Software Requirement Specification

AC-2 COURSE REGISTRATION

Divyanshu Srivastava 21BCS080
Divyanshu Sharma 21BCS079
Devesh Kumar 21BCS075
Deepranjan Kumar 21BCS74
Aman Sharma 21BCS19

Faculty Mentor - Dr. Vinod Kumar Jain
Student Mentor - Pranav Bharadwaj 21BCS160

1 Introduction

1.1 Introduction about the Fusion – A brief Description

FusionIIIT stands as a transformative initiative within PDPM Indian Institute of Information Technology, Design and Manufacturing, Jabalpur. Developed with precision using Python 3.8 and leveraging the Django Web framework, this student-driven endeavor is reshaping the operational landscape of our institute. Going beyond mere functionality, FusionIIIT assumes the role of a strategic partner, seamlessly integrating and automating diverse functions to enhance the efficiency and coherence of campus life.

In essence, FusionIIIT serves as a sophisticated digital solution, addressing administrative complexities, optimizing academic processes, and extending its influence across various departments. On the administrative front, it adeptly manages intricate paperwork and procedural challenges. Within the academic sphere, it introduces a digital framework, thereby facilitating a more streamlined approach to learning and course management. Its impact transcends these realms, ensuring a holistic and well-coordinated environment throughout the campus.

To encapsulate, FusionIIIT transcends its utility as a mere tool; it embodies a sophisticated ally committed to augmenting the organizational structure and overall experience at PDPM IIITDM Jabalpur. In the realm of campus life, FusionIIIT stands as a testament to efficiency, cohesion, and innovation.

1.2 Purpose of the module:

Academic Course Registration (AC-2) is a software designed to manage different activities related to the Course Registrations of PDPM IIITDM Jabalpur. The module seeks to optimize resource utilization, to provide automated features and a smooth interface to the Academic staff, students and faculties to handle different Course registration and corresponding related activities .

1.3 Scope of the module

This software will take care of the following activities:

Admission, Courses Registration , add-drop ,replace , apply for backlog , fee deposit, manage schedule, course-list generation, roll list generation and more.

2. User/Actor characteristics

2.1 Student:

Student are the registered students of PDPM IIITDM Jabalpur who wish to registered themselves with the offered courses for the upcoming semester

Role : Initiates the registration related process

Specific Functionalities:

- They can view offered courses
- They can add, drop and replace the course
- They can do pre registration
- They can do final registration and can do the payment
- They can view the registration
- They can fill backlog form

2.2 Acad admin

Acad Admin of Institute is responsible for managing and displaying the course.

Role:- Manage the registration process.

Specific Functionalities:

- Manage schedule and view registration process.
- Generate list of course and roll list .
- Configure Pre-Registration and Main-Registration.

2.3 Faculty

Faculty are the teaching staff of the institute which are responsible for teaching the courses .

