

**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** | **Academics Module - SOE,SOS,Student Support** |
| **MODULE OWNER** | **Mr. Subhasis Das,Ms. Suguna Srinivasan,Mr. Anil Das** |

**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

**Academics module**

**Department - Admission / SOE / SOS**

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** | **Updated By** |
| 16-August-2023 | 1.0 | Subhodeep Dey | All |  |  |
| 31-August-2023 | 1.1 |  | Project control,Section 3, 7 | Sharmistha Panda | Sharmistha Panda |

# 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. Subhasis Das (subhasis.das@worldskillcenter.org)  Ms. Suguna Srinivasan (suguna.sirinivasan@worldskillcenter.org)  Mr. Anil Das (anil.das@worldskillcenter.org) |

Table Of Contents

[HIGH LEVEL DESIGN (HLD) SIGN-OFF 1](#_Toc9919)

[Authorization Memorandum 1](#_Toc19069)

[Project Control 3](#_Toc19389)

[1 Introduction 7](#_Toc19262)

[1.1 Background 7](#_Toc10474)

[1.2 Scope and Purpose of the document 7](#_Toc2445)

[1.3 Assumptions 8](#_Toc4126)

[1.4 Dependencies 8](#_Toc679)

[1.5 Current IT Environment 8](#_Toc22859)

[1.6 Document Structure 9](#_Toc18390)

[1.7 Required System Architecture 9](#_Toc27967)

[1.8 Interoperability Framework 10](#_Toc4247)

[1.9 Technical Decision Summary 10](#_Toc30665)

[1.10 Architectural Risks & Mitigation 10](#_Toc27317)

[1.11 Reference 11](#_Toc981)

[2 Enterprise Architecture Framework (Business Architecture) 11](#_Toc24698)

[2.1 Business Drivers 11](#_Toc6508)

[2.2 Business Concerns 11](#_Toc27941)

[2.3 Business Goals 12](#_Toc29890)

[2.4 Business Value Chain 12](#_Toc6718)

[2.5 Business Context Diagram 12](#_Toc26302)

[2.6 Business Process Flow 12](#_Toc13456)

[2.6.1 Academics Process Flow (Generic Process Flow) 12](#_Toc12951)

[2.6.2 Course Manager Process Flow 13](#_Toc15809)

[2.6.3 Trainer Process Flow 13](#_Toc6520)

[2.6.4 System Context Diagram 14](#_Toc10213)

[2.7 Conceptual Architecture 14](#_Toc3159)

[3 Architecture Views 15](#_Toc4741)

[3.1 Student Group 15](#_Toc32090)

[3.1.1 Use Case Diagram 15](#_Toc15594)

[3.1.2 Design of Workflow 16](#_Toc21886)

[3.1.3 Validations 16](#_Toc16299)

[3.1.4 Field List: 16](#_Toc30811)

[3.1.5 Processes After Form Submission 18](#_Toc28628)

[3.1.6 Process Flow: 18](#_Toc20289)

[3.1.7 Pre-requisites and Dependency 19](#_Toc8223)

[3.2 Class Scheduling Tool 19](#_Toc24066)

[3.2.1 Use Case Diagram 19](#_Toc22805)

[3.2.2 Design of Workflow 20](#_Toc14104)

[3.2.3 Validations 20](#_Toc10000)

[3.2.4 Field List: 21](#_Toc4628)

[3.2.5 Processes After Form Submission 22](#_Toc21877)

[3.2.6 Process Flow: 22](#_Toc7802)

[3.2.7 Pre-requisites and Dependency 22](#_Toc26787)

[3.3 Class Schedule 22](#_Toc15983)

[3.3.1 Use Case Diagram 22](#_Toc29796)

[3.3.2 Design of Workflow 23](#_Toc28647)

[3.3.3 Validations 23](#_Toc6888)

[3.3.4 Field List: 24](#_Toc8246)

[3.3.5 Processes After Form Submission 25](#_Toc8334)

[3.3.6 Process Flow: 25](#_Toc17708)

[3.3.7 Pre-requisites and Dependency 25](#_Toc8973)

[3.4 Student Attendance Tool 25](#_Toc12334)

[3.4.1 Use Case Diagram 25](#_Toc14001)

[3.4.2 Design of Workflow 26](#_Toc29411)

[3.4.3 Validations 26](#_Toc721)

[3.4.4 Field List: 27](#_Toc10856)

[3.4.5 Processes After Form Submission 28](#_Toc32427)

[3.4.6 Process Flow: 28](#_Toc27961)

[3.4.7 Pre-requisites and Dependency 29](#_Toc14908)

[3.5 Student Attendance 29](#_Toc14625)

[3.5.1 Use Case Diagram 29](#_Toc15064)

[3.5.2 Design of Workflow 29](#_Toc27124)

[3.5.3 Validations 29](#_Toc29069)

[3.5.4 Field List: 30](#_Toc18150)

[3.5.5 Processes After Form Submission 31](#_Toc16794)

[3.5.6 Process Flow: 31](#_Toc25524)

[3.5.7 Pre-requisites and Dependency 31](#_Toc32629)

[3.6 Assignment 31](#_Toc15356)

[3.6.1 Use Case Diagram 32](#_Toc21817)

[3.6.2 Design of Workflow 32](#_Toc25208)

[3.6.3 Validations 32](#_Toc8066)

[3.6.4 Field List: 33](#_Toc11006)

[3.6.5 Processes After Form Submission 34](#_Toc11758)

[3.6.6 Process Flow: 34](#_Toc11621)

[3.6.7 Pre-requisites and Dependency 34](#_Toc27870)

[3.7 Assignment Upload 34](#_Toc12488)

[3.7.1 Use Case Diagram 34](#_Toc27194)

[3.7.2 Design of Workflow 35](#_Toc18883)

[3.7.3 Validations 35](#_Toc30009)

[3.7.4 Field List: 36](#_Toc2964)

[3.7.5 Processes After Form Submission 37](#_Toc22908)

[3.7.6 Process Flow: 37](#_Toc27465)

[3.7.7 Pre-requisites and Dependency 37](#_Toc30724)

[3.8 Leave Application for Student 37](#_Toc11619)

[3.8.1 Use Case Diagram 38](#_Toc21699)

[3.8.2 Design of Workflow 38](#_Toc14746)

[3.8.3 Validations 38](#_Toc9696)

[3.8.4 Field List: 40](#_Toc19871)

[3.8.5 Processes After Form Submission 41](#_Toc6625)

[3.8.6 Process Flow: 41](#_Toc3855)

[3.8.7 Pre-requisites and Dependency 41](#_Toc29782)

[3.9 Reason for Leave 41](#_Toc15870)

[3.9.1 Use Case Diagram 41](#_Toc23745)

[3.9.2 Design of Workflow 42](#_Toc5303)

[3.9.3 Validations 42](#_Toc22984)

[3.9.4 Field List: 42](#_Toc1705)

[3.9.5 Processes After Form Submission 43](#_Toc762)

[3.9.6 Process Flow: 43](#_Toc24476)

[3.9.7 Pre-requisites and Dependency 43](#_Toc1447)

[3.10 Class Advisor and Manager Assignment 43](#_Toc3486)

[3.10.1 Use Case Diagram 43](#_Toc15779)

[3.10.2 Design of Workflow 44](#_Toc29512)

[3.10.3 Validations 44](#_Toc26855)

[3.10.4 Field List: 45](#_Toc19794)

[3.10.5 Processes After Form Submission 46](#_Toc30058)

[3.10.6 Process Flow: 46](#_Toc19848)

[3.10.7 Pre-requisites and Dependency 46](#_Toc8880)

[3.11 Mentor Allocation 46](#_Toc24219)

[3.11.1 Use Case Diagram 47](#_Toc15584)

[3.11.2 Design of Workflow 47](#_Toc24432)

[3.11.3 Validations 47](#_Toc9559)

[3.11.4 Field List: 48](#_Toc22990)

[3.11.5 Processes After Form Submission 49](#_Toc15639)

[3.11.6 Process Flow: 49](#_Toc6047)

[3.11.7 Pre-requisites and Dependency 49](#_Toc10124)

[3.12 Mentor Mentee Communication 49](#_Toc21284)

[3.12.1 Use Case Diagram 49](#_Toc20591)

[3.12.2 Design of Workflow 50](#_Toc25029)

[3.12.3 Validations 50](#_Toc13781)

[3.12.4 Field List: 51](#_Toc30307)

[3.12.5 Processes After Form Submission 51](#_Toc16301)

[3.12.6 Process Flow: 51](#_Toc16226)

[3.12.7 Pre-requisites and Dependency 52](#_Toc30779)

[4 Tools And Technologies 52](#_Toc1466)

[5 Non-Functional Requirements 52](#_Toc21837)

[5.1 Performance 52](#_Toc24016)

[5.2 Reliability 53](#_Toc66)

[5.3 Usability 53](#_Toc16364)

[5.4 Compatibility 53](#_Toc15130)

[5.5 Security 53](#_Toc19843)

[5.6 Compliance 53](#_Toc13087)

[5.7 Documentation 53](#_Toc29105)

[5.8 Security: User access and authorization 53](#_Toc22011)

[6 Database Design 55](#_Toc16016)

[Prototype 57](#_Toc9254)

[7 Prototype 58](#_Toc13913)

[7.1 Student Group 58](#_Toc5136)

[7.2 Class Scheduling Tool 58](#_Toc26220)

[7.3 Class Schedule 59](#_Toc29570)

[7.4 Student Attendance Tool 59](#_Toc11186)

[7.5 Student Attendance 59](#_Toc24829)

[7.6 Assignment 60](#_Toc29430)

[7.7 Assignment Upload 61](#_Toc1320)

[7.8 Leave Application for Student 61](#_Toc7468)

[7.9 Reason for Leave 62](#_Toc13446)

[7.10 Class Advisor and Manager Assignment 62](#_Toc11960)

[7.11 Mentor Allocation 63](#_Toc9230)

[7.12 Mentor Mentee Communication 63](#_Toc27195)

[8 Definitions and Acronyms 63](#_Toc28509)

[9 Deployment Description 64](#_Toc14954)

# 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 **Academic Education module** of ERP Product. This module helps in organizing the entire academic of education set-up, such as Student Database, Trainers Information, Student Attendance, Time table creation, Mentor Mentees communication, Assignments, etc.

The scope of Education module

* Class occurrence monitoring
* Faculty Workload / Lesson Plan 
* 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 
* Students Profile: Personal Information, Contact Details, Academic Details.
* Re-Admission Process after discontinuing in studies
* Issue of Identity Card
* Students Back paper tracking etc.
* Students Profile. :Personal Information, Contact Details, Academic Details 
* Biometric Attendance (Face recognition) 
* Hostel Management (Hostel seat allotment, attendance, fees etc) 
* Scholarships. 
* Rewards and achievements. 
* Students’ Evaluation/remarks of Faculty. 
* Students Exams records of all 6 semesters  Parent Enquiry/Alert/Messaging Management. 
* Messaging System 
* Students Grievance System etc.

Following functionalities are covered in this Design document:

* **Courses and Subjects, Timetable preparation and management**
* **Class Attendance of students**
* **Class occurrence monitoring**

## 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

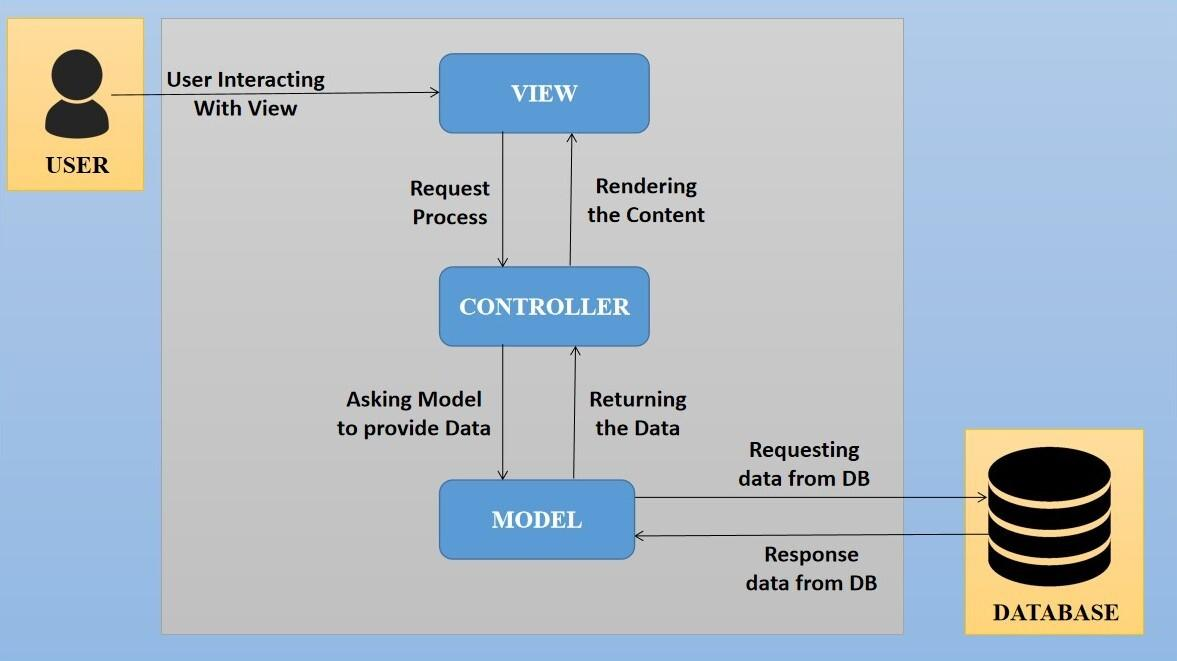
|  |  |
| --- | --- |
| **Hardware Requirements** | **Software Requirements** |
| Processor: Intel CPU with at least 8 cores | Operating System: Ubuntu 22.04 LTS |
| RAM: 16 GB to 32 GB | Web Server: Nginx |
| Storage: A minimum of 160 GB SSD | Database server: MariaDB |
| Network: 1 Gbps | Python: 3.10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

# 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.

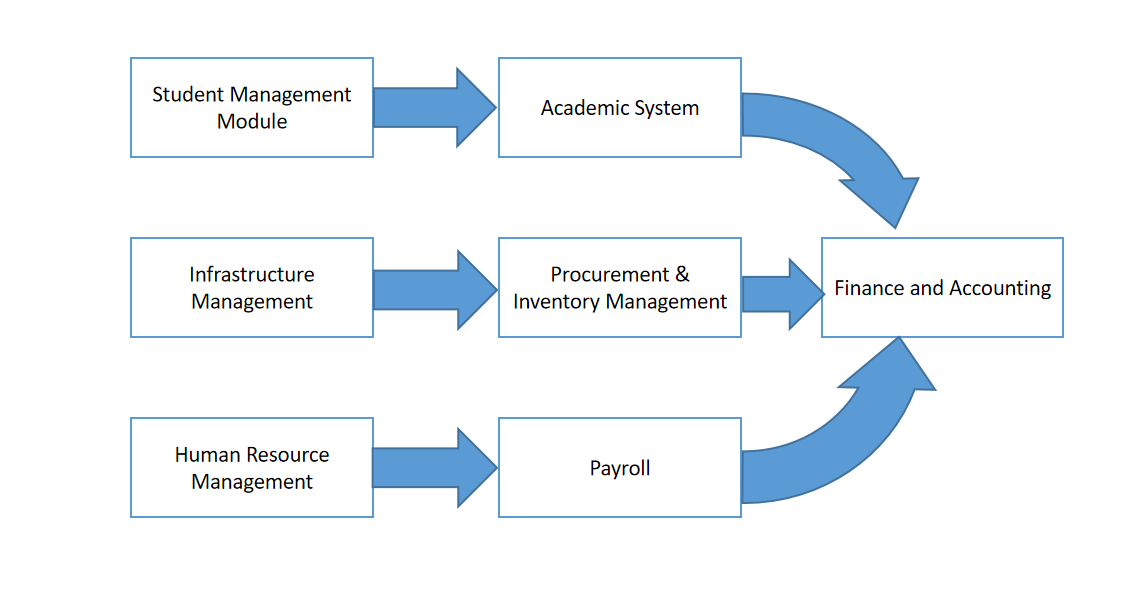
Production/Operations Management: The ERP system facilitates the planning, scheduling, and execution of production processes. It optimizes resource allocation, tracks work progress, manages bills of materials, and monitors production costs to enhance overall operational efficiency and quality.

