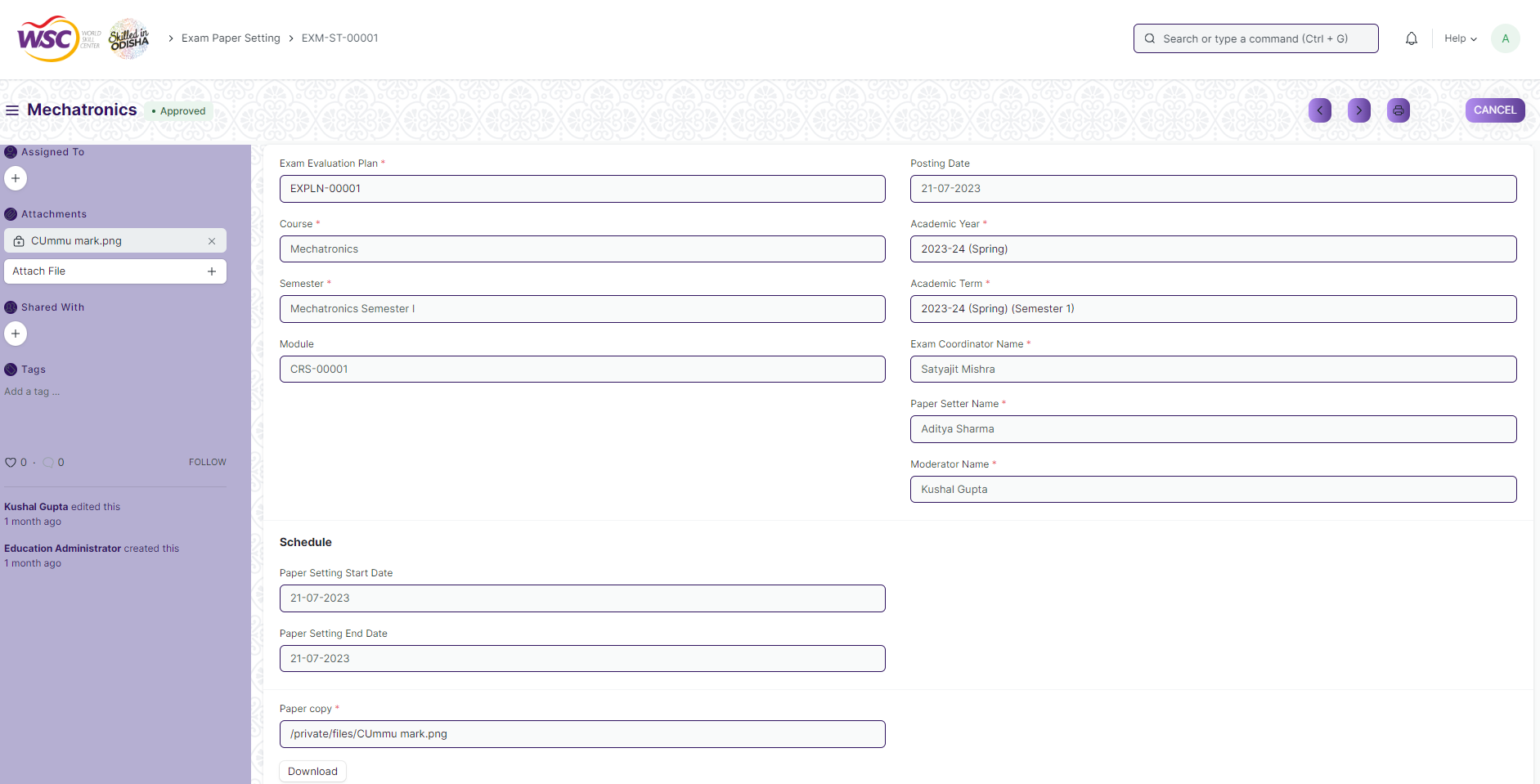


Figure : Exam Paper Setting

## Component Wise Evaluation

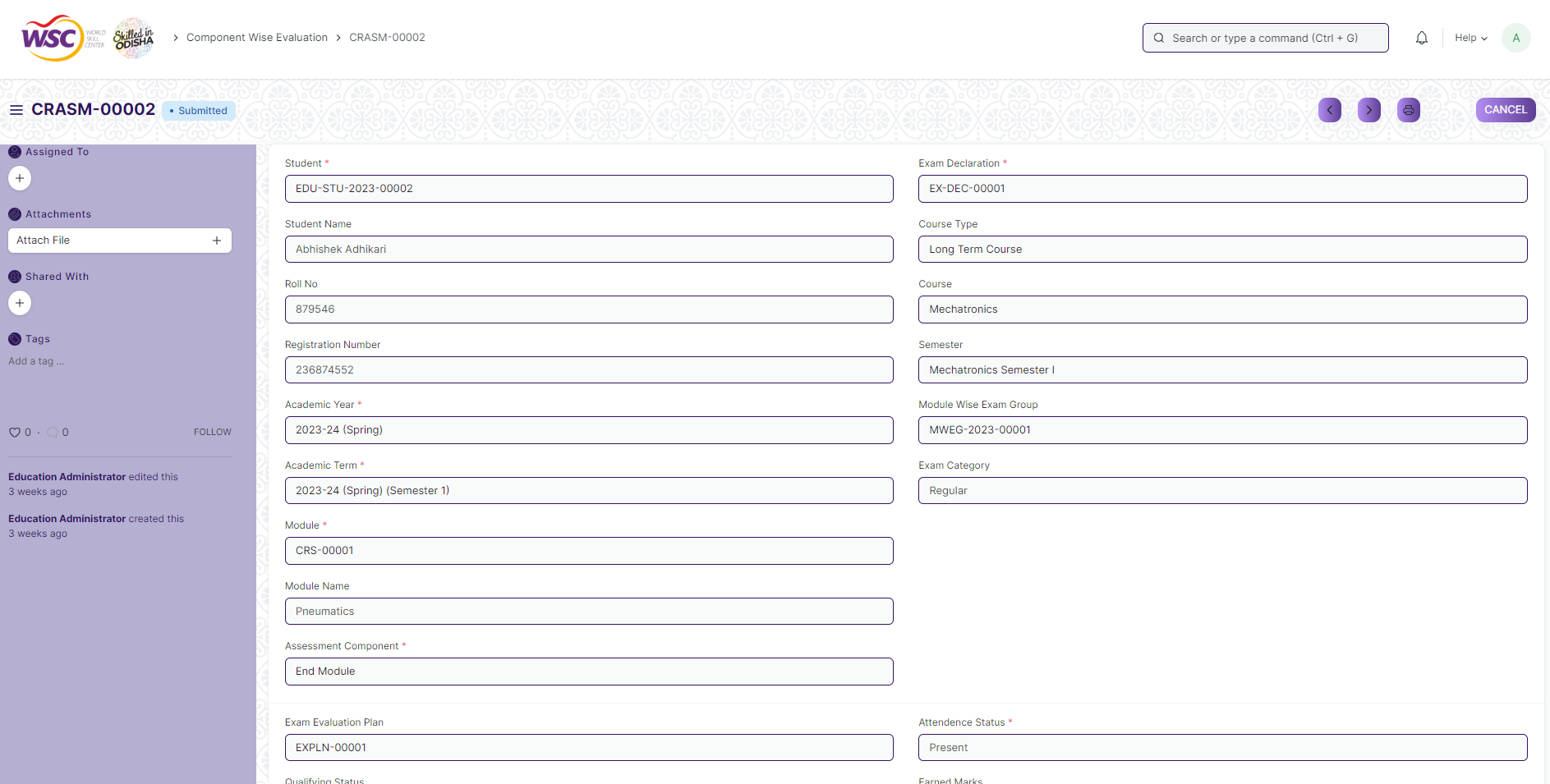


Figure : Component Wise Evaluation

## Component Wise Evaluation Tool

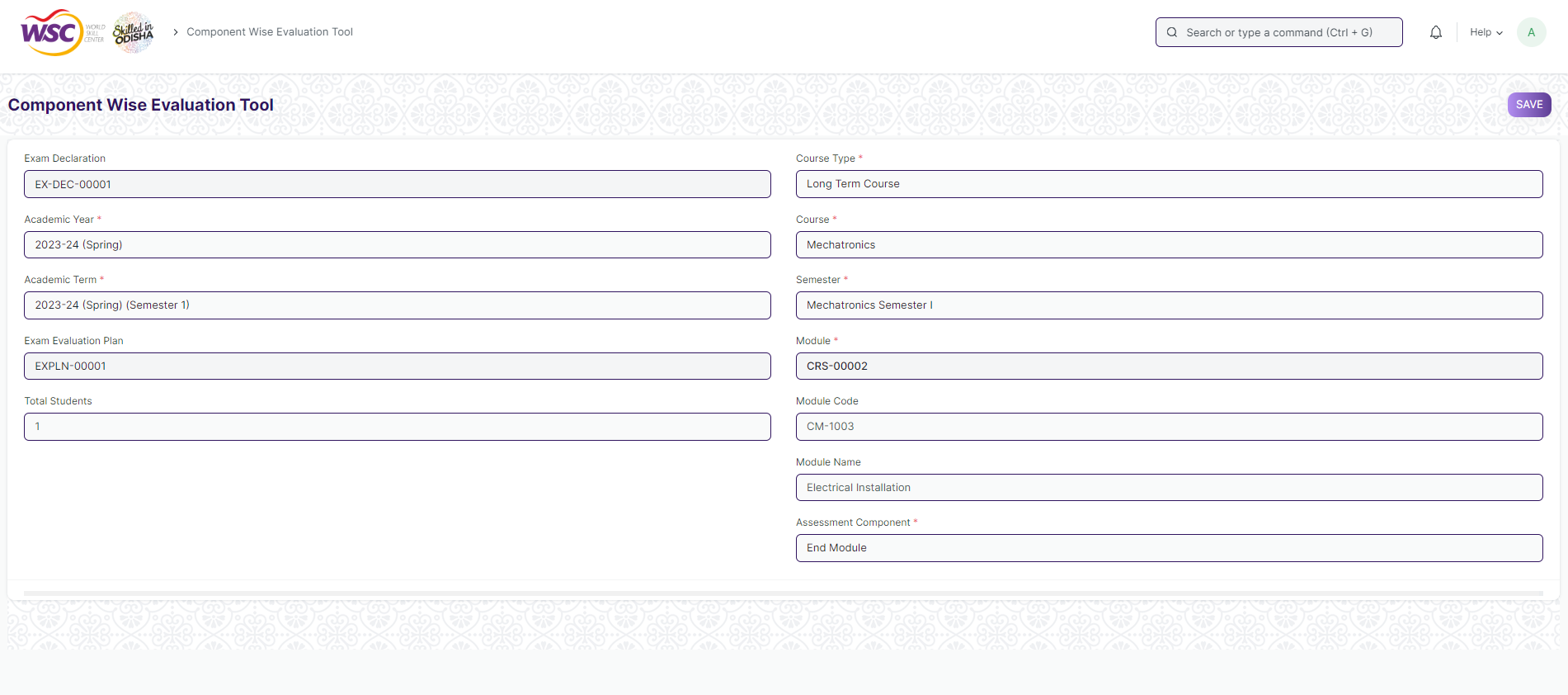