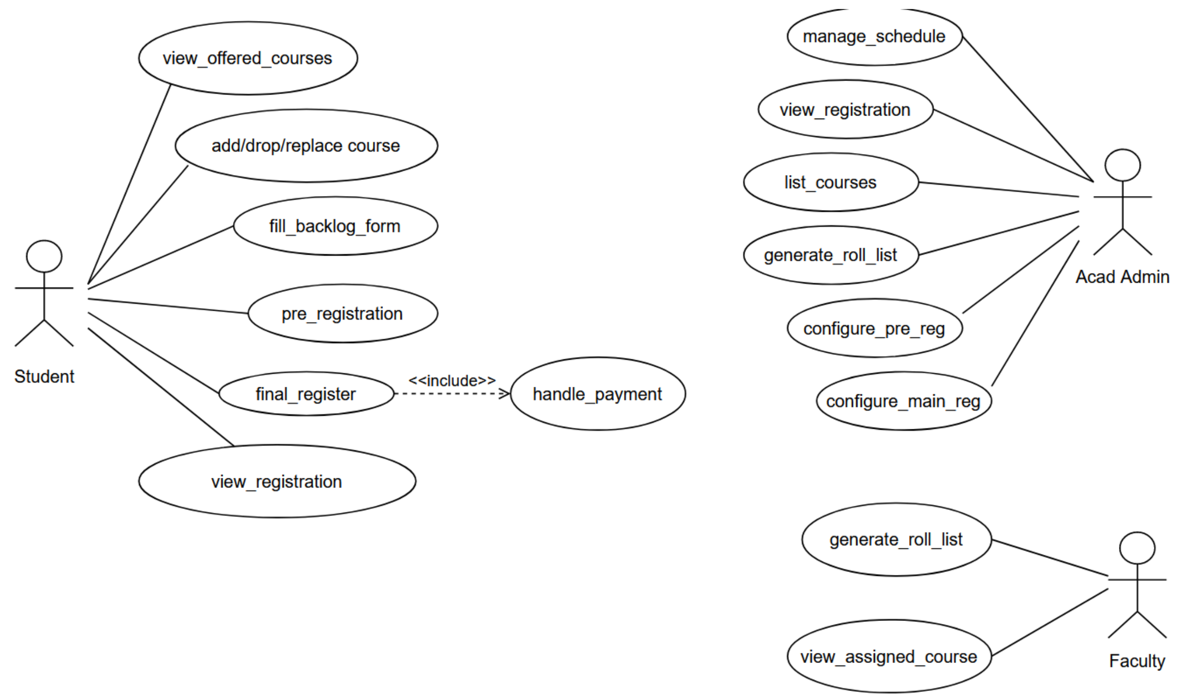
Role:- Viewing the courses under him/her and generating roll list

Specific Functionalities:

- View the courses that he will be teaching in the upcoming/current semester
- Generate the roll list of any course being taught by him/her

3. Functional Requirements

3.1 Use Case Diagram



3.2 Use Case Description

3.2.1 For Student:

Use Case #1

Use Case ID	**UC#1**
Use Case Name	**Add a course**

Description	The use case describes how to add courses to the course list of a UG student.
Actor(s)	UG Student
Precondition	UG Students must be logged in the system.
Main Flow	1.UG Student selects the option of adding courses from a list of options.
	2. UG Student is displayed a list of courses he/she can add in his course list.
	3. UG Student chooses a course from a list of courses which he/she wants to add.
	4. UG student submits the form.
Post-condition	1.Successful completion
	2. Course is added to the course list of UG students.
	3. UG Student is added to the attendance sheet of added courses.
Alternate flow	1. Constraints not met
	2. UG Student is prompted a message about constraints not met due to either of following conditions:
	3. UG Students with CPI less than 8.0 are not eligible.
	4. UG Students cannot add more than two courses in a semester.

	5. UG Students cannot have more than 26 credits in a semester.
	6. UG Student is redirected to the form.
Sub Flow	NIL

Use Case #2

Use Case ID	**UC#2**
Use Case Name	**Drop a course**
Description	The use case describes how to drop courses from the course list of a UG student.
Actor(s)	UG Student
Precondition	UG Students must be logged in the system.
Main Flow	1 .UG Student selects the option of dropping course from a list of options.
	2. UG Student is displayed on his/her course list.
	3.UG Student chooses the course from the list which he/she wants to drop.
	4. UG student submits the form.
Post-condition	1.Successful completion

	2. Course/s is/are removed from the course list of UG students.
	3. UG Student is removed from the attendance sheet of removed courses.
Alternate flow	1. Constraints not met
	2. UG Students cannot drop more than two courses in a semester.
	3. UG students cannot have less than 12 credits in a semester.
Sub Flow	NIL

Use Case # 3

Use Case ID	**UC#3**
Use Case Name	**Replace with Swayam Course**
Description	The use case describes how to replace a course from the course list with a swayam course of same credit.
Actor(s)	Student
Precondition	Students must be logged in the system.
Main Flow	1 . Student selects the option of replacing course from a list of options.
	2. Student is displayed on his/her course list.