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.

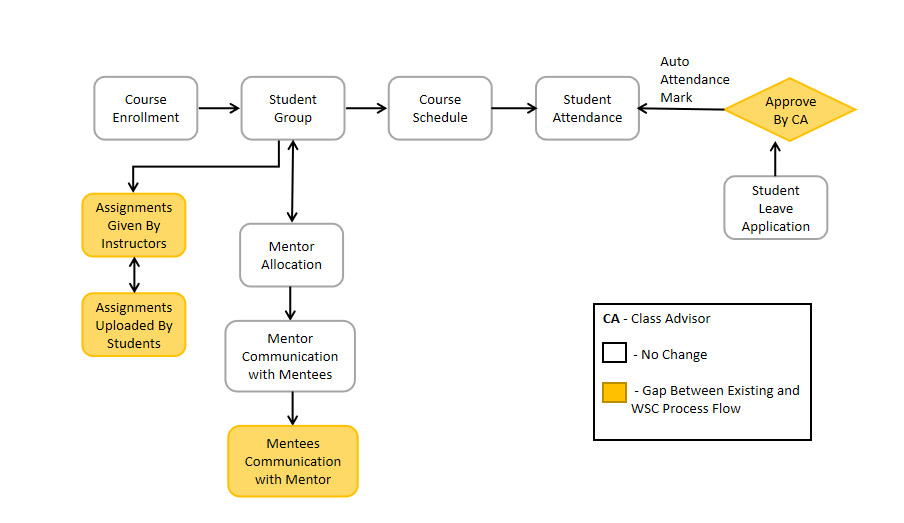
Human Resources Management: ERP systems handle various HR functions, including payroll, employee records, recruitment, performance management, and training. This leads to streamlined HR processes, improved workforce management, and enhanced employee satisfaction.

## Business Context Diagram

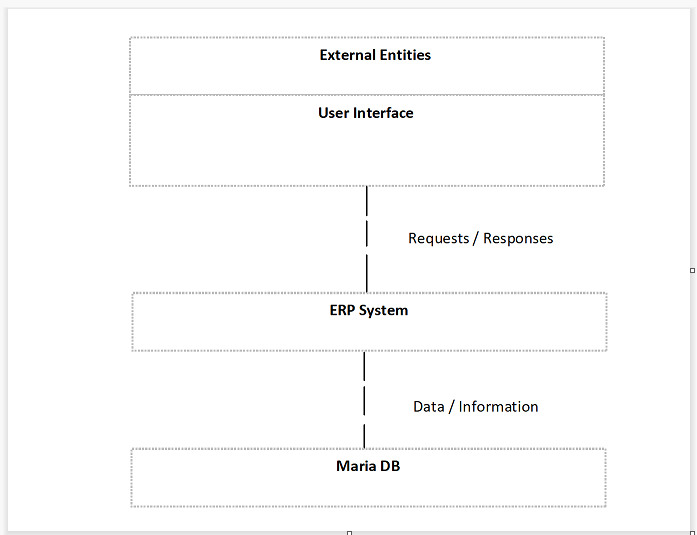


## Business Process Flow

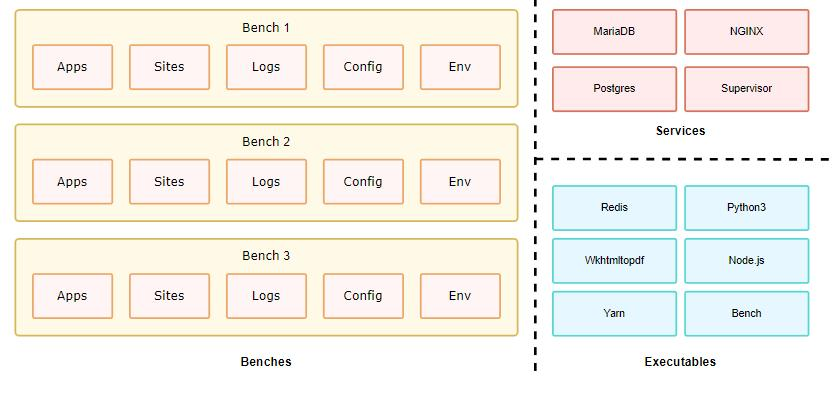
### Academics Process Flow (Generic Process Flow)



### 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

## Student Group

Student group is a collection of students who share common characteristics or attributes.

### Use Case Diagram

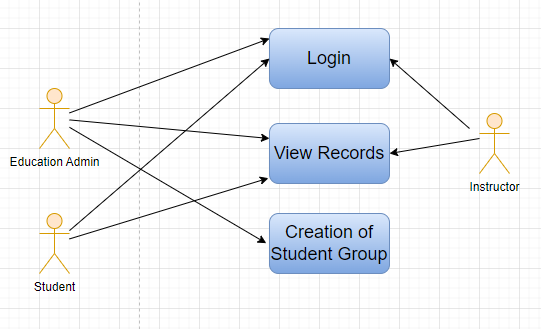


Figure : Student Group Use case diagram

### 

### 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 Student Group screen.

The following validation is done on client side:

* Mandatory checks for fields : Academic Year, Group Based on, Student Group, Student, Instructors, Course
* Linked Fields : The Academic Year field is linked with the Academic Year Screen. The Academic Term field is linked with the Academic Term Screen. The Program field is linked with the Program Screen. The Class field is linked with the Class Screen. The Batch field is linked with the Batch Screen. The Student Category field is linked with the Student Category Screen. The Student Category field is linked with the Student Category Screen. The Course field is linked with the Course Screen. The Student field is linked with the Student Screen. The Instructor field is linked with the Instructor 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:

* def validate(doc, method): This definition is triggered when the user saves the screen. It contains all the definitions that should occur when the user saves the screen. The definitions that are triggered are for validating students, validating course, checking and handling duplicate student groups, getting options from student for custom field

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Academic Year | Link Field | Yes | Value Fetched From the Academic Year list and will show in a drop down. User will select one of them. |  |  |
| 2 | Group Based on | Drop down | Yes | Options : Batch Course Activity  Mentor Communication |  | 1. Combined Course   (D) Exam Declaration |
| 3 | Student Group | Text | Yes | User will give input |  |  |
| 4 | Max Strength | Number |  | User will give input |  |  |
| 5 | Academic Term | Link Field |  | Link Field to Academic Term Screen and filtered applied w.r.t Academic Year |  |  |
| 6 | Program | Link Field |  | Value will be fetched from Program list in a drop down.User will select one value. |  | ( R ) Course |
| 7 | Class | Link Field |  | User need to select the Class Name: Class A, Class B, Class C, Class D |  |  |
| 8 | Batch | Link Field |  | Link Field to Batch Screen |  |  |
| 9 | Student Category | Link Field |  | Value will be fetched from Student Category list in drop down. User will select one value. |  |  |
| 10 | Course | Link Field |  | Value will be fetched from course list , and course field will be visible when a user select group based on courses. |  | ( R ) Module |
| 11 | Disabled | Check box |  | If Checked, the student group record will not be able to visible in any respective transaction |  |  |
| 12 | Get Students | Button |  | 1. User will click the Get Students Button.   2. On click list of students with respect to the course will auto fetched in the students table |  |  |
| 13 | **Students** | Table |  | On clicking Get students button, all the students will be fetched according to the inputted parameters. And the description of the table is given below. |  |  |
| 14 | **Instructors** | Table |  | 1. Instructor list need to be selected according to the inputted parameters . 2. when student group based on course selected then only those instructor list will be on drop down w.r.t. the entered course. 3. Description of the table is given below. |  | ( R )Trainer |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student Group Student** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field | Those students will come who are belongs to the particular course, semester and In group based up on activity and mentor communication, user needs to manually entered the students. | Yes |  |  |
| 2 | Student Name | Text | Student Name will auto fetched when student Id fetched in student Field |  |  |  |
| 3 | Group Roll Number | Number | Roll will auto create |  |  |  |
| 4 | Active | Check box | If unchecked then this Student will not visible in any transaction |  |  |  |
| 5 | Gender | Link Field | Link Field to Gender Screen | Auto Fetch from student screen |  |  |
| 6 | Category | Link Field | Link Field to Student Category Screen | Auto Fetch from student screen |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student Group Instructor** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Instructors | Link Field | Value will fetch from Instructor list.User will select value. | Yes |  | (R) Trainer |
| 2 | Instructor Name | Read only | After selecting the instructor id , this value will be auto filled. |  |  | (R) Trainer Name |
| 3 | Course | Link Field | If course wise group created, then those course will come whose name selected in the above course field | Yes |  | (R) Module |

### Processes After Form Submission

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

### Process Flow:

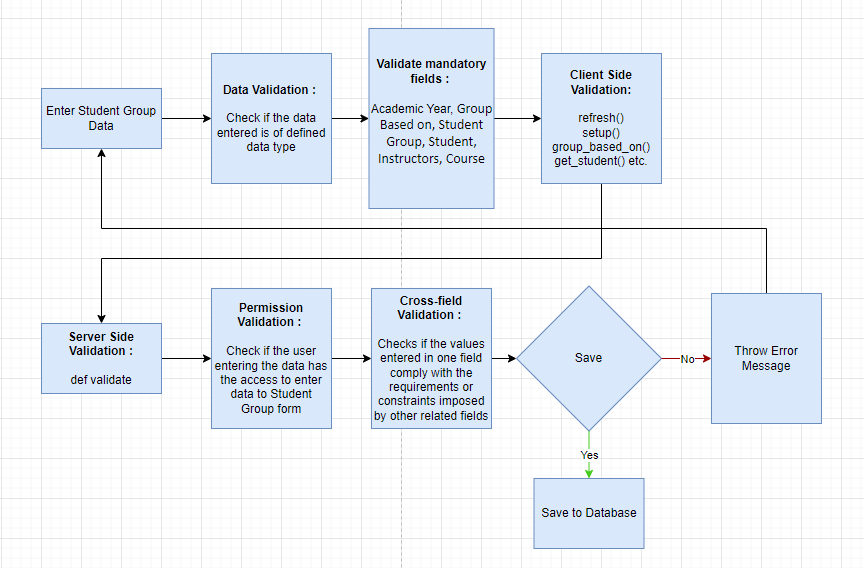


Figure : Student Group Process Flow

### Pre-requisites and Dependency

* Student
* Program Enrollment
* Course Enrollment
* Student Batch Name
* Student Category
* Instructor

## Class Scheduling Tool

The Class Scheduling Tool allows a user to create Class Schedules in bulk for the particular Course.

### Use Case Diagram

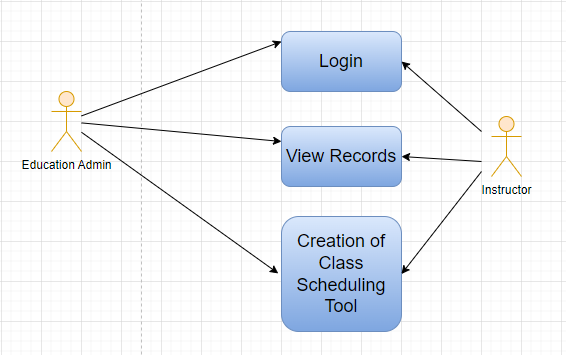


Figure : Class Scheduling Tool Use case diagram

### 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 : Student Group, Course, Class Name, Instructor, Room, From Time, Course Start Date, To Time, Course End Date
* Linked Fields : The Student Group field is linked with the Student Group Screen. The Course field is linked with the Course Screen. The Class Name field is linked with the Class Screen. The Program field is linked with the Program Screen. The Academic Year field is linked with the Academic Term Screen. The Instructor field is linked with the Instructor Screen. The Room field is linked with the Room 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:

* def schedule\_course(self, days): This definition Creates course schedules as per specified parameters
* def validate\_mandatory(self, days): This definition Validates all mandatory fields
* def validate\_date(self): This definition Validates if Course Start Date is greater than Course End Date
* def delete\_course\_schedule(self, rescheduled, reschedule\_errors, days): This definition Delete all course schedule within the Date range and specified filters
* def make\_course\_schedule(self, date): This definition makes a new Course Schedule. param date: Date on which Course Schedule will be created.
* def get\_instructor(doctype, txt, searchfield, start, page\_len, filters): This definition is used to get instructors based to courses
* def get\_course(doctype, txt, searchfield, start, page\_len, filters): This definition is used to get courses(module) based on certain parameters

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
|  | Student Group | Link Field | Yes | Value will be fetched from Student Group List, User will select one of them |  |  |
|  | Course | Link Field | Yes | Value will be fetched from course list.User will select one. |  | ( R ) Module |
|  | Class Name | Link Field | Yes | Auto fetched from the Student Group |  | (N) |
|  | Program | Link Field |  | Value will be fetched from  Program list, user will select. |  | ( R ) Course |
|  | Academic Year | Link Field |  | Value will be fetched from  Academic year list,user  Will select one. |  |  |
|  | Academic Term | Link Field |  | Based on the academic year,  Academic term will show in a drop down. User will select. |  |  |
|  | Instructor | Link Field | Yes | Value will be fetched from  Instructor list,user will select one. |  | ( R ) Trainer |
|  | Instructor Name | Read Only |  | Automatically filled up after  Selecting Instructor. |  | ( R ) Trainer Name |
|  | Room | Link Field | Yes | Value will be fetched from  Room list, user will select one. |  |  |
|  | From Time | Time | Yes | User will select time. |  |  |
|  | Course Start Date | Date | Yes | Start date should be before the End Date |  | ( R ) Module Start Date |
|  | To Time | Time | Yes | User will select time. |  |  |
|  | Course End Date | Date | Yes | Course End date should be greater then Course Start Date |  | ( R ) Module End Date |

### Processes After Form Submission

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

### Process Flow:

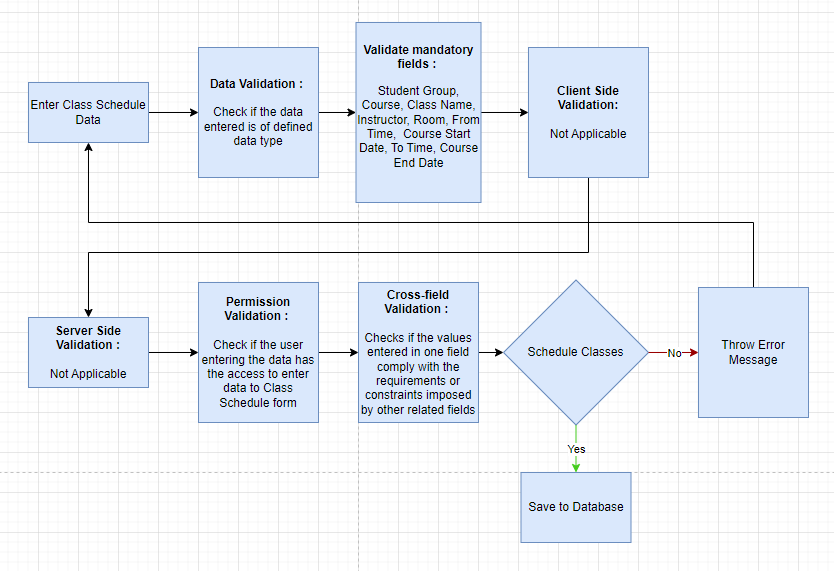


Figure : Class Scheduling Tool Process Flow

### Pre-requisites and Dependency

* Student Group
* Instructor
* Course
* Class
* Classroom

## Class Schedule

A class schedule is a specialized functional area or a division within the Groups and schedule. The Class Schedule can be used for scheduling a particular course by the User.