Figure : Component Wise Evaluation Tool

## Component Wise Reevaluation

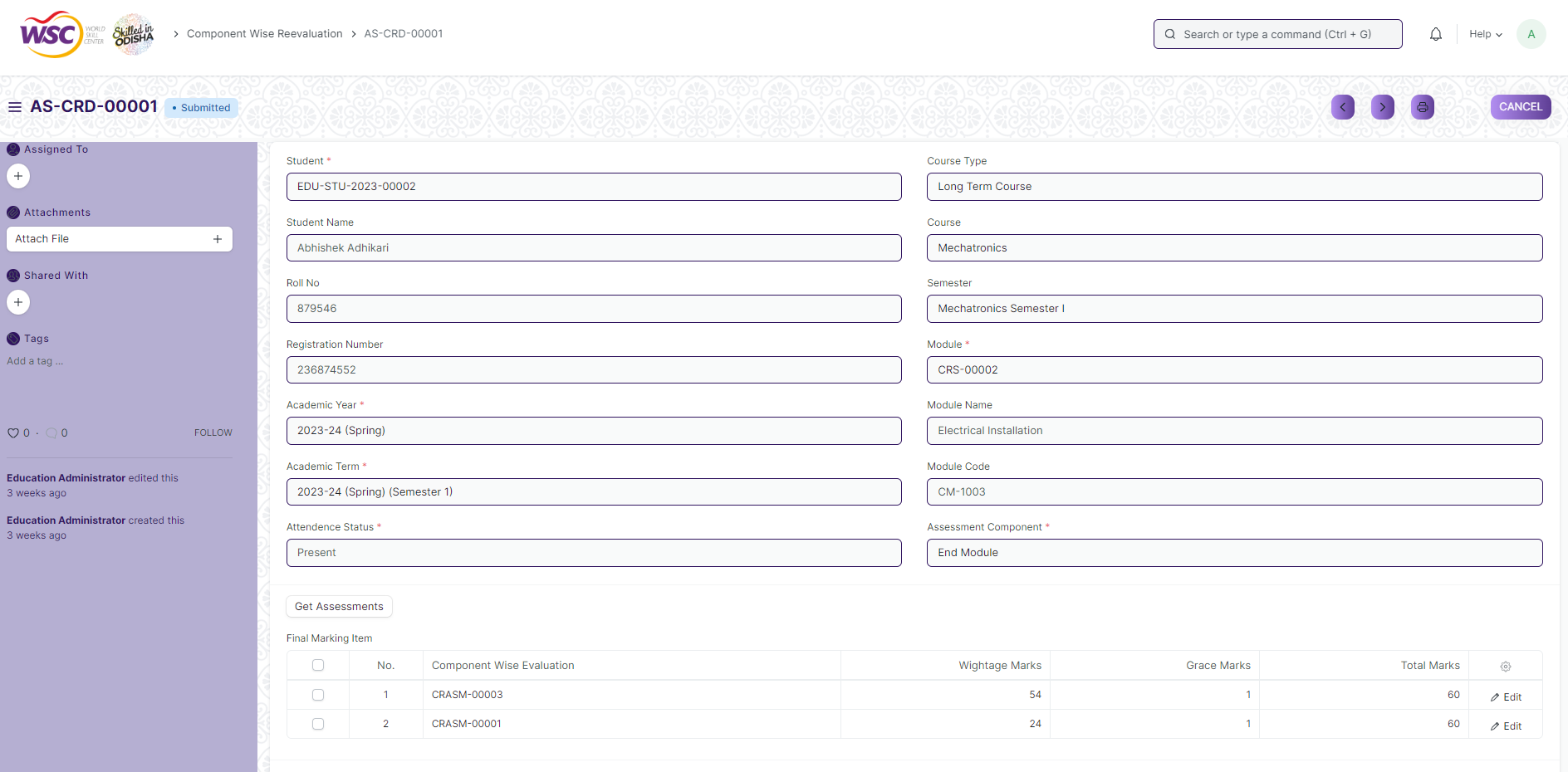


Figure : Component Wise Reevaluation

## Component Wise Reevaluation Tool

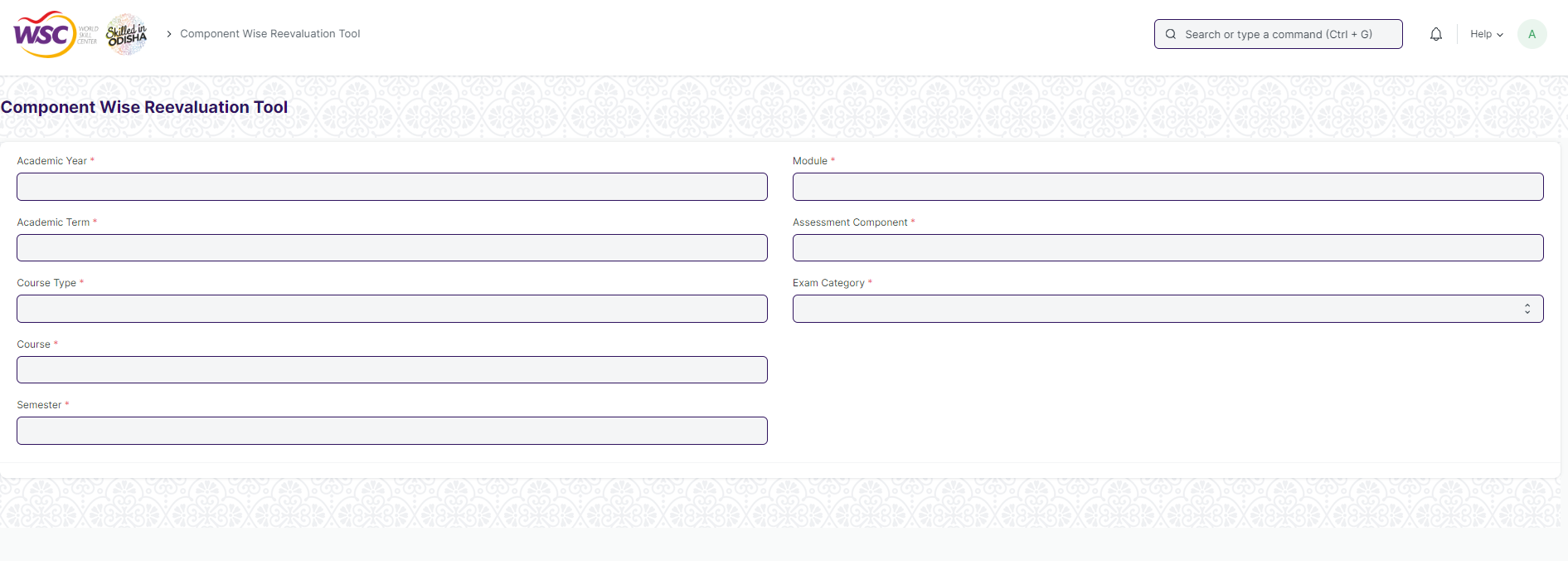


Figure : Component Wise Reevaluation Tool