	3.Student chooses the course from the list which he/she wants to replace.
	4.List of swayam courses that students have completed will be shown.
	5.Among completed swayam courses student shall choose the corresponding swayam course which he will replace with.
	6. Student submits the choices.
Post-condition	1.Successful completion
	2. Old Course/s is/are removed from the course list of UG students.
	3. UG Student is removed from the attendance sheet of removed courses.
Alternate flow	1. Constraints not met
	2. Students cannot replace the course or student haven't completed enough swayam course to replace.
Sub Flow	NIL

Use Case #4

Use Case ID	**UC#4**
Use Case Name	**Final Registration**

Description	The use case handles the student registration system.
Actor(s)	Student
Precondition	1.Student must be logged in the system.
	2.Final-Registration form which includes courses of the next semester is provided to the students.
Main Flow	1. Students confirm the courses to enroll for next semester.
	2. Student selects the fee payment mode and enters the transaction number.
	3. Students click the Register button to register.
Post-condition	Registration is completed successfully.
Alternate flow	Students can change the opted course.
Sub Flow	NIL

Use Case #5

Use Case ID	**UC#5**
Use Case Name	**Pre Registration**
Description	The use case handles the student registration system.

Actor(s)	Student
Precondition	1.Student must be logged in the system.
	2.Pre-Registration form which includes courses of the next semester is provided to the students.
Main Flow	1. Students select the courses to enroll for next semester.
	2. Students click the Register button to register.
Post-condition	Pre-registration done successfully.
Alternate flow	Return back to the same page with blank form.
Sub Flow	NIL

Use Case #6

Use Case ID	**UC#6**
Use Case Name	**Fill_backlog_form**
Description	This use case helps the student to fill the form for covering their backlog courses.
Actor(s)	Student
Precondition	1. Students must be logged in the system.

Main Flow	1. Student will select this option.
	2. Form will be opened on the screen, in which student have to fill all the details regarding the course and other required details .
	3. Student finally submits the form.
Post-condition	1. Submission successful.
Alternate flow	1. Form submission fails, and student is redirected back to same page.
Sub Flow	NIL

Use Case #7

Use Case ID	**UC#7**
Use Case Name	**view_registrarion**
Description	This use case describes how the Students can check their profile and courses after registration.
Actor(s)	Student

Precondition	1. Student must be logged in the system.
Main Flow	1. Student can select the option of view registration to see their profile and courses they have registered for.
Post-condition	1.All the related details will be displayed on the screen.
Alternate flow	1. Student haven't completed the registration yet , so nothing to display yet.
Sub Flow	NIL

Use Case #8

Use Case ID	**UC#8**
Use Case Name	**view_offered_courses**
Description	This use case help students to see all the offered courses to them in the upcoming semester.
Actor(s)	Student
Precondition	1. Student must be logged in the system.

Main Flow	1. Student clicks on this option.
	2. Courses based on the year , semester and branch of student are fetched from the database.
	3. List of all the courses that is being offered to the student will be displayed based on the year and branch of the student.
Post-condition	1. Student will see the list of courses.
Alternate flow	1. Problem in fetching courses from database .
Sub Flow	NIL

3.2.2 For Acad Admin:

Use Case #9

Use Case ID	**UC#9**
Use Case Name	**Manage_schedule**

Description	This use case describes how the Academic Person updates and adds the schedule of courses and time table.
Actor(s)	Acad Admin
Precondition	1. Academic Person must be logged in the system.
Main Flow	1. Academic person select this option from list of options.
	2.All the schedules is displayed.
	3. Option of adding/deleting/updating the schedules is displayed.
	4. Academic person can manage the schedule with these options.
Post-condition	1. Changes in schedule will be displayed to everyone.
Alternate flow	NIL
Sub Flow	NIL