### Use Case Diagram

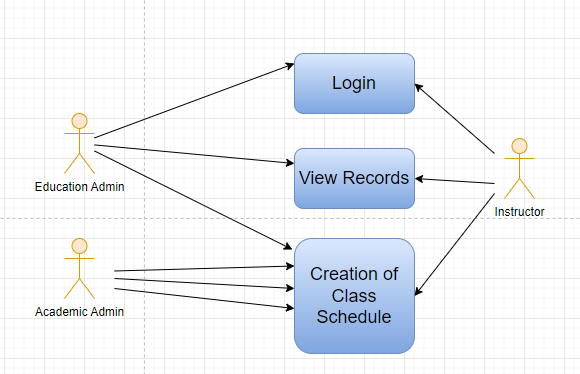


Figure : Class Schedule Use case diagram

### 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 Class Schedule screen.

The following validation is done on client side:

* Mandatory checks for fields : Student Group, Course, Instructor, Room, From Time, To Time
* Linked Fields : The Student Group field is linked with the Student Group Screen. The Course field is linked with the Course Screen. The Program field is linked with the Program Screen. The Instructor field is linked with the Instructor Screen. The Room field is linked with the Room 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:

* def validate(self): This definition is triggered when the screen is saved. It contains all the below definitions
* def set\_title(self): This definition is used to set\_title of the document
* def validate\_course(self): This definition is used to get courses based on student group
* def validate\_date(self): This definition is used to validate if from\_time is greater than to\_time
* def validate\_overlap(self): This definition is used to validate overlap for Student Group, Instructor, Room, Course Schedules, etc.
* def get\_course\_schedule\_events(start, end, filters=None): Returns events for Course Schedule Calendar view rendering.:param start: Start date-time.:param end: End date-time.:param filters: Filters (JSON).

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | | **Field Type** | | **Validation/Action** | | **Mandatory** | | **Remarks** | | **R/N/D** | |
| 1 | Student Group | | Link Field | | Value will be fetched from student group list in a drop down. User will select . | | Yes | |  | |  | |
| 2 | Instructor | | Link Field | | Value will be fetched from the child table (Student Group Instructor) of the selected instructor | | Yes | |  | | ( R ) Trainer | |
| 3 | Instructor Name | | Read only | | Auto fetch instructor name from the above instruction field | |  | |  | | ( R ) Trainer Name | |
| 4 | **Additional Instructor** | | **Table** | |  | |  | | This table is described below | |  | |
| 5 | Program | | Link Field | | Value will be fetched from the selected student group. | |  | |  | | ( R ) Course | |
| 6 | Course | | Link Field | | Value will be fetched from selected student group. | | Yes | |  | | ( R ) Module | |
| 7 | Schedule Date | | Date | | User will enter the schedule date of the course | |  | |  | |  | |
| 8 | Room | | Link Field | | Value will be fetched from the room list in a drop down . user will select. | | Yes | |  | |  | |
| 9 | From Time | | Time | | User will select the time. | | Yes | |  | |  | |
| 10 | To Time | | Time | | User will select the time. | | Yes | |  | |  | |
| **ID** | | **Field Name** | | **Field Type** | | **Validation/Action** | | **Mandatory** | | **Remarks** | | **R/N/D** | |
| 1 | | Instructor | | Link Field | | Value will be fetched from the child table (Student Group Instructor) of the selected instructor | | Yes | |  | | ( R ) Trainer | |
| 2 | | Instructor Name | | Read only | | Auto fetch instructor name from the above instruction field | |  | |  | | ( R ) Trainer Name | |

### Processes After Form Submission

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

### Process Flow:

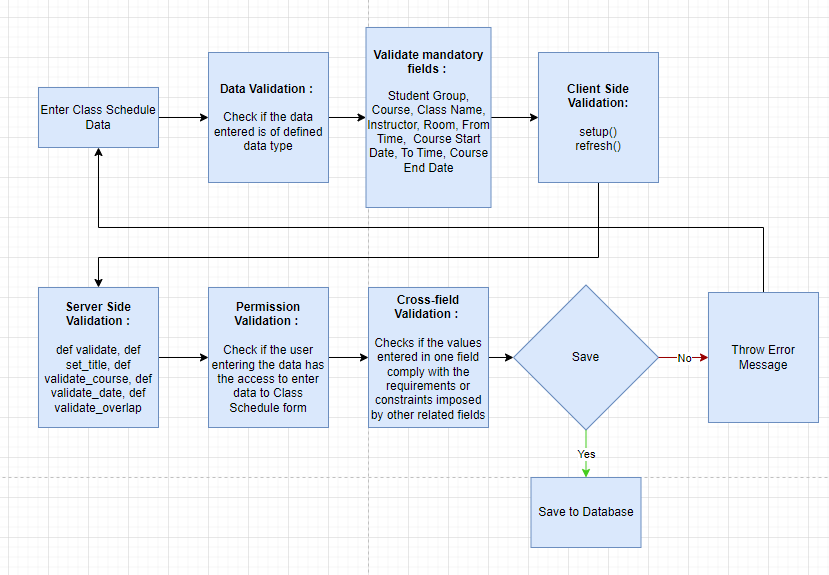


Figure : Class Schedule Process Flow

### Pre-requisites and Dependency

* Student Group
* Instructor
* Course
* Class
* Classroom

## Student Attendance Tool

The Student Attendance Tool allows user to bulk update the Attendance for Students based on Student Group and Course Schedule.

### Use Case Diagram

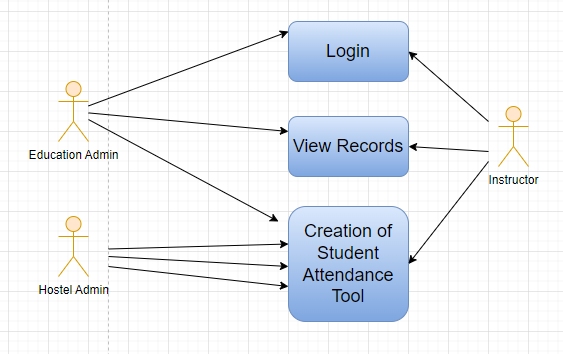


Figure : Student Attendance Tool Use case diagram

### 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 Student Attendance Tool screen.

The following validation is done on client side:

* Mandatory checks for fields : Student Group, Date
* Linked Fields : The Student Group field is linked with the Student Group Screen. The Course Schedule field is linked with the Course Schedule 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.

**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 mark\_attendance(students\_present, students\_absent, students\_on\_leave, course\_schedule=None, student\_group=None,building=None,hostel\_category=None, date=None,attendance\_for=None): This definition Creates Multiple Attendance Records. The following are the parameters involved:
  + students\_present: Students Present JSON.
  + students\_absent: Students Absent JSON.
  + course\_schedule: Course Schedule.
  + student\_group: Student Group.
  + date: Date.
* def make\_attendance\_records(student, student\_name, status, course\_schedule=None, student\_group=None,building=None,hostel\_category=None, date=None,attendance\_for=None): This definition Creates/Update Attendance Record. The following are the parameters involved:
  + student: Student.
  + student\_name: Student Name.
  + course\_schedule: Course Schedule.
  + status: Status (Present/Absent)
* def get\_student\_attendance\_records(based\_on, date=None, student\_group=None, course\_schedule=None): This definition enables fetching of student records based on the user inputs such as based\_on(class schedule or student group), student\_group, class\_schedule

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Based On | Dropdown |  | Options : Student Group Course Schedule |  |  |
| 2 | Group Based On | Dropdown |  | Options :Batch Course Activity |  |  |
| 3 | Student Group | Link Field | Yes | Value will be fetched from  Student group list.User  Will select one. |  |  |
| 4 | Academic Year | Link Field |  | Value will be fetched from  Academic year,user will  Select one. |  |  |
| 5 | Academic Term | Link Field |  | Value will be fetched  Based on selected academic year. |  |  |
| 6 | Course Schedule | Link Field |  | Value will be fetched based on course schedule list. |  | ( R ) Module Schedule |
| 7 | Date | Date | Yes | User will select the date, Date should not be the future date |  |  |
| 8 | Check All | Button |  | On Click all the students attendance will marked automatically |  |  |
| 9 | Uncheck All | Button |  | On Click all the students attendance will unchecked automatically |  |  |
| 10 | Mark Attendance | Button |  | On click a popup will come having the status of number of student absent and present |  |  |
| 11 | Student List | Table |  |  |  |  |
| **Student List Table** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Student Id | Link Field |  | Auto fetch on the basis of Student Group | Need to be read only field |  |
| 2 | Student Name | Text |  | Auto fetch on the basis of Student Group | Need to be read only field |  |
| 3 | Roll Number | Text |  | Auto fetch on the basis of Student Group | Need to be read only field |  |
| 4 | On Leave | Dropdown |  | Options:  Yes  No  Auto fetch when a student leave application is approved or rejected | Need to be read only field |  |

### Processes After Form Submission

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

### Process Flow:

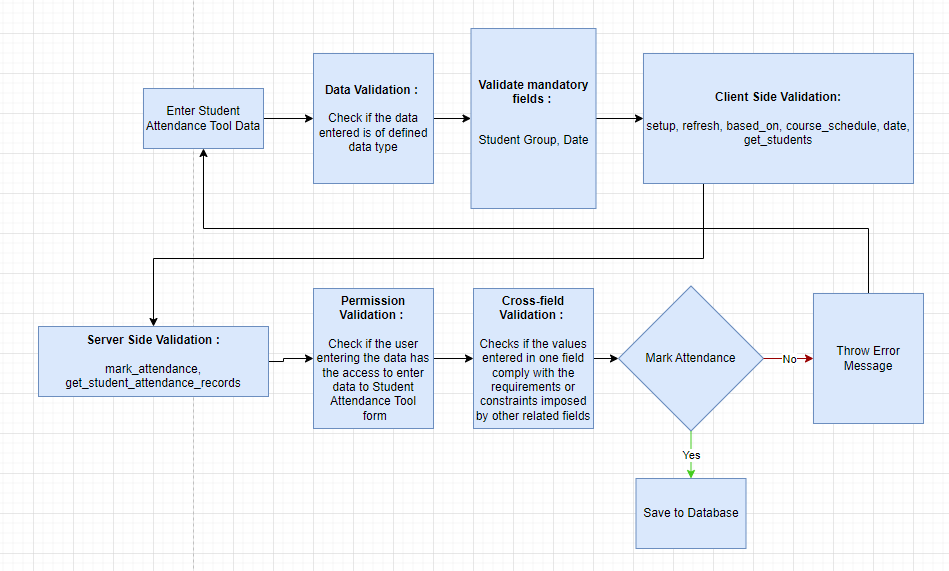


Figure : Student Attendance Tool Process Flow

### Pre-requisites and Dependency

* Student
* Course Schedule
* Student Group

## Student Attendance

The Student Attendance allows you to keep track and mark the attendance of a student for a day. Attendance records can be created against students on a daily basis.

### Use Case Diagram

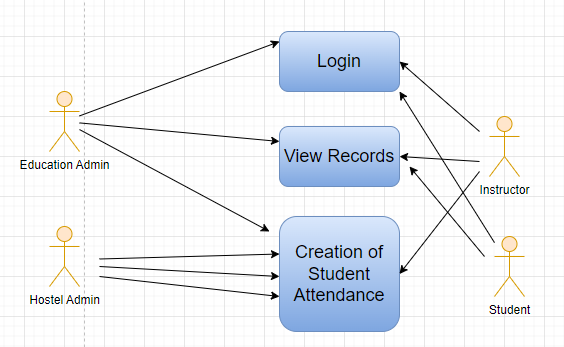


Figure : Student Attendance Use case diagram

### 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 Attendance screen.

The following validation is done on client side:

* Mandatory checks for fields : Student, Date, Status
* Linked Fields : The Student field is linked with the Student Screen. The Course Schedule field is linked with the Course Schedule Screen. The Student Group field is linked with the Student Group Screen. The Module field is linked with the Module 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:

* def validate(self): This definition is triggered when the screen is saved. It contains all the below definitions.
* def set\_date(self): This definition is used to set\_date based on course schedule
* def validate\_date(self): This definition validates such that attendance cannot be mark for future dates and attendance cannot be marked outside of Academic Year.
* def set\_student\_group(self): If attendance is marked based on course schedule student group is fetched based on that schedule if present.
* def validate\_student(self): This definition validates if the student belongs to the student group for which attendance is being marked
* def validate\_duplication(self): This definition is used to check if the Attendance Record is Unique
* def validate\_is\_holiday(self): This definition is used to check if the date for which attendance is marked is a holiday

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Validation/Action** | **Mandatory** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field | Value will be fetched from Student list, user will select. | Yes |  |  |
| 2 | Student Name | Read only | After selecting student id, this field will be  Automatically filled up. |  |  |  |
| 3 | Course Schedule | Link Field | Value will be fetched from course schedule list. |  |  | 1. Module Schedule |
| 4 | Student Group | Link Field | Value will be fetched from Student group list in which the student is present. |  |  |  |
| 5 | Date | Date | User will select the current date | Yes |  |  |
| 6 | Status | Dropdown | Options : Present, Absent, On Leave  User will select. | Yes |  |  |
| 7 | Reason | Long Text | If Status is On Leave, then reason field will be visible |  |  |  |
| 8 | Course | Link Field | Value will be fetched from Course list that the student have enrolled. User will select |  |  | ( R ) Module |

### Processes After Form Submission

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

### Process Flow:

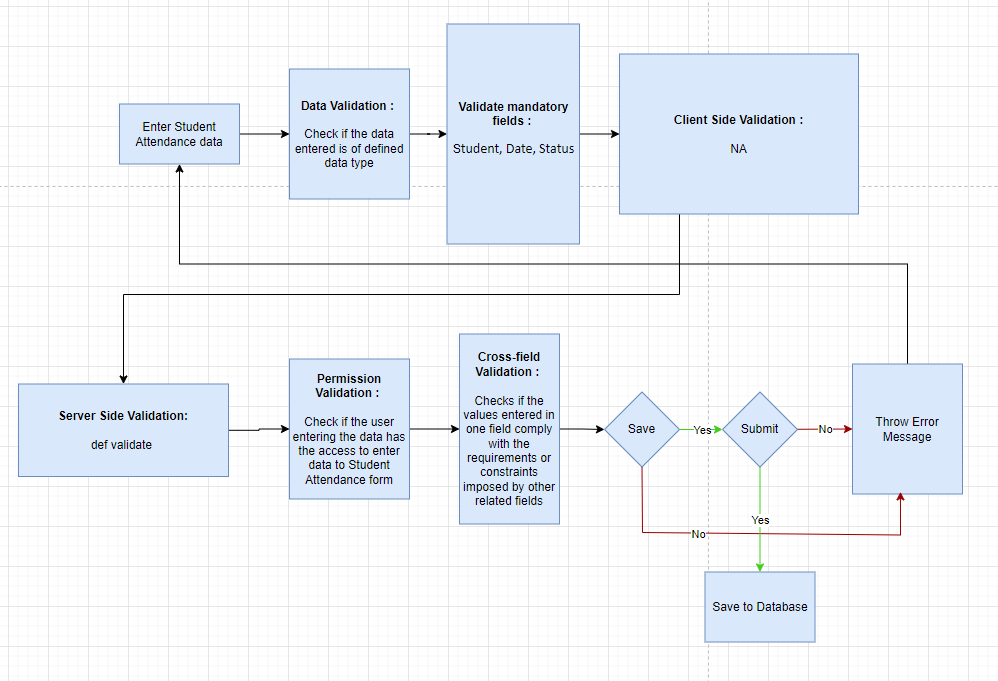


Figure : Student Attendance Process Flow

### Pre-requisites and Dependency

* Student
* Course Schedule
* Student Group

## Assignment

In Assignment Screen Trainers can easily provide assignment to their respective students, so that it gives a strong bonding between the students and the trainers.