## Final Semester Result

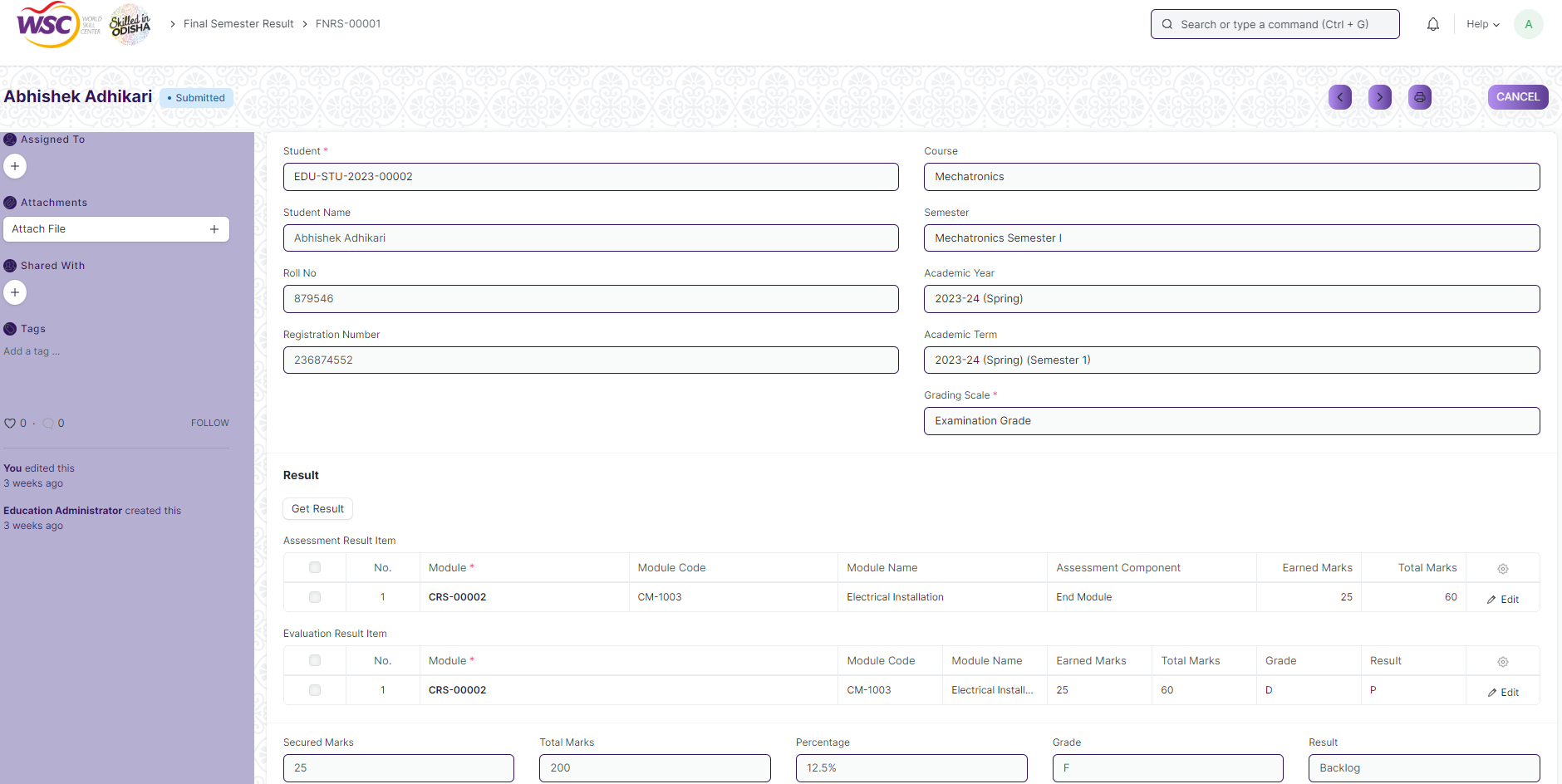


Figure : Final Semester Result

## Final Semester Result Tool

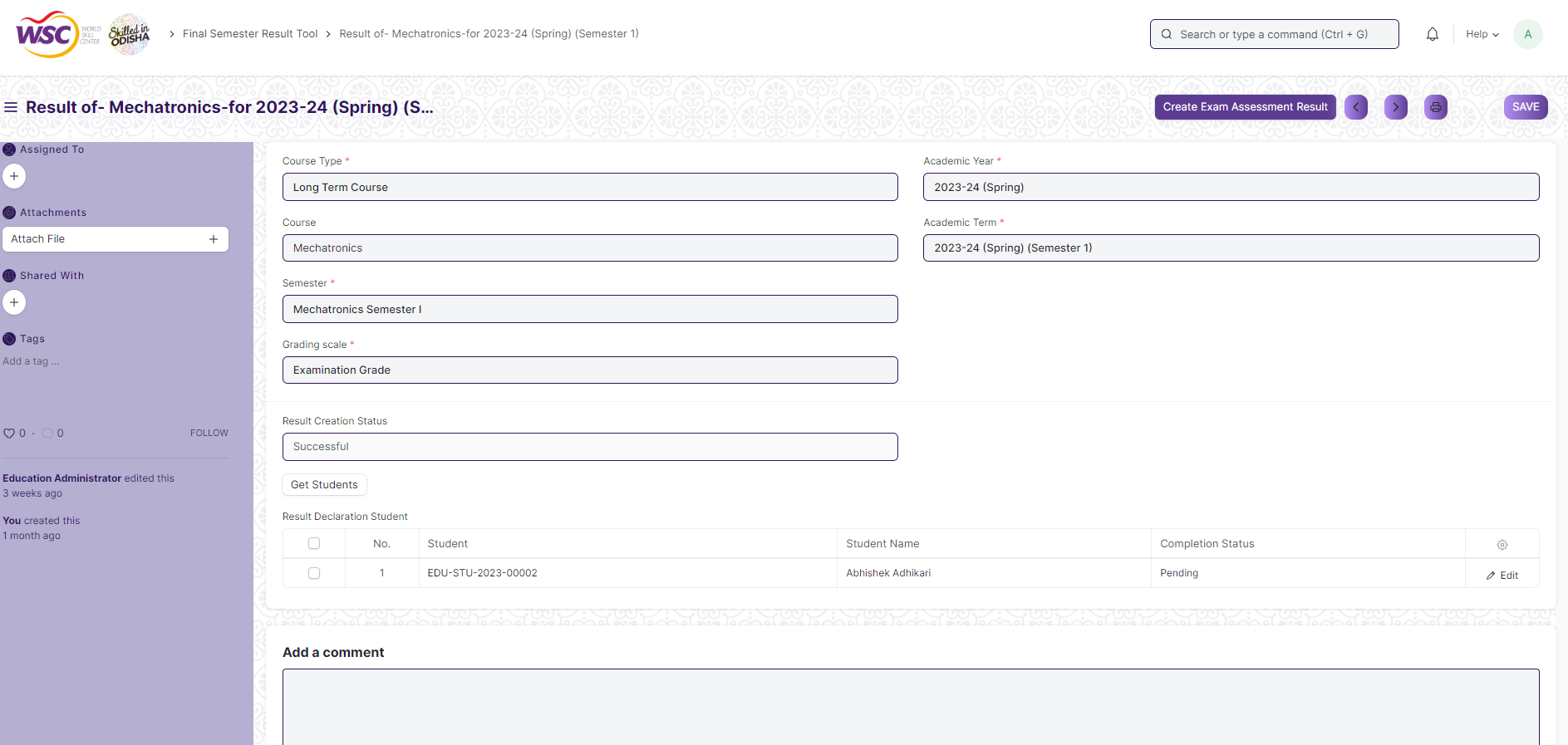


Figure : Final Semester Result Tool

## Cumulative MarkSheet Tool



Figure : Cumulative Marksheet Tool