Use Case #10

Use Case ID	**UC#10**
Use Case Name	**View_registration**

Description	This use case describes how the Academic Person checks the final registration of students for the semester
Actor(s)	Academic Person
Precondition	1. Academic Person must be logged in the system.
Main Flow	1.Academic Person selects the option of Checking Registrations of all students.
Post-condition	1.Student can check the updated registered courses through his link.
Alternate flow	NIL
Sub Flow	NIL

Use Case #11

Use Case ID	**UC#11**
Use Case Name	**Generate_roll_list**
Description	This use case allows an Academic Admin to generate a roll list for a specific course, containing student information such as names, ID numbers, and contact details.
Actor(s)	Acad Admin
Precondition	1. The Academic Admin is logged into the Fusion system. 2. The Course Registration module is accessible. 3. The registration for the desired course has been finalized.

Main Flow	1.The Academic Admin initiates the use case by accessing the "Generate Roll List" option within the Course Registration module
	2.The system prompts the Academic Admin to select the course for which they want to generate the roll list.
	3.The system retrieves the list of registered students for the selected course from the database.
	4.The system generates a roll list in a specified format (e.g., PDF, Excel, printable webpage).
	5.The system displays the generated roll list to the Academic Admin.
	6.The Academic Admin can download, print, or save the roll list as needed.
Post-condition	The roll list is generated and available for the Academic Admin's use.
Alternate flow	NIL
Sub Flow	NIL

Use Case #12

Use Case ID	**UC#12**
Use Case Name	**Configure_pre_reg**

Description	This use case allows an Academic Admin to configure the settings for the pre-registration period, including dates, eligibility criteria, and student permissions.
Actor(s)	Acad Admin
Precondition	1.The Academic Admin is logged into the Fusion system.
	2.The Course Registration module is accessible.
Main Flow	1.The Academic Admin initiates the "Configure Pre-Registration" section.
	2.The system displays the current pre-registration settings.
	3.The Academic Admin modifies the following settings as needed like start and end dates, eligible students groups etc.
	4.The Academic Admin confirms the changes.
	5.The system validates the inputs and saves the updated settings.
	6.The system displays a confirmation message indicating that the pre-registration settings have been successfully configured.
Post-condition	The pre-registration settings are updated and ready for the upcoming pre-registration period.
Alternate flow	1. If the Academic Admin enters invalid data (e.g., end date before start date), the system displays an error message and prompts for correction.

Sub Flow	NIL

Use Case #13

Use Case ID	**UC#13**
Use Case Name	**Configure_main_reg**
Description	This use case allows an Academic Admin to configure the settings for the main-registration period, including dates, eligibility criteria, and student permissions.
Actor(s)	Acad Admin
Precondition	1.The Academic Admin is logged into the Fusion system.
	2.The Course Registration module is accessible.
Main Flow	1.The Academic Admin initiates the "Configure Main-Registration" section.
	2.The system displays the current main-registration settings.

	3.The Academic Admin modifies the following settings as needed like start and end dates, eligible students groups etc;
	4.The Academic Admin confirms the changes.
	5.The system validates the inputs and saves the updated settings.
	6.The system displays a confirmation message indicating that the main-registration settings have been successfully configured.
Post-condition	The main-registration settings are updated and ready for the upcoming main-registration period.
Alternate flow	1. If the Academic Admin enters invalid data (e.g., end date before start date), the system displays an error message and prompts for correction.
Sub Flow	NIL

Use Case #14

Use Case ID	**UC#14**
Use Case Name	**Generate course list**
Description	This use case allows an Academic Admin to generate a list of courses offered in a specific semester or academic year, including relevant course details.
Actor(s)	Academic person
Precondition	1. The Academic Admin is logged into the Fusion system.