### Use Case Diagram

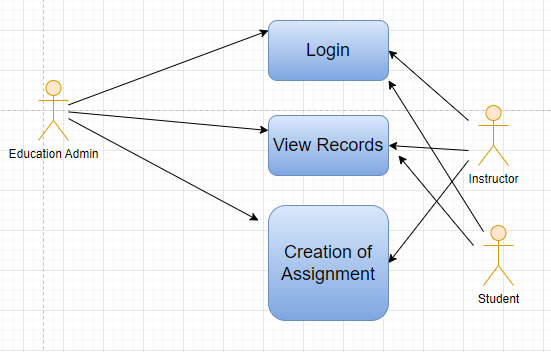


Figure : Assignment Use Case Diagram

### 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 Assignment screen.

The following validation is done on client side:

* Mandatory checks for fields : Instructor Name, Student Group
* Linked Fields : The Instructor name field is linked with the Instructor Screen. The Student Group field is linked with the Student Group 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 Topic field is linked with the Topic 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:

* def get\_details(participant\_group\_id): This whitelisted definition is used to get details of the participants based on Participant Group and other conditions.
* def get\_instructor\_name(participant\_group\_id, instructor\_id): This whitelisted definition is used to get trainer names based on participants
* def get\_criteria\_details(course, assessment\_criteria): This whitelisted definition is used to get total marks, passing marks and weightage based on module and assessment criteria

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Instructor Name | Link Field | Yes | Links with Instructor Screen, filters w.r.t Student Group |  |  |
| 2 | Student Group | Link Field | Yes | Links with Student Group Screen |  |  |
| 3 | Programs | Link Field |  | It will auto fetch when Student Group is entered, filters w.r.t Student Group |  | (R) Course |
| 4 | Semester | Link Field |  | It will auto fetch when Student Group is entered, filters w.r.t Student Group |  |  |
| 5 | Course | Link Field |  | It will auto fetch when Student Group is entered, filters w.r.t Student Group |  | (R) Module |
| 6 | Topic | Link Field |  | Links with Topic Screen and filters topics should come with respect to the above course field |  | (R) Sub Module |
| 7 | Assignment Name | Text |  | User Input |  |  |
| 8 | Assignment Attach Button | Attach |  | 1. On attachment a dialogue box will be popup where user can attach the files, images , takes the picture with the help camera and can link any URL.   If photos or images attached then the size will be maximum 200 kb. |  |  |
| 9 | Description | Long Text |  |  |  |  |
|  | Submit |  |  | On Submit the assignment will be able to visible only to their respective students. |  |  |

### Processes After Form Submission

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

### Process Flow:

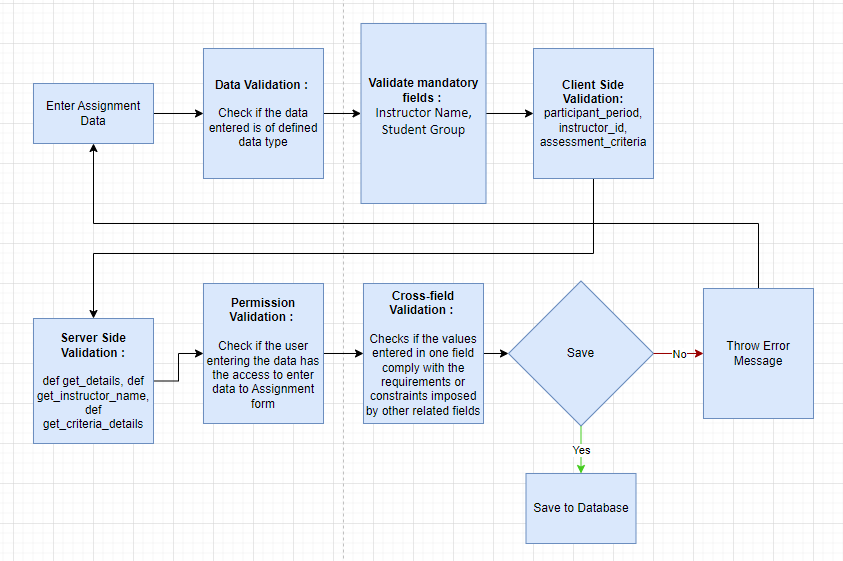


Figure : Assignment Process Flow

### Pre-requisites and Dependency

* Instructor
* Student Group

## Assignment Upload

In Assignment Upload Screen Students can easily upload their assignment and it will securely go to their respective trainers.

### Use Case Diagram

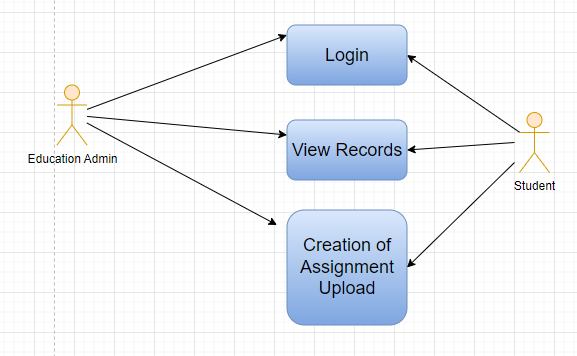


Figure : Assignment Upload Use Case Diagram

### 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 Assignment Upload screen.

The following validation is done on client side:

* Mandatory checks for fields : Instructor Name, Assignment Number, Assignment Name, Description
* Linked Fields : The Instructor name field is linked with the Instructor Screen. The Assignment Number field is linked with the Assignment Screen. The Student ID field is linked with the Student 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.The Course field is linked with the Course Screen.The Topic field is linked with the Topic 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:

* def get\_details(participant\_group\_id): This whitelisted definition is used the get programs, course, topic, academic year, academic term, participant\_id, instructor\_id based on participant group
* def get\_instructor\_name(participant\_group\_id, instructor\_id): This whitelisted definition is used to get trainer details based on participant group
* def get\_participant\_name(participant\_group\_id, participant\_id): This whitelisted definition is used to get based on participant group
* def get\_assignment\_list(instructor\_name, participant\_group\_id, programs, course, topic): This whitelisted definition is used to get the assignments based on certain parameters such as trainer name, participant group, programs, course etc.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Instructor Name | Link Field | Yes | Links with Instructor Screen |  |  |
| 2 | Assginment Number | Link Field | Yes | Links with Assignment Screen |  |  |
| 3 | Student Id | Link Field |  | Link with Student Screen |  |  |
| 4 | Student name | Text Field |  | It will auto fetch when Student Id is entered |  |  |
| 5 | Programs | Link Field |  | It will auto fetch when Student Id is entered |  | ( R ) Course |
| 6 | Semester | Link Field |  | It will auto fetch when Student Id is entered |  |  |
| 7 | Academic Year | Link Field |  | Link with Academic Year Screen |  |  |
| 8 | Academic Term | Link Field |  | Link with Academic Term and Value will be fetched based on selected Academic year. |  |  |
| 9 | Course | Link Field |  | It will auto fetch when Student Id is entered |  | ( R ) Module |
| 10 | Topic | Link Field |  | Links with Topic Screen and filters topics should come with respect to the course |  | (R) Sub Module |
| 11 | Assignment Name | Text | Yes | User Entered |  |  |
| 12 | Assignment Attach Button | Attach |  | 1. On attachment a dialogue box will be popup where user can attach the image, take the picture with the help camera and can link any URL. 2. If photos or images attached then the size will be maximum 200 kb. |  |  |
| 13 | Description | Long Text | Yes |  |  |  |
|  | Submit |  |  | On Submit assignment will be shown only to that particular instructor. |  |  |

### Processes After Form Submission

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

### Process Flow:

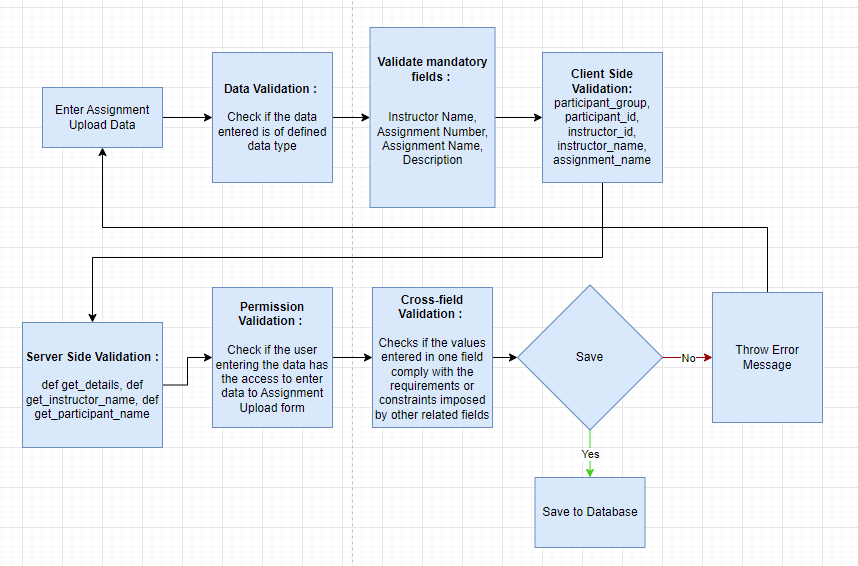


Figure : Assignment Upload Process Flow

### Pre-requisites and Dependency

* Student Group
* Assignment

## Leave Application for Student

This screen allows students to apply for leave and get it approved by the responsible authorities such as Class Advisor and Course Manager. The Student Leave Application allows you to keep a track of leaves for a student.

### Use Case Diagram

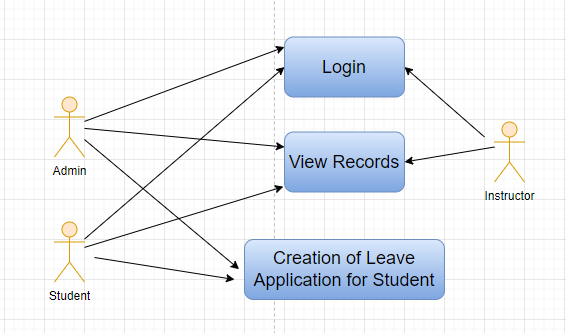
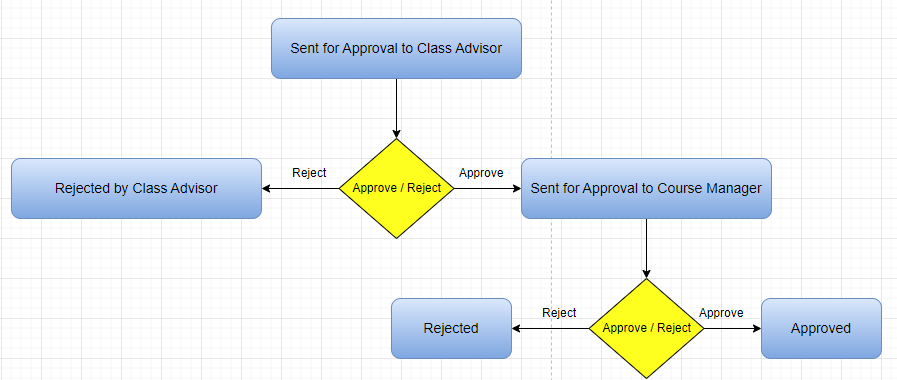


Figure : Leave Application for Student Use Case Diagram

### Design of Workflow

* The workflow is set in the following order :
  + Class Advisor
  + Course Manager
* The following is diagram illustrates the traversal of the workflow states:



### 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 Leave Application for Student screen.

The following validation is done on client side:

* Mandatory checks for fields : Student, From Date, To Date, Leave Criteria, Leave Applicability Hostel, Leave Type, Reason
* Linked Fields : The Student name field is linked with the Student Screen. The Department field is linked with the Department Screen. The Leave Type field is linked with the Leave Type Screen. The Class Schedule field is linked with the Class Schedule 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:

* def on\_submit(self): This definition contains all the logic that executes when the screen is submitted. It contains the logic that students’ whose attendance is already marked but the leave application is approved later. Therefore, when this leave application is approved, it will update the student attendance status as ‘on\_leave’. Another logic includes adding attendance details to ‘Student Leave Process’ in Hostel Module if the student applies for leave in hostel.
* def validate(self):

The function goes through a series of checks to ensure the validity of the leave application and handles various email notifications based on the workflow state of the application.

The code first checks if any class-wise leave applicability has been selected. It iterates through the list of class-wise leave configurations and appends the result of the leave\_applicability\_check function to a check\_list. If all elements in this list are either 0 or None, it implies that no classes have been selected for leave application, and an error is raised using frappe.throw.

Next, the code queries the database to identify any duplicate leave applications for the same student and overlapping date ranges. If duplicate applications are found, the code iterates through them and examines their workflow states. If the application has been rejected, it is ignored. Otherwise, an error message is thrown specifying the duplicate application's details.

Following this, the code handles various scenarios based on the workflow state of the application. If the state is "Sent for Approval to Class Advisor," an email is sent to the class advisor for approval. If the state is "Sent for Approval to Course Manager," an email is sent to the course manager for approval. In cases where the application is rejected or approved, different emails are sent to the student and possibly the deputy director.

In summary, this code ensures that leave applications submitted by students are validated, checked for duplicates, and appropriate email notifications are sent out based on the workflow state of the application.

* def current\_education(student\_no):

This definition defines a whitelisted function called current\_education. This function is meant to be exposed to external API calls, ensuring controlled access to certain data. When invoked with a student\_no parameter, the function fetches information related to the student's current educational details from the "Current Educational Details" document in the database. Specifically, it retrieves data fields including the student's enrolled programs, semesters, academic year, and academic term. These details are gathered based on the provided student\_no as a filter condition for the parent field.

* def get\_classes(from\_date=None,to\_date=None, curr=None,leave\_criteria=None):

This code defines a whitelisted function named get\_classes. This function is intended to be accessible via external API calls and serves to retrieve class schedule information based on certain criteria. When invoked, the function expects parameters such as from\_date, to\_date, curr, and leave\_criteria.

Depending on the provided parameters, the function retrieves class schedule data from the "Course Schedule" document in the database. If the leave\_criteria is set to "Class-Wise Leave," and valid values for from\_date, to\_date, and curr (which likely represents program or semester information) are provided, the function fetches course schedule records that match the specified program and fall within the date range. These records include details like course name, room name, schedule date, and timings. The results are grouped by the schedule name.

Similarly, if leave\_criteria is set to "Full Day" and valid date range and program information is given, the function retrieves course schedule records for the specified program and date range. Additionally, a loop iterates through the fetched course\_schedule list and assigns a check field with a value of 1 to each entry.