## Final Semester Result Report

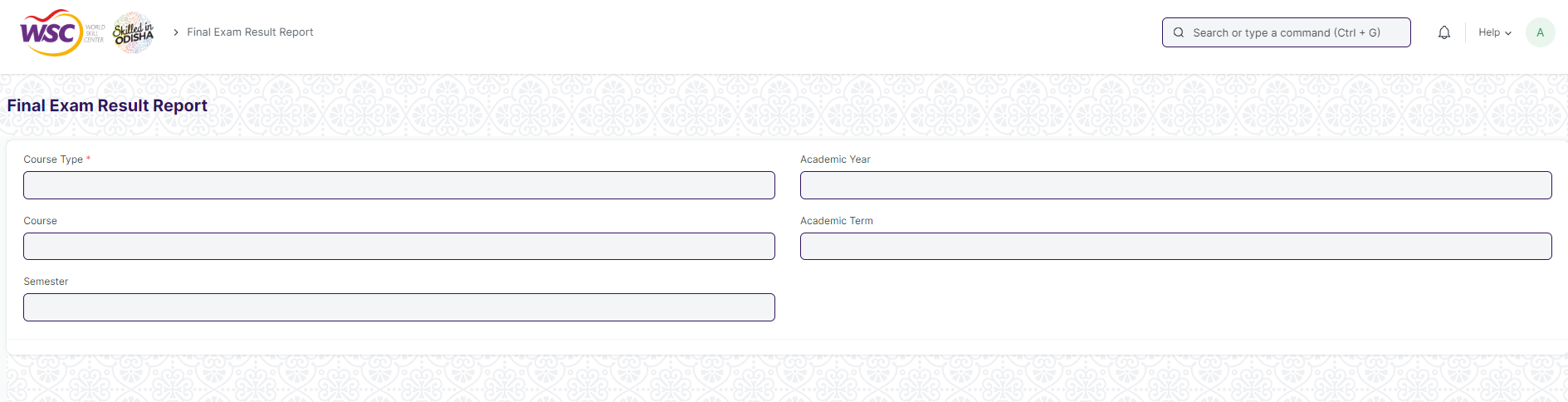


Figure : Final Semester Result Report

## Course Wise Performance

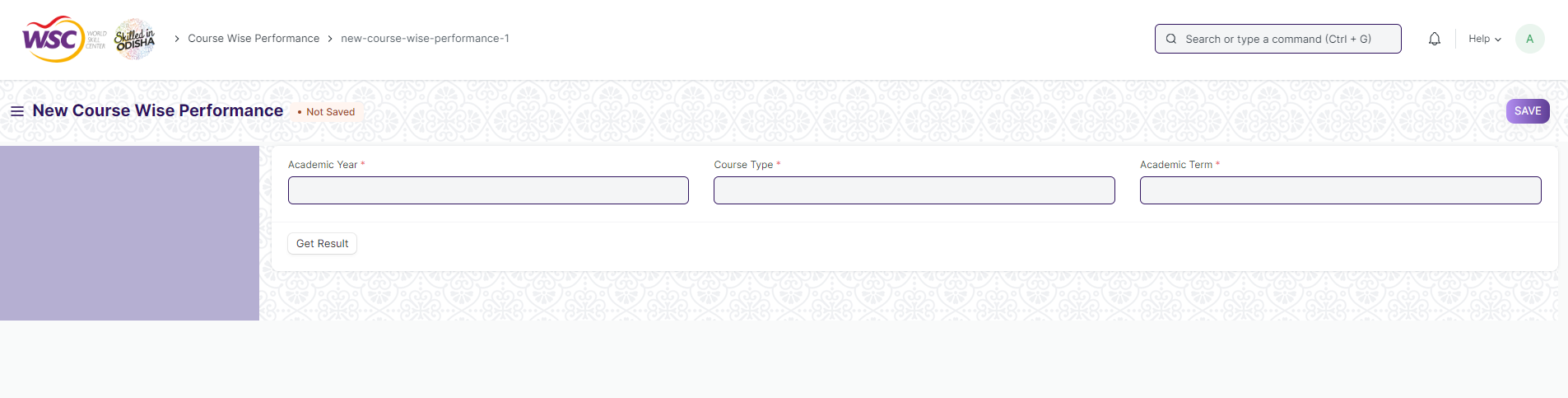


Figure : Course Wise Performance

## Module Wise Performance

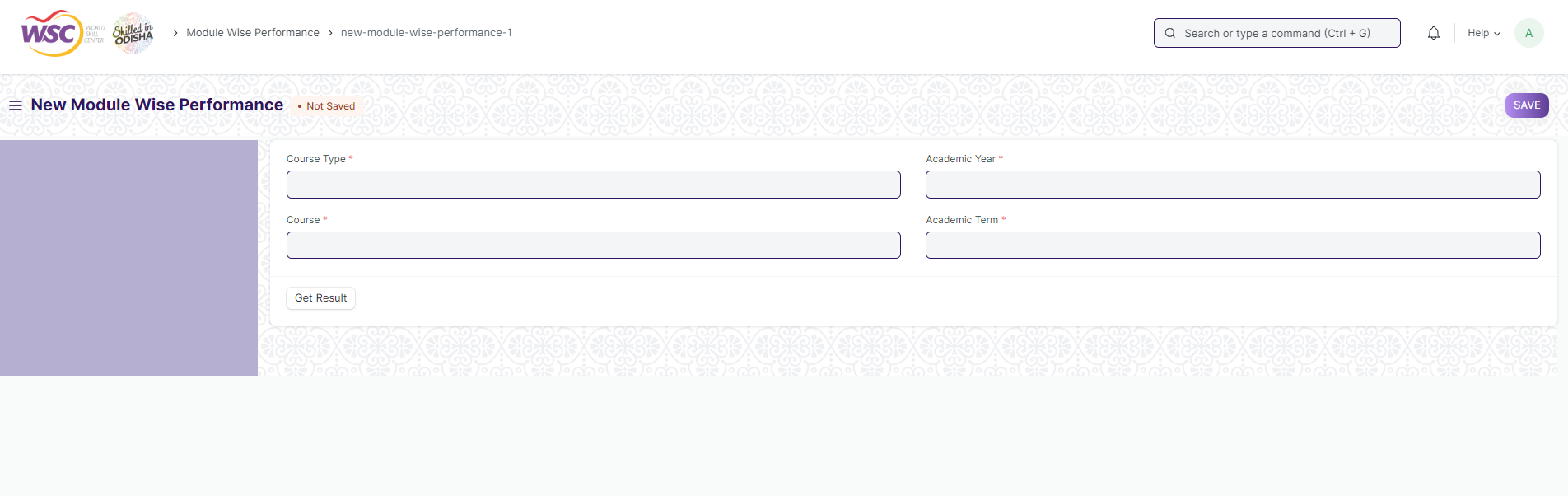