	2.Course information for the desired semester or year is available in the system.
	3.The Course Registration module is accessible.
Main Flow	1.The Academic Admin initiates the use case by accessing the "Generate Course List" option within the Course Registration module.
	2. The system prompts the Academic Admin to specify semester or academic year for which they want to generate the list and desired format for the list
	3.The system retrieves course information from the database based on the specified parameters.
	4.Fetched list of courses is displayed to the user.
Post-condition	List generated successfully.
Alternate flow	NIL
Sub Flow	NIL

3.2.3 For Faculty:

Use Case #15

Use Case ID	**UC#15**

Use Case Name	View Assigned courses
Description	This use case allows an Faculty to view list of courses that he is going to teach in upcoming semester.
Actor(s)	Faculty
Precondition	1. The Faculty is logged into the Fusion system.
	2.Courses information for the desired semester or year is available in the system.
Main Flow	1.The Faculty initiates the use case by accessing the "Get courses under him" option within the Course Registration module.
	2.The system retrieves course information from the database based on the specified parameters.
	3.Fetched list of courses is displayed to the user.
Post-condition	List generated successfully.
Alternate flow	1.Faculty is not teaching any courses. 2. List of courses is not yet updated in the database.
Sub Flow	NIL

Use Case #16(same as gen_roll_list for academic person)

Use Case ID	**UC#16**
Use Case Name	**Generate Roll list**
Description	This use case allows an Faculty to generate a roll list for a specific course, containing student information such as names, ID numbers, and contact details.
Actor(s)	Faculty
Precondition	1. The Faculty is logged into the Fusion system.
	2.Course information for the desired semester or year is available in the system.
	3.The Course Registration module is accessible.
Main Flow	1.The Faculty initiates the use case by accessing the "Generate Roll List" option within the Course module
	2.The system prompts the Faculty to select the course for which they want to generate the roll list.
	3.The system retrieves the list of registered students for the selected course from the database.
	4.The system generates a roll list in a specified format (e.g Excel).
Post-condition	Roll List generated successfully.
Alternate flow	NIL

Sub Flow	NIL

3.3 Other Functional Requirements

1. This module will use notification and alert to notify the students about the added courses, scheduled time for pre and final registration etc.
2. This software should be compatible with different browsers like Google Chrome, Firefox, Edge and Safari. Users should be able to use and navigate on any browser available.
3. The system should raise errors and invalid flags when users go opposite to the exceptions.

3.4 Other Constraints

3.4.1 User Interfaces

The color palette should match the design and pattern of the FusionIIIT dashboard. All the buttons should be clickable and there should be smooth navigation from one functionality to another. Inter module navigation should be reachable from within the module. The UI should be easy to use and intuitive.

3.4.2 Tech stack

Web application uses Django framework for frontend and backend functionality. PostgreSQL is used as the RDBMS.

Mobile application uses Flutter SDK for its frontend and Django API for its backend. MySQL is used for the database.

3.4.3 Business rules

1. There should be eligibility criteria for admission of prospective students after the system has processed their application.
2. The system should display a list of available courses for registration.
3. Students should meet the course prerequisites before allowing registration.
4. Add/drop time period should be specified to students.
5. Automated processing of backlog applications with notification outcomes.
6. Dynamic course and roll list generation.

4. Non-Functional Requirements

4.1. Performance:

- Response Time: The system should respond to user actions within reasonable time frames, even during peak registration periods.
- Scalability: The system should be able to handle a growing number of students and courses without significant performance degradation.
- Load Testing: Conduct thorough load testing to identify and address potential bottlenecks.

4.2. Availability:

- Uptime: The system should be highly available during registration periods, with minimal downtime for maintenance or outages.

4.3. Security:

- Authentication and Authorization: Enforce strong authentication mechanisms for users and implement role-based access controls to sensitive data and functionalities.
- Data Protection: Protect sensitive student and course information from unauthorized access, modification, or disclosure.

4.4. Maintainability:

- Modular Design: Structure the code in a modular and well-organized manner to facilitate updates and bug fixes.
- Documentation: Maintain comprehensive and up-to-date documentation for developers and administrators.
- Version Control: Use version control systems to track code changes and manage updates effectively.