**Notification**

* Email Notification will be sent to class advisor when student applies for leave.
* Upon approval/rejection by class advisor notification will sent to course manager.
* Upon approval/rejection by course manager notification will sent to course manager, class advisor and student.
* On final approval notification will also be sent to the deputy director.
* To send notifications a configuration must be done on the Class Advisor and Course Manager Assignment screen.

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Student | Link | Yes | Links with Student Screen |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll No. | Data |  |  |  |  |
| 4 | Registration No | Data |  |  |  |  |
| 5 | Department | Link |  |  |  |  |
| 6 | From Date | Date | Yes | From Date should be less than To Date |  |  |
| 7 | To Date | Date | Yes | To Date should be greater than From Date |  |  |
| 8 | Current Education Details | Table |  |  |  |  |
| 9 | Leave Criteria | Select | Yes |  |  |  |
| 10 | Leave Applicability Hostel | Select | Yes |  |  |  |
| 11 | Class Wise Leave | Table |  | Links to Class Wise Leave table |  |  |
| 12 | Leave Type | Link | Yes |  |  |  |
| 13 | Reason | Text Editor | Yes |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Class Wise Leave** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Class Schedule ID | Link |  | Links to Class Schedule |  |  |
| 2 | Module Name | Data |  |  |  |  |
| 3 | Class Room | Data |  |  |  |  |
| 4 | Schedule Date | Data |  |  |  |  |
| 5 | From Time | Data |  |  |  |  |
| 6 | To Time | Data |  |  |  |  |
| 7 | Leave Applicability Check | Check |  |  |  |  |

### Processes After Form Submission

* Leave Application workflow will be active
* Notification for Leave Application will be sent to their respective stakeholders

### Process Flow:

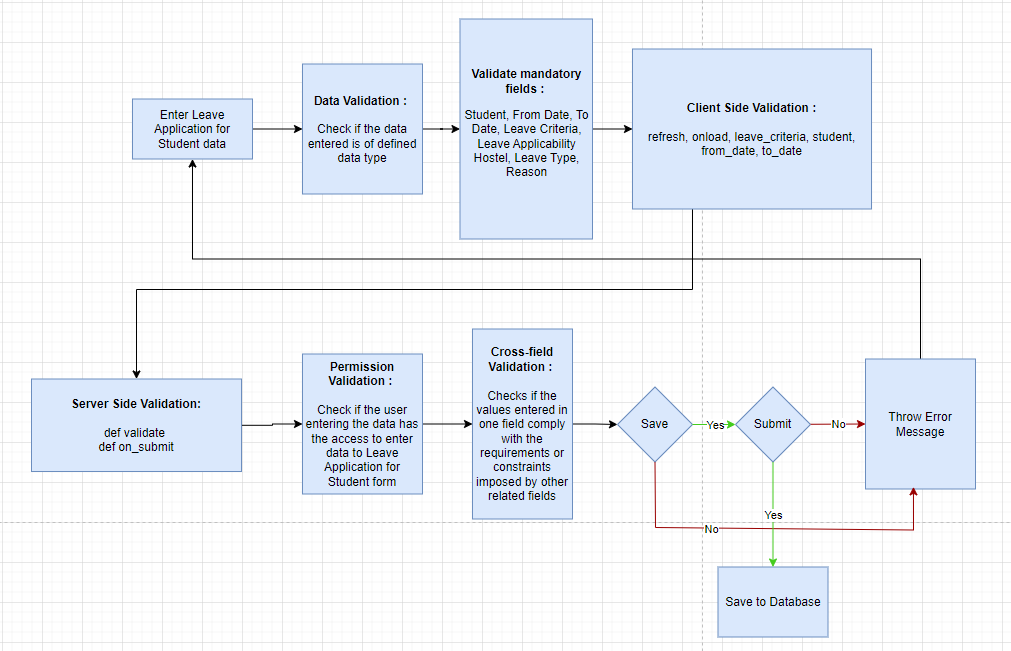


Figure : Leave Application for Student Process Flow

### Pre-requisites and Dependency

* Student
* Leave Type(Reason for Leave)
* Class Schedule
* Department

## Reason for Leave

This screen allows user to define the type of leave a student can apply.

### Use Case Diagram

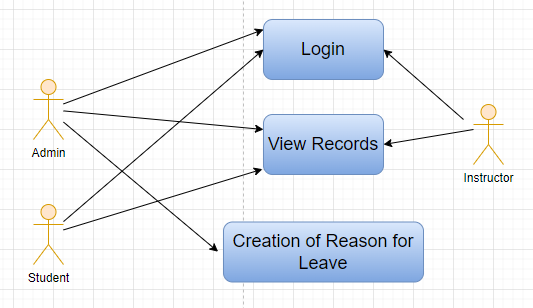


Figure : Reason for Leave Use Case Diagram

### 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 Reason for Leave 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 is no server side validations for Reason for Leave.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Reason | Data |  |  |  |  |

### Processes After Form Submission

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

### Process Flow:

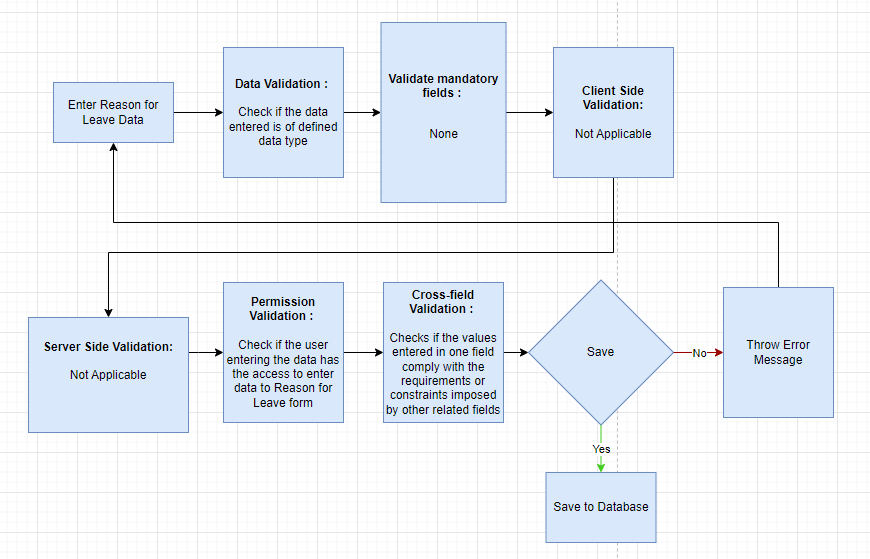


Figure : Reason for Leave Process Flow

### Pre-requisites and Dependency

* None

## Class Advisor and Manager Assignment

This screen allows user to define the class advisor, course manager and deputy director to whom notifications will be sent for a particular set of students when the student applies for leave.

### Use Case Diagram

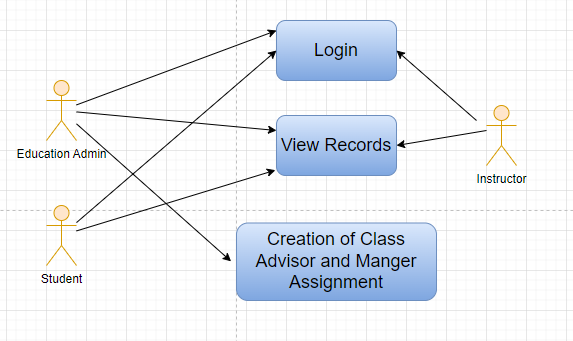


Figure : Class Advisor and Manager Assignment Use Case Diagram

### 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 Class Advisor and Manager Assignment screen.

The following validation is done on client side:

* Mandatory checks for fields : Academic Year, Program Grades, Academic Term, Programs, Semester
* Linked Fields : The Academic Year field is linked with the Academic Year Screen. The Program Grades field is linked with the Program Grades Screen. The Academic Term field is linked with the Academic Term Screen. The Programs field is linked with the Programs Screen. The Semester field is linked with the Semester Screen. The Course Manager Name field is linked with the Trainer Screen. The Course Advisor Name field is linked with the Trainer Screen. The Employee 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 server side validations for Class Advisor and Manager Assignment:

* def get\_students(academic\_term=None, programs=None,class\_data=None,semester=None): This definition enables fetching of students based on academic\_term, program, semester etc. It uses the get\_program\_enrollment definition to determine the students current education details.
* def get\_program\_enrollment(academic\_term,programs=None,class\_data=None): This definition runs a sql query to return all the students whose are enrolled in a particular academic\_term, program etc.
* def get\_cm\_email(employee=None): This definition fetches the employees’ user id to get the Course Manager email
* def get\_ca\_email(employee\_1=None): This definition fetches the employees’ user id to get the Class Advisor email

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Academic Year | Link | Yes | Links to Academic Year |  |  |
| 2 | Program Grades | Link | Yes | Links to Program Grade |  |  |
| 3 | Class | Link |  |  |  |  |
| 4 | Academic Term | Link | Yes | Links to Academic Term |  |  |
| 5 | Programs | Link | Yes | Links to Programs |  |  |
| 6 | Semester | Link | Yes | Links to Semester |  |  |
| 7 | Get Students | Button |  |  |  |  |
| 8 | Students Details | Table |  | Links to Course Manager Assignment Student Details |  |  |
| 9 | Course Manager Name | Link |  | Links to Trainer |  |  |
| 10 | Employee | Data |  |  |  |  |
| 11 | Employee Number | Data |  |  |  |  |
| 12 | Department | Data |  |  |  |  |
| 13 | CM Email | Data |  |  |  |  |
| 14 | Course Advisor Name | Link |  | Links to Trainer |  |  |
| 15 | Employee | Data |  |  |  |  |
| 16 | Employee Number | Data |  |  |  |  |
| 17 | Department | Data |  |  |  |  |
| 18 | CA Email | Data |  |  |  |  |
| 19 | Employee | Link |  | Links to Trainer |  |  |
| 20 | DD Name | Data |  |  |  |  |
| 21 | Employee Number | Data |  |  |  |  |
| 22 | DD Department | Data |  |  |  |  |
| 23 | DD Email | Data |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Course Manager Assignment Student Details** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Student | Link |  | Links to Student |  |  |
| 2 | Student Name | Data |  |  |  |  |
| 3 | Roll Number | Data |  |  |  |  |
| 4 | Permanent Registration Number | Data |  |  |  |  |

### Processes After Form Submission

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

### Process Flow:

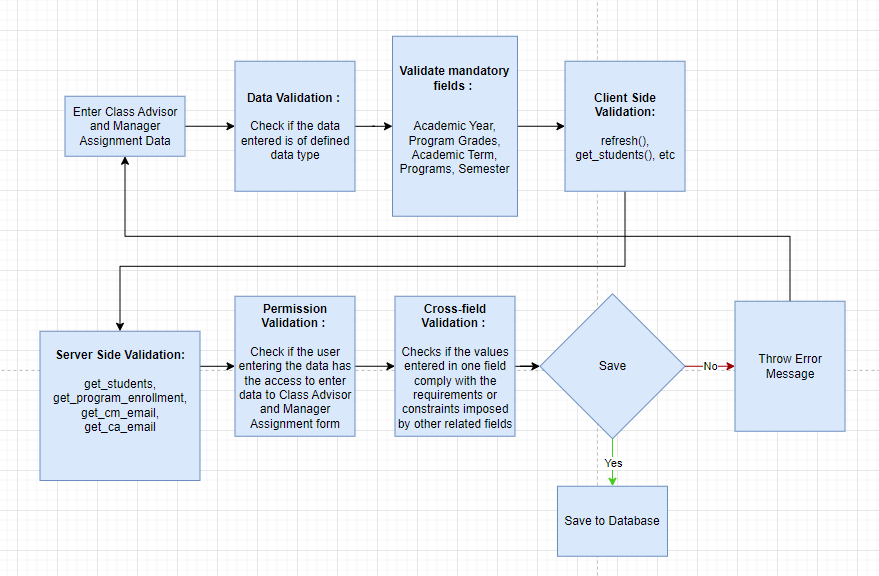


Figure : Class Advisor and Manager Assignment Process Flow

### Pre-requisites and Dependency

* Student
* Class Advisor
* Course Manager
* Deputy Director

## Mentor Allocation

Mentor Allocation is a process in which Mentors (Instructors) are allocated to different student groups or students on various basis. It can either be a hostel mentor allocation or program wise allocation.

### Use Case Diagram

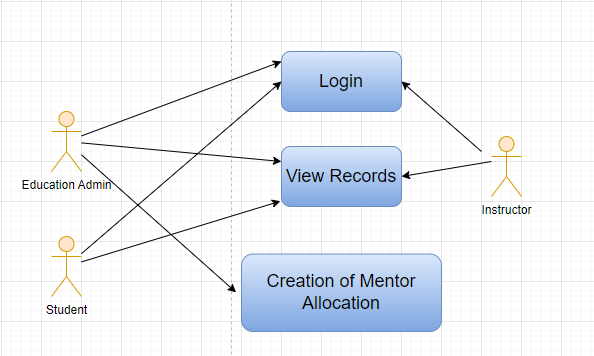


Figure : Mentor Allocation Use Case Diagram

### 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 Mentor Allocation screen.

The following validation is done on client side:

* Mandatory checks for fields : Mentor, Programs
* Linked Fields : The Mentor field is linked with the Employee Screen. The Allocation From field is linked with the Academic Year Screen. The Allocation From field is linked with the Academic Year Screen. The Academic Year field is linked with the Academic Year Screen. The Programs field is linked with the Programs Screen. The Semester field is linked with the Program 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.

There following are the server side validations for Mentor Allocation:

* def validate(doc, method): This definition contains all the logic for the mentor allocation screen when it is saved. It contain two more definitions namely, validate\_semester and validate\_students which is used to validate if the semester belongs to the program and students already have a mentor allocated or not respectively.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Mentor | Link Field | Yes | Value will be fetched from Employee List, User will select one |  |  |
| 2 | Mentor Name | Text |  | Value will be Filled up after selecting Mentor |  |  |
| 3 | Allocation From | Link Field |  | Value will be fetched from Academic Year  User will select one |  |  |
| 4 | Allocation To | Link Field |  | Value will be fetched from Academic Year  User will select one |  |  |
| 5 | Academic Year | Link Field |  | Value will be fetched from Academic Year  User will select one |  |  |
| 6 | Programs | Link Field | Yes | Value will be fetched from program list  User will select one |  | (R) Courses |
| 7 | Semester | Link Field |  | Value will be fetched from the program list, where a program is selected  User will select one |  |  |
| 8 | **Mentee List** | Child Table |  | User will add data in the child table |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mentee List** | | | | | | |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Student | Link Field |  | User need to select the students manually |  |  |
| 2 | Student Name | Text |  | After selecting student Id, Student Name will be automatically filled up |  |  |
| 3 | User | Text |  |  |  |  |
| 4 | Roll No | Text |  | After selecting student Id, roll number will be automatically filled up |  |  |

### Processes After Form Submission

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

### Process Flow:

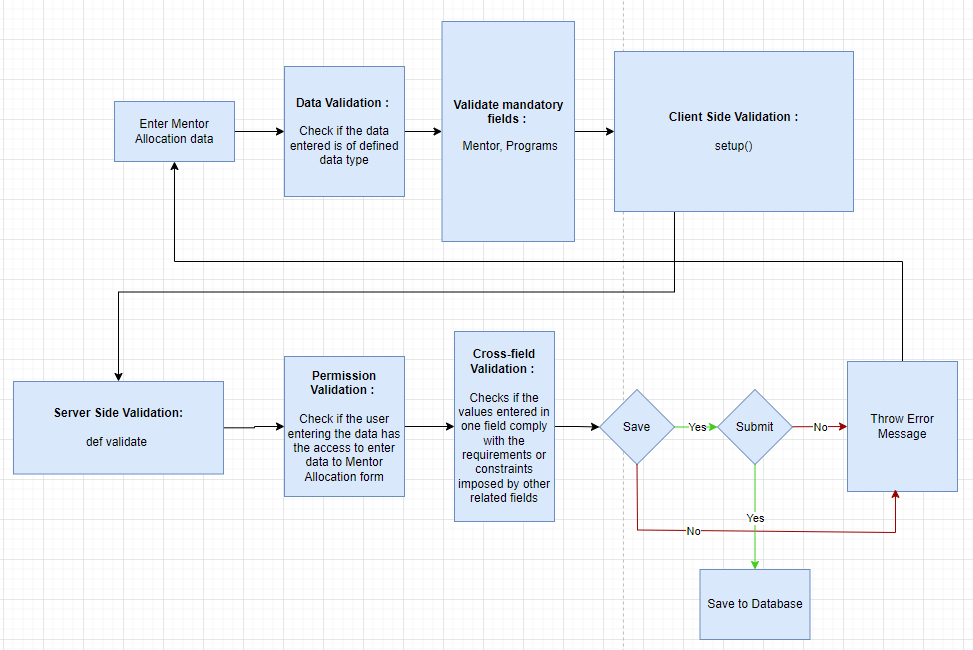


Figure : Mentor Allocation Process Flow

### Pre-requisites and Dependency

* Student
* Mentor
* Programs

## Mentor Mentee Communication

In Mentor Communication, mentors or trainers can communicate with their respective students to share different information and other activities.