Figure : Module Wise Performance

## Cumulative MarkSheet

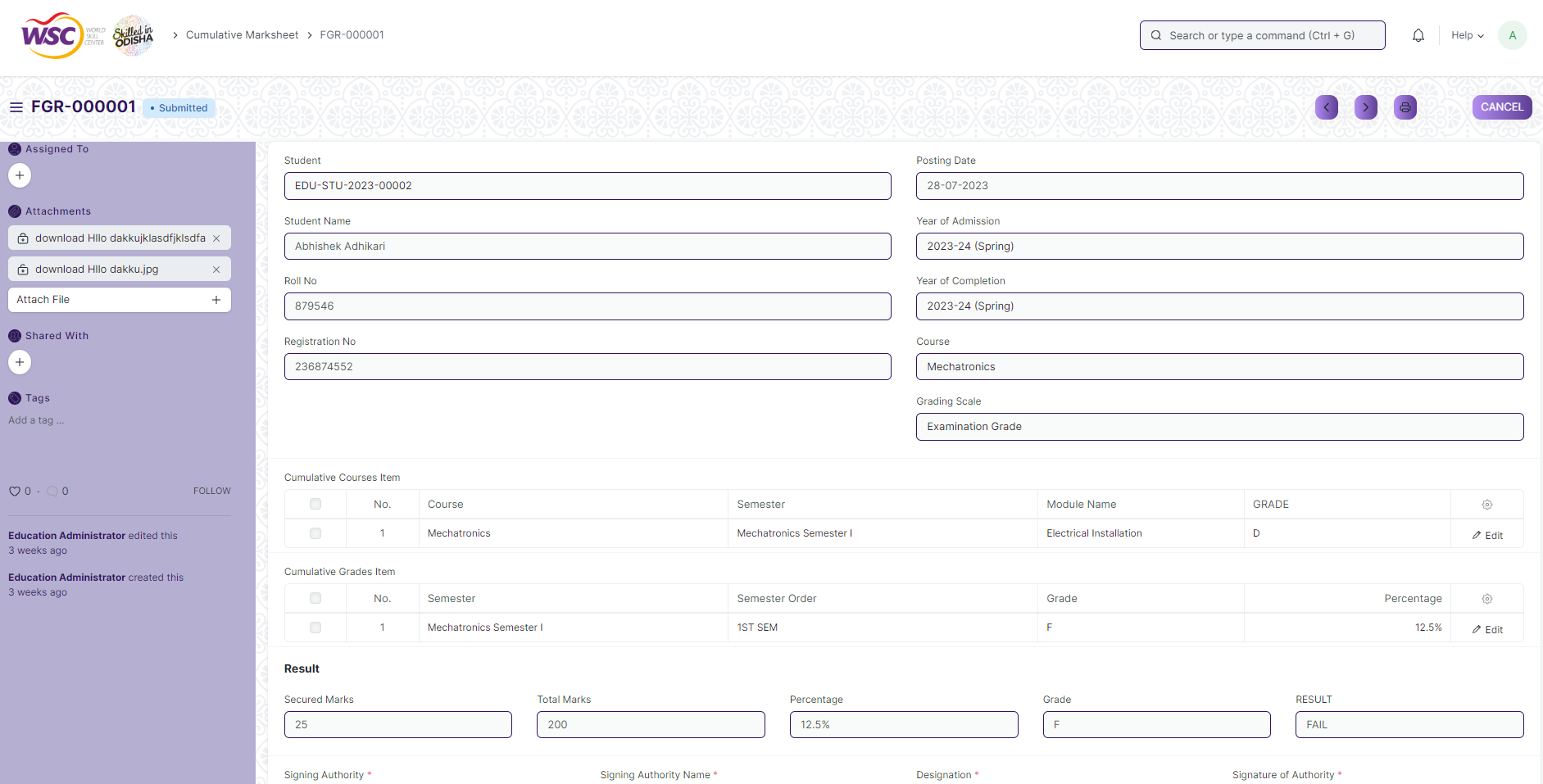


Figure : Cumulative Marksheet

## Student BackPaper Tracking

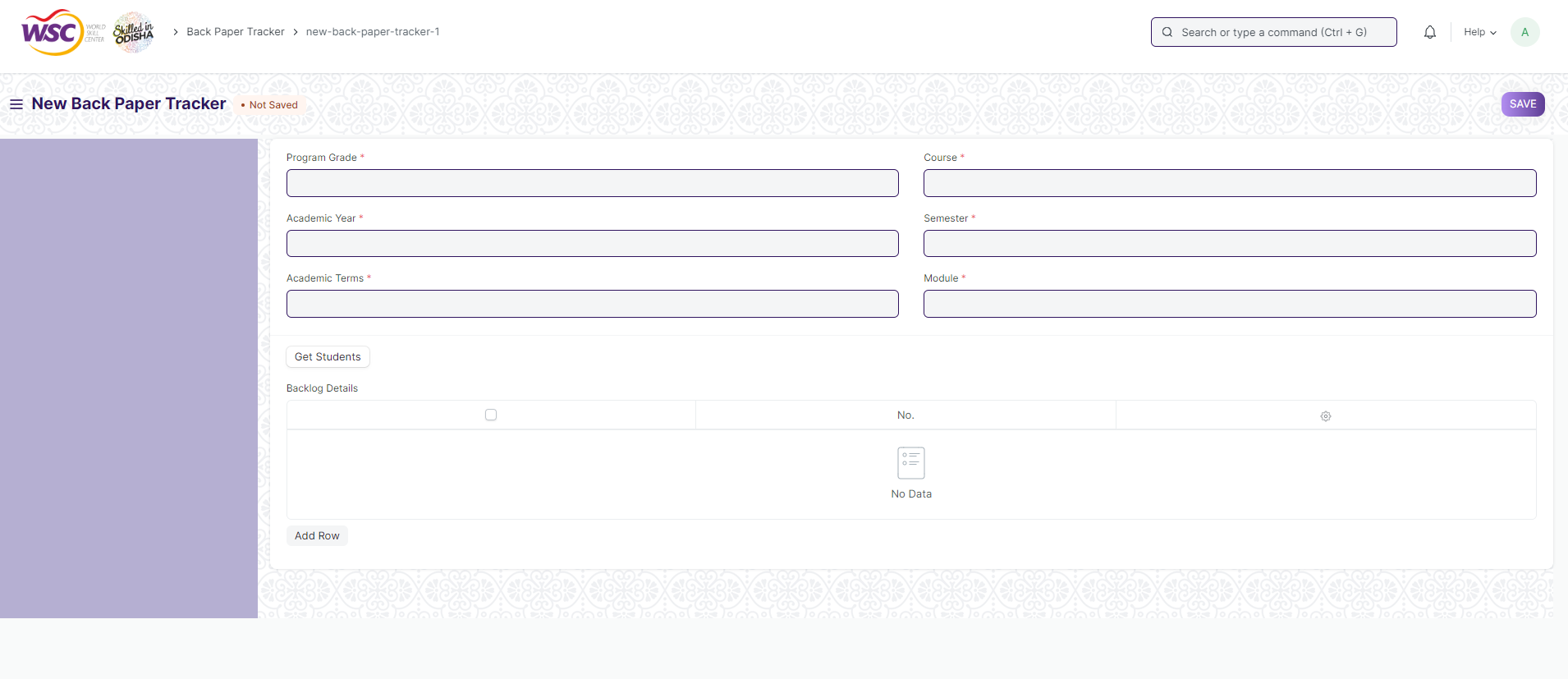


Figure : Student Backpaper Tracking

# Definitions and Acronyms

The following table explains the terms and abbreviations used in the document:

|  |  |
| --- | --- |
| **Abbreviation** | **Description** |
| HLD | High Level Design |
| LLD | Low Level Design |
| API | Application Programming interface |
| NA | Not Applicable |
| ERP | Enterprise Resource Planning |
| HRMS | Human Resource Management System |
| GDPR | General Data Protection Regulation |
| PCI DSS | Payment Card Industry Data Security Standard |

# Deployment Description

* Application Name: Campus Management Application At World Skill Center (WSC)
* Deployment Environment: Production
* Server Information: IP address:117.250.67.19, domain name:erp.worldskillcenter.org, hosting provider: OCAC
* Deployment Date: 13-06-2023
* Database Information: MariaDB 10.6.\*, Installed on the same server as the Application