### Use Case Diagram

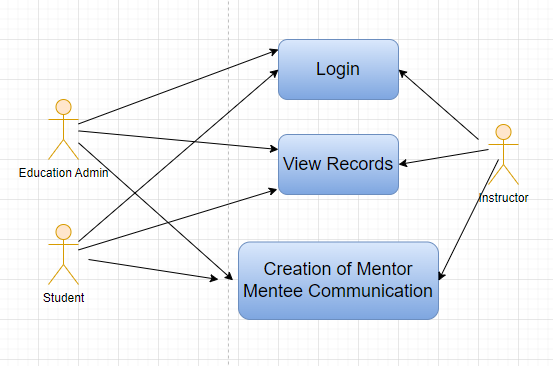


Figure : Mentor Mentee Communication Use Case Diagram

### 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 Mentor Mentee Communication screen.

The following validation is done on client side:

* Mandatory checks for fields : None
* Linked Fields : The Mentor Allocation field is linked with the Mentor Allocation Screen. The Programs field is linked with the Programs Screen. The Student Disciplinary Complain field is linked with the Student Disciplinary Complain 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 Mentor Mentee Communication:

* def validate(doc, method): This definition is triggered when the form is saved. This definition triggers another definition called validate\_students(doc) which is used to check whether the student has an allocated mentor.

**Notification**

Not Applicable

### Field List:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Field Name** | **Field Type** | **Mandatory** | **Validation/Action** | **Remarks** | **R/N/D** |
| 1 | Date | Date |  |  |  |  |
| 2 | Mentor Allocation | Link |  | Links to Mentor Allocation |  |  |
| 3 | Mentor Name | Data |  |  |  |  |
| 4 | Student | Link |  |  |  |  |
| 5 | Student Name | Data |  |  |  |  |
| 6 | Programs | Link |  | Links to Programs |  |  |
| 7 | Description | Text Editor |  |  |  |  |
| 8 | Comment Message | Text |  |  |  |  |
| 9 | Student Disciplinary Complain | Link |  | Links to Student Disciplinary Complain |  |  |

### Processes After Form Submission

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

### Process Flow:

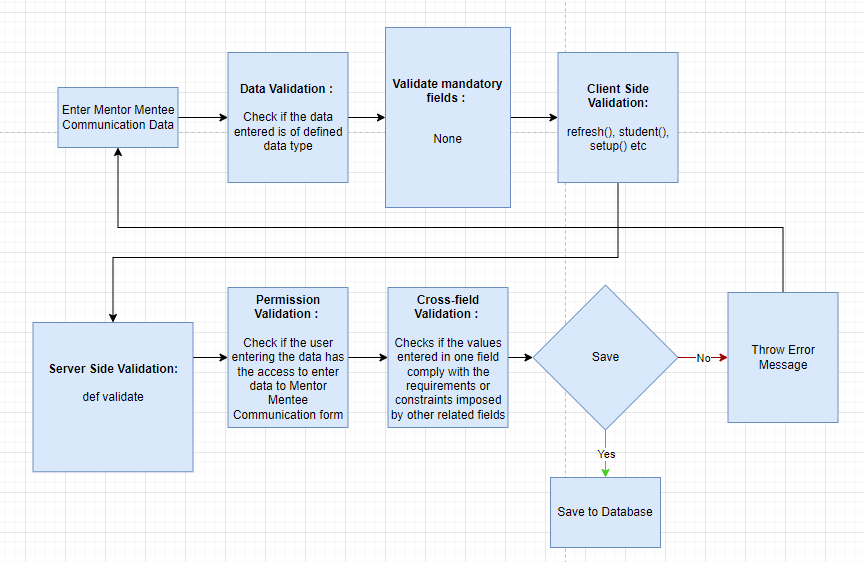


Figure : Mentor Mentee Communication Process Flow

### Pre-requisites and Dependency

* Mentor Allocation
* Programs
* Student Disciplinary Complain

# 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:

**Student Group**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Student | Yes | No | No | No | NA | NA | NA |
| 3 | Education Admin | Yes | Yes | Yes | Yes | NA | NA | NA |
| 4 | Instructor | Yes | No | No | No | NA | NA | NA |

**Class Scheduling Tool**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Instructor | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |

**Class Schedule**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Academics Head | No | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Education Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 3 | Instructor | No | Yes | Yes | Yes | No | NA | NA | NA |
| 4 | Student | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 5 | Academic Management Group | Yes | Yes | No | No | No | NA | NA | NA |

**Student Attendance Tool**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Instructor | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 3 | Hostel Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |

**Student Attendance**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Instructor | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 3 | Student | No | Yes | No | No | No | No | No | No |
| 4 | Hostel Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Assignment**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Instructor | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 3 | Student | No | Yes | No | No | No | NA | NA | NA |

**Assignment Upload**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Student | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |

**Leave Application for Student**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Instructor | No | Yes | Yes | No | Yes | Yes | No | No |
| 3 | Student | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

**Reason for Leave**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | No | No | No |
| 2 | Instructor | Yes | Yes | No | No | No | No | No | No |
| 3 | Student | Yes | Yes | No | No | No | No | No | No |

**Class Advisor and Manager Assignment**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | No | No | No |
| 2 | Instructor | No | No | No | No | No | No | No | No |
| 3 | Student | No | No | No | No | No | No | No | No |

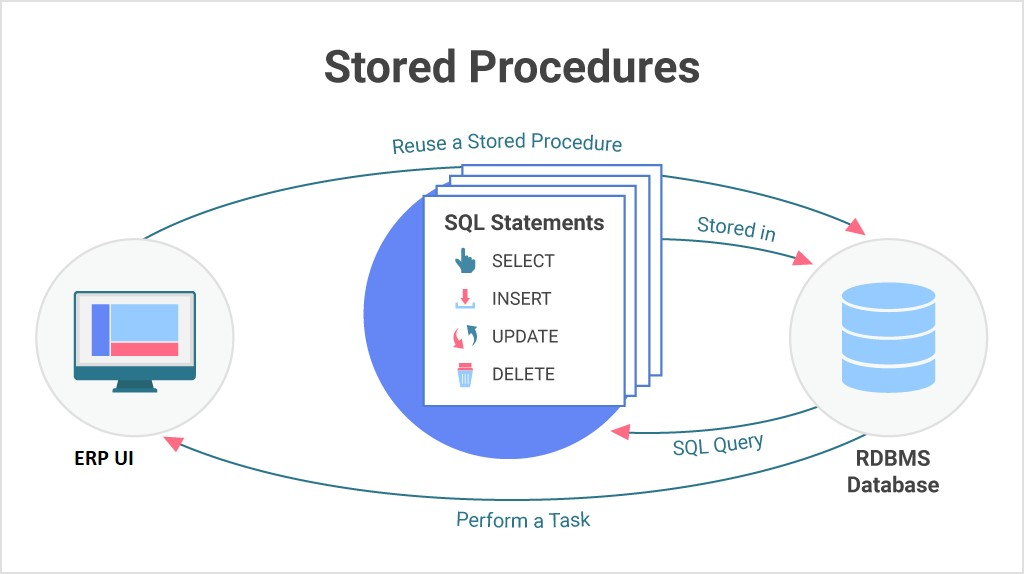
**Mentor Allocation**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 2 | Instructor | Yes | Yes | No | No | No | No | No | No |
| 3 | Student | Yes | Yes | No | No | No | No | No | No |

**Mentor Mentee Communication**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Role** | **Select** | **Read** | **Write** | **Create** | **Delete** | **Submit** | **Cancel** | **Amend** |
| 1 | Admin | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |
| 2 | Instructor | Yes | Yes | Yes | Yes | Yes | NA | NA | NA |

# Database Design



**Prototype**

# Prototype

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

## Student Group

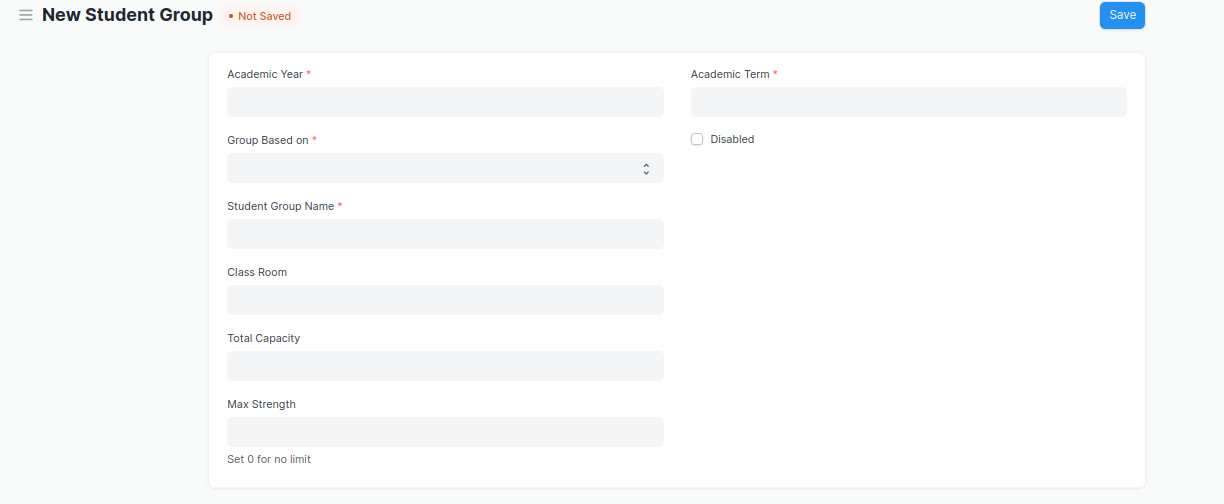


Figure : Student Group

## Class Scheduling Tool

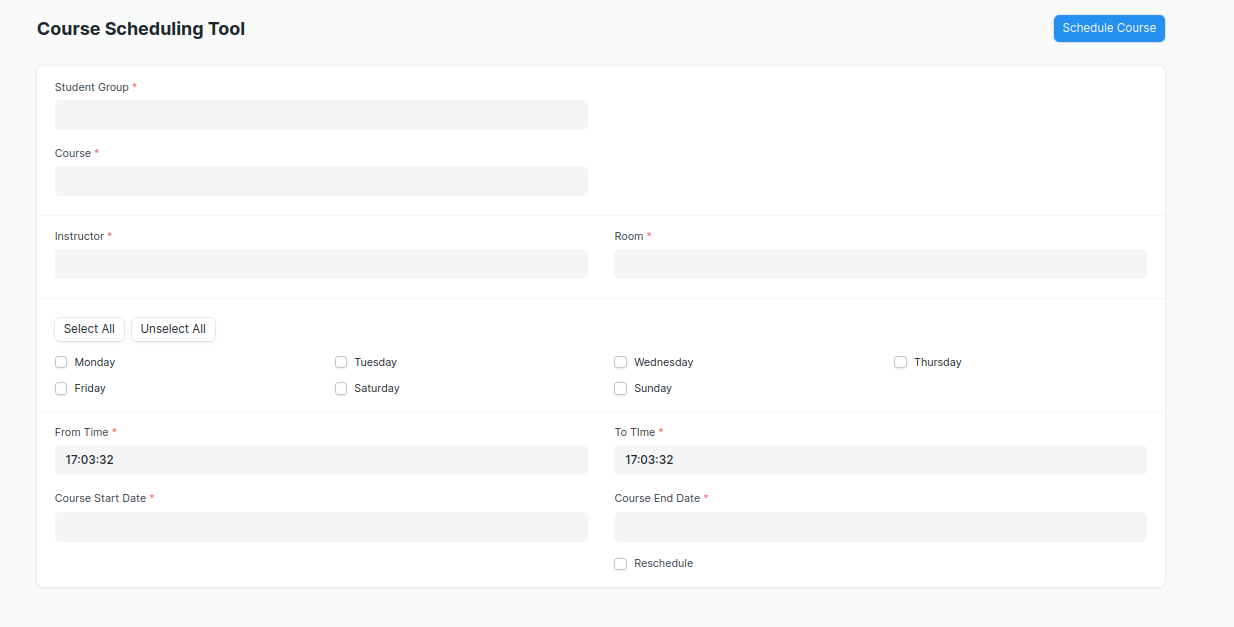


Figure : Class Scheduling Tool

## Class Schedule

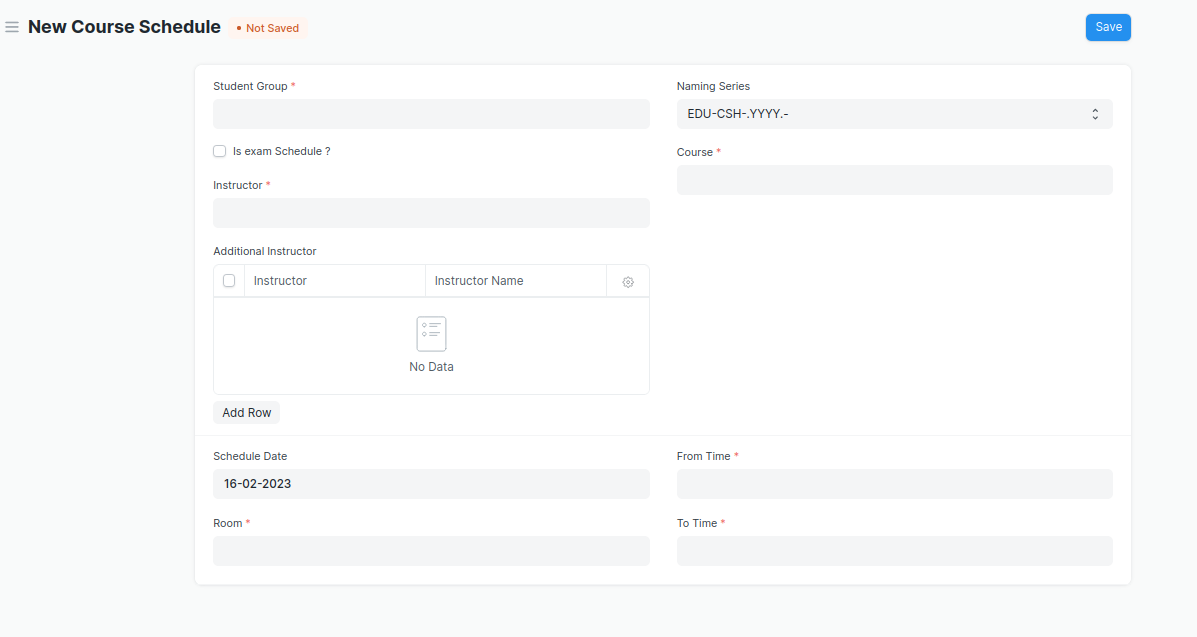


Figure : Class Schedule

## Student Attendance Tool

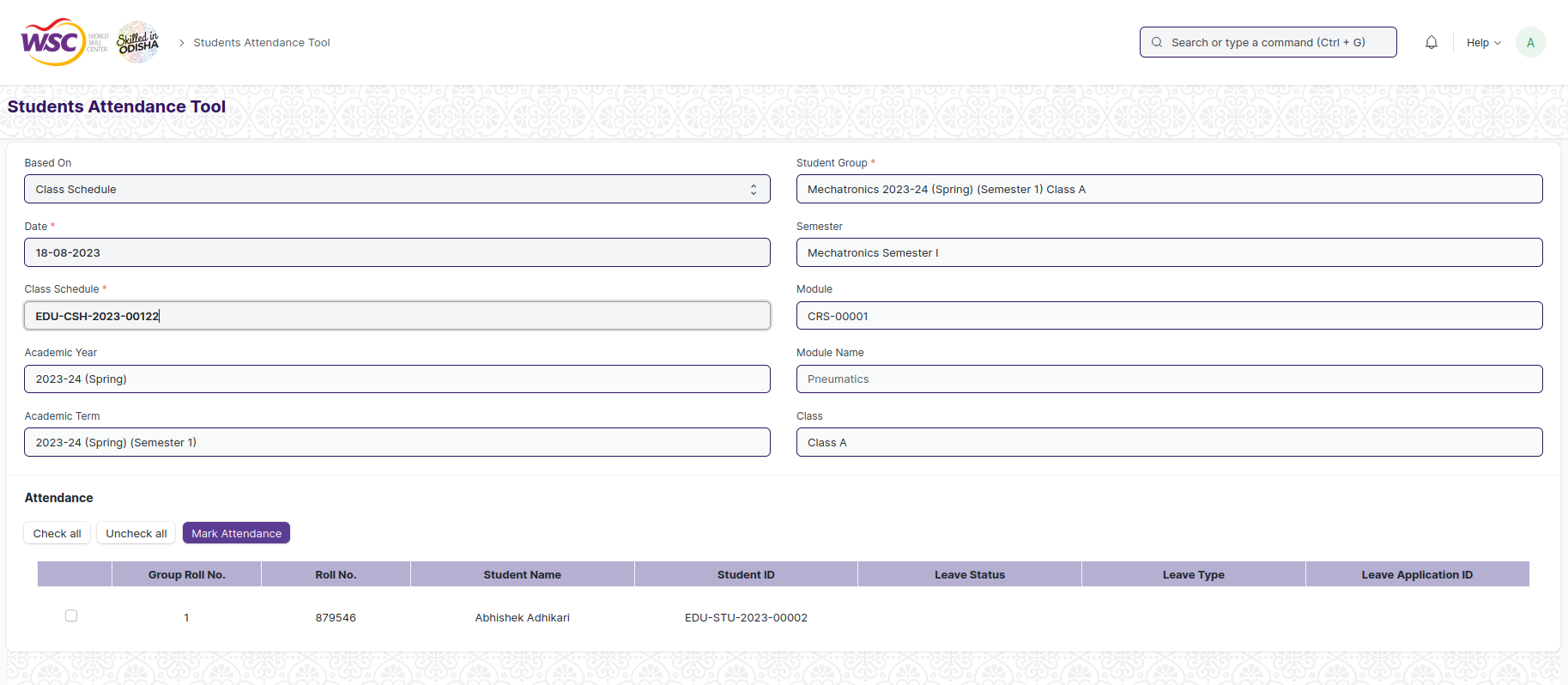


Figure : Student Attendance Tool

## Student Attendance

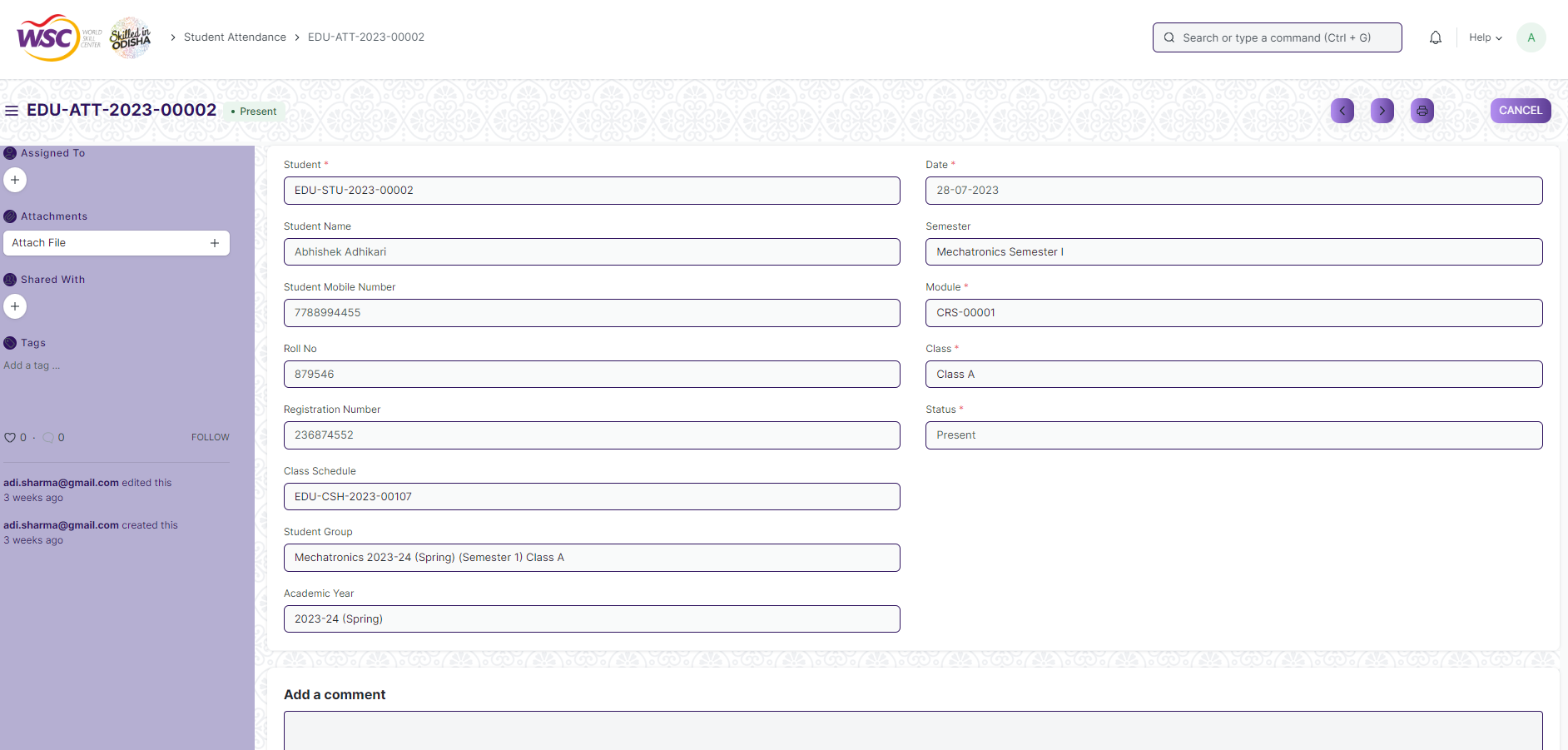


Figure : Student Attendance

## Assignment

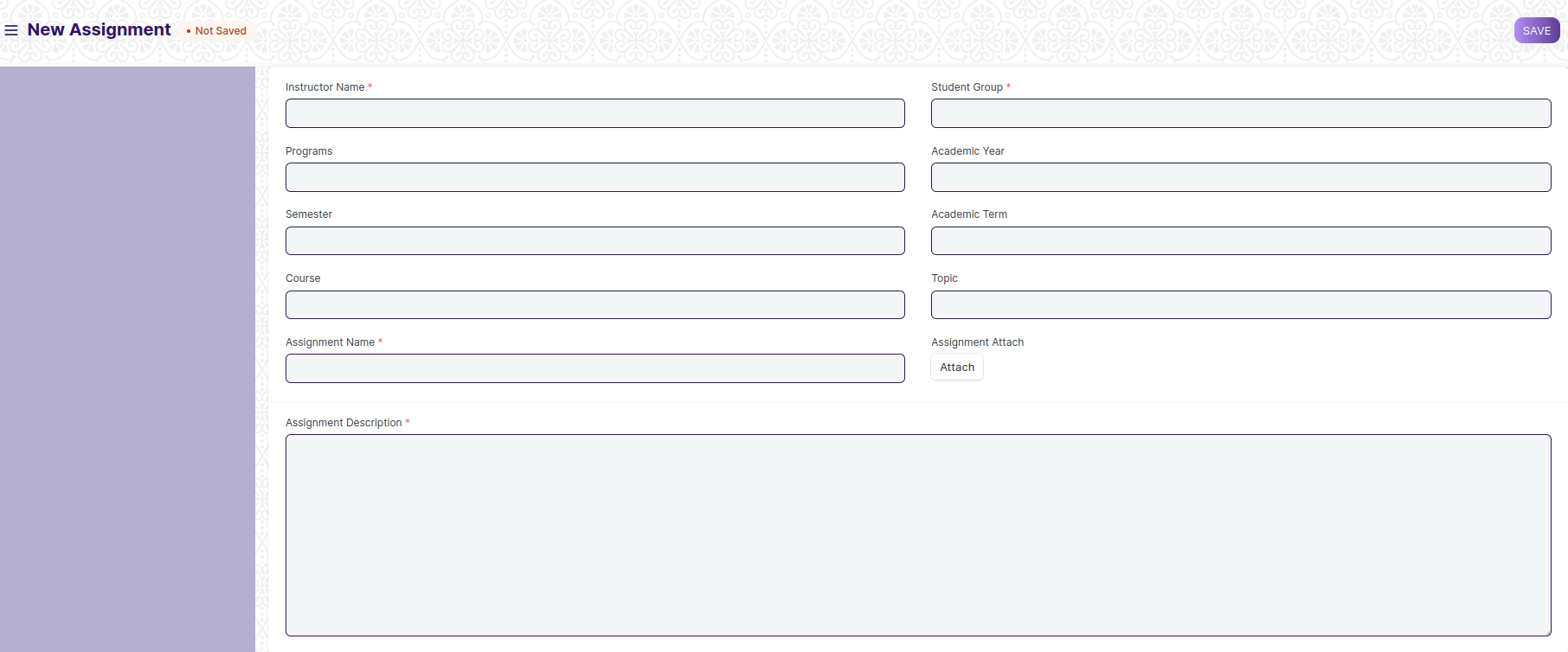


Figure : Assignment

## Assignment Upload

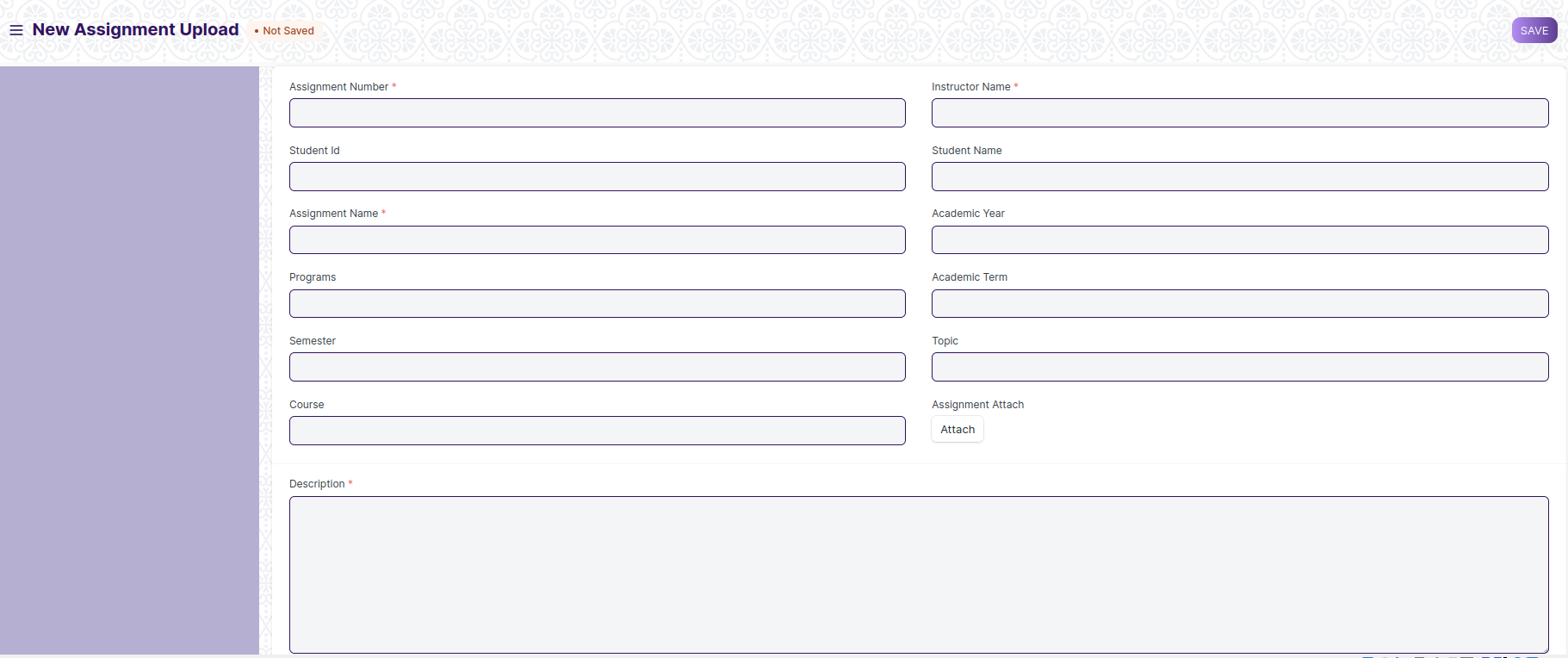


Figure : Assignment Upload

## Leave Application for Student

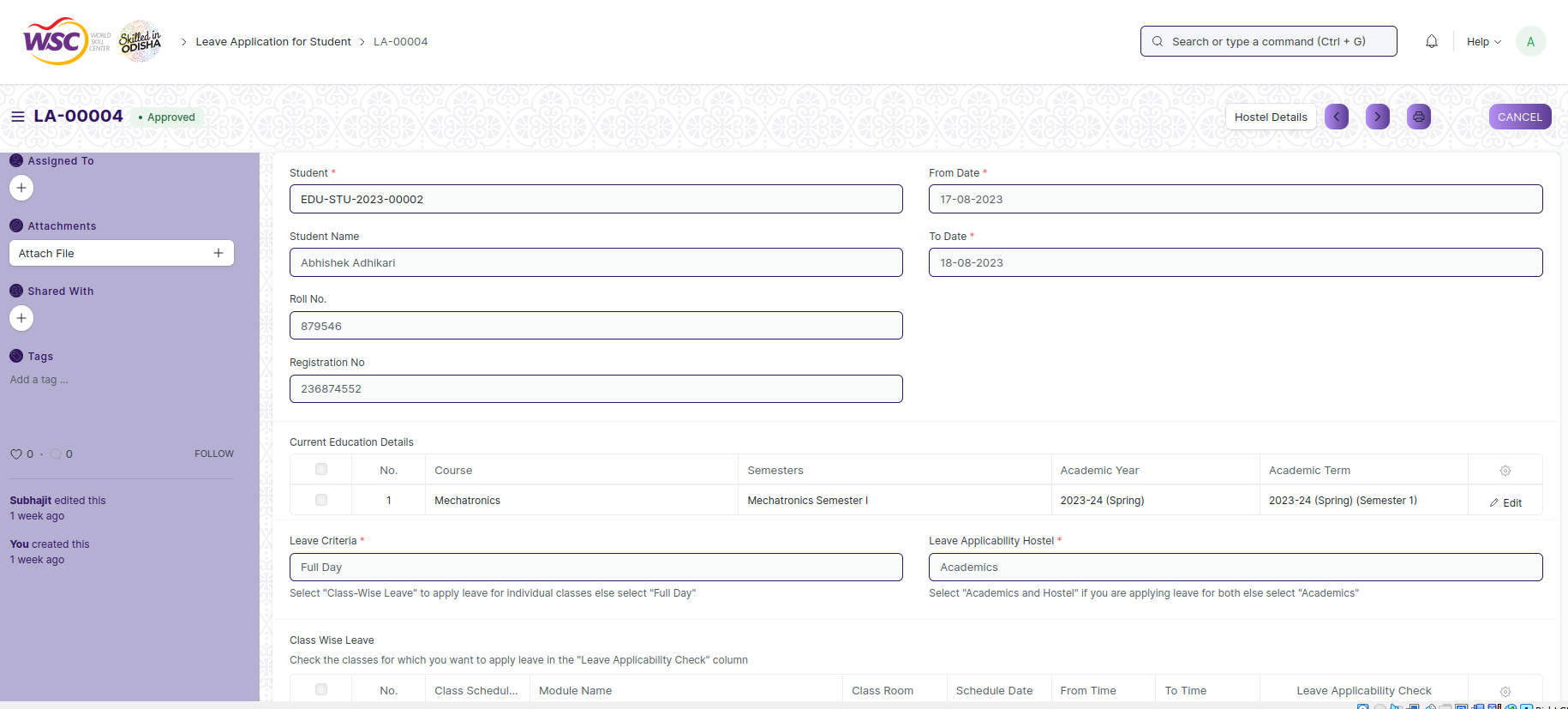


Figure : Leave Application for Student

## Reason for Leave

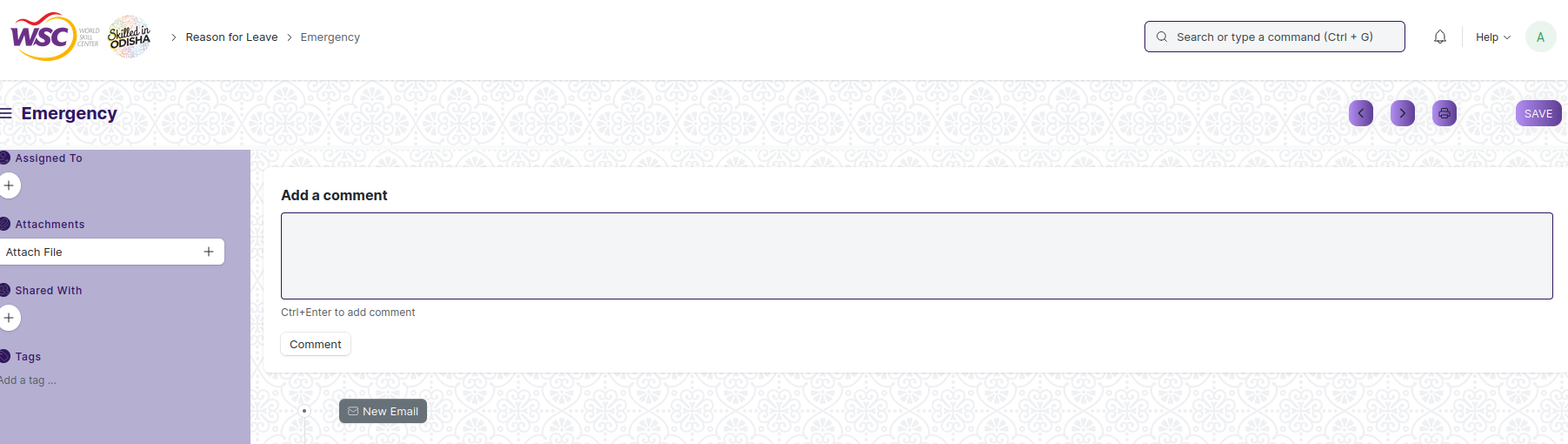


Figure : Reason for Leave

## Class Advisor and Manager Assignment

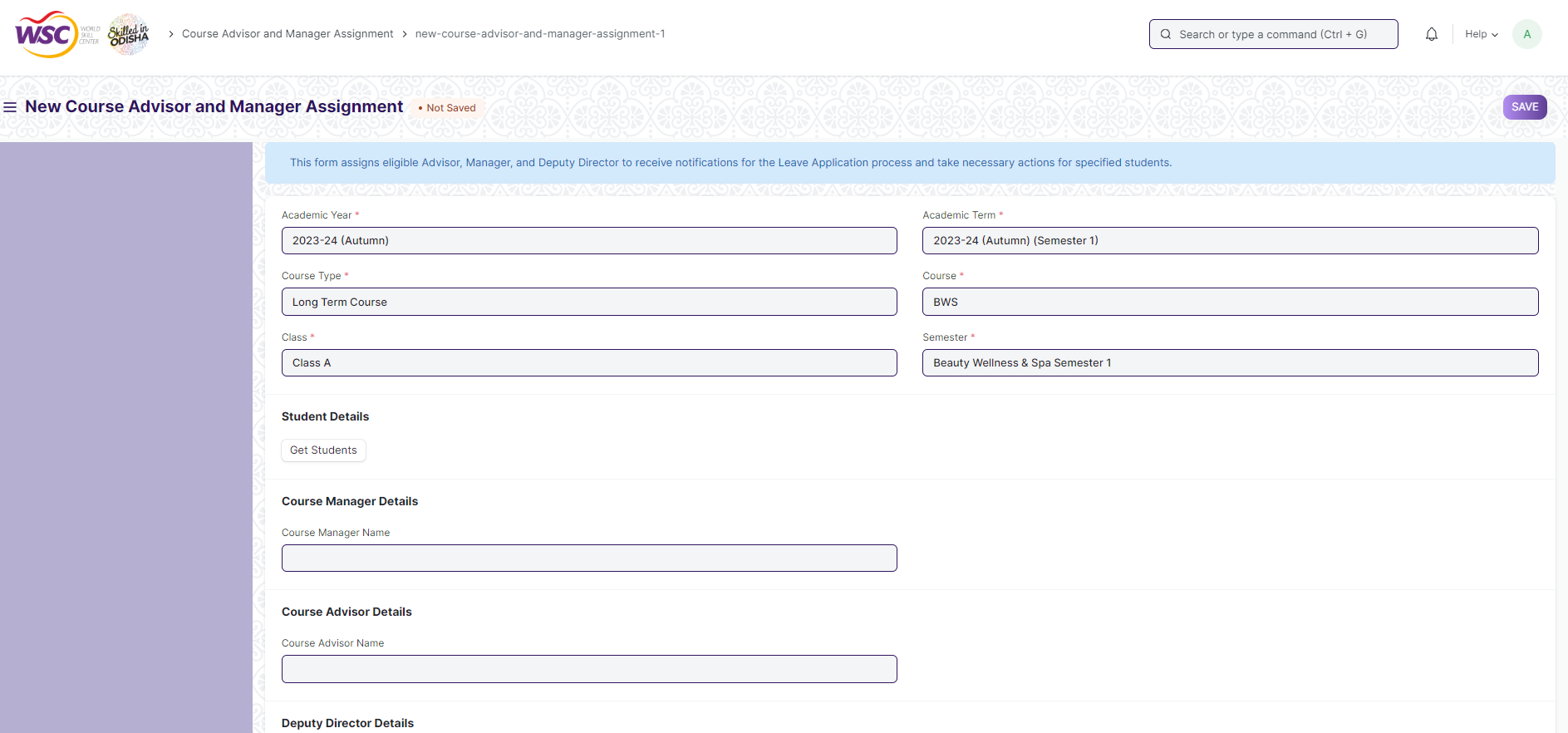


Figure : Class Advisor and Manager Assignment

## Mentor Allocation

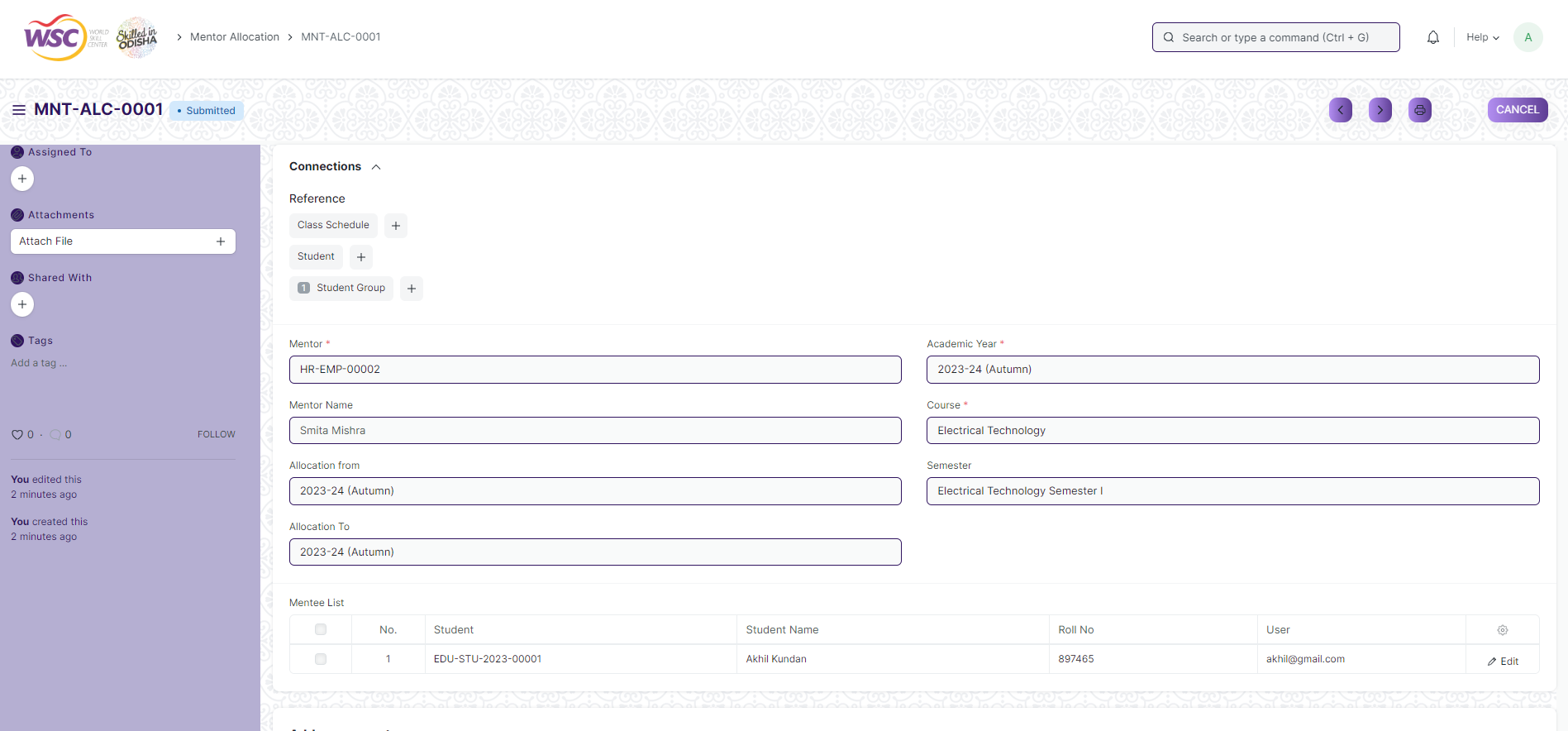


Figure : Mentor Allocation

## Mentor Mentee Communication

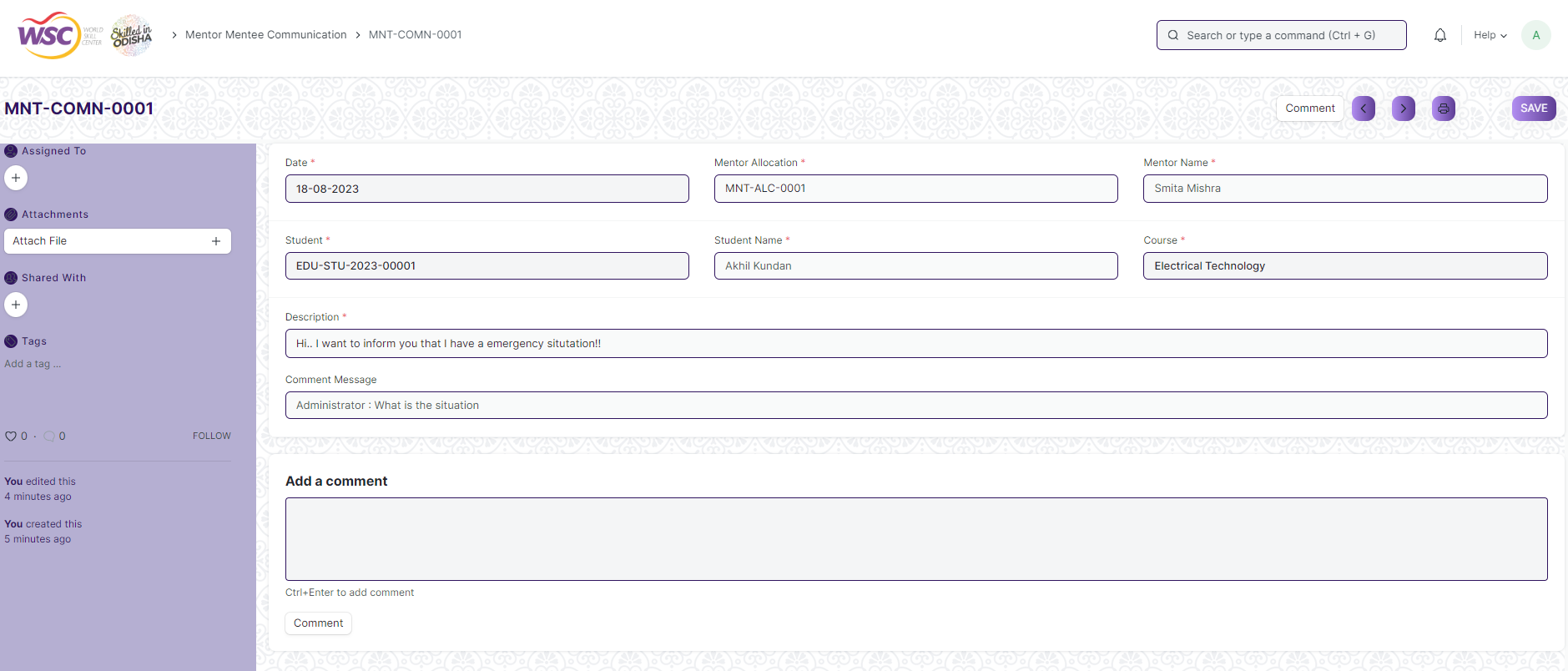


Figure : Mentor Mentee Communication

# 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