4.5. Scalability:

- Capacity: The system should be able to accommodate expected growth in student enrollment and course offerings without significant performance degradation.
- Architecture: Design the system with scalability in mind, using technologies and architectures that can handle increased load.

5. Module Dependencies with other Fusion Modules

5.1 UI Level:

1. Integration within Fusion's UI:

- Placement within Navigation Menu: The module should be integrated into Fusion's main navigation menu, likely under a section dedicated to academic activities.
- Distinct Sections for Actors: Within the module, there should be clear sections or tabs for students and academic admins, ensuring a tailored experience for each actor.
- Consistent Styling and Navigation: The module's UI elements should align with Fusion's overall visual design and navigation patterns to maintain a cohesive user experience.

2. Actor-Specific Functionalities:

Students:

- View Offered Courses:
  - Access via a dedicated "Course Catalog" or "Course Search" tab within the module.
- Add Courses:
  - Provide a mechanism to add selected courses to a "Cart" or "Registration List."
  - Display clear feedback on course availability and potential conflicts.
- Drop Courses:
  - Provide a mechanism to drop selected courses.
  - Display clear message whether the course was successfully dropped and if yes which is the new course allotted if applicable.
- Replace Courses:
  - Provide a mechanism to replace selected courses with courses of Swayam
  - Proper list of Swayam courses offered by the department of the particular student should be displayed.
- Finalize Registration:
  - Present a summary of selected courses for confirmation.
  - Handle payment processing (if applicable).
  - Generate and display a registration confirmation.
- View Registration Status:
  - Display a comprehensive list of registered courses with details (e.g., meeting times, instructors, credits).
  - Allow for printing or exporting registration information.
- Fill Backlog Form:
  - Provide a dedicated form within the module for students to apply for backlog clearance.
  - Integrate with relevant approval workflows and notifications.
- Pre-Registration:
  - Offer a separate section for students to indicate their course preferences for upcoming semesters.
  - Indicate any pre-requisites or course availability constraints.

- Final-Registration:
  - Offer a separate section for students to finalize their course preferences for upcoming semesters.

Academic Admins:

- Manage Course Schedule:
    - Provide a calendar-based interface for creating and managing course schedules.
    - Integrate with faculty assignment and classroom allocation systems (if applicable).
- View Registration Details:
    - Display lists of registered students for each course, with options to filter and sort data.
    - Allow for exporting student lists for further analysis or reporting.
- List Courses:
    - Provide a comprehensive view of all courses offered, with options to edit course information.
- Generate Roll Lists:
    - Allow admins to generate class rosters for each course, with options for downloading or printing.
- Configure Pre-Registration and Main Registration Settings:
    - Offer dedicated settings sections within the module for admins to adjust registration deadlines, rules, and permissions.

Faculty:

- Generate Roll Lists:
    - Allow Faculty to generate class roll lists for each course, with options for downloading or printing.
- View assigned courses:
    - Provide a comprehensive view of all courses offered by the faculty that he is going to teach in the upcoming semester .

5.2 DB Level Dependencies:

1. Students:

   a. Owner: Course Management Module
   b. User: Course Registration Module (for admin data)
   c. Potential Fields: student_id, username, password, role (student/admin), other personal details
2. Courses:
   a. Owner: Course Management Module (if separate) or Course Registration Module
   b. User: Course Registration Module, Academic Admin Module, Timetable Module, Fee Management Module
   c. Potential Fields: course_id, course_name, department, credits, prerequisites, instructor_id, semester
3. Registration:
   a. Owner: Course Registration Module
   b. User: Course Registration Module, Academic Admin Module, Transcript Module, Fee Management Module
   c. Potential Fields: registration_id, student_id, course_id, semester, grade (optional), payment_status
4. Schedule:
   a. Owner: Timetable Module (if separate) or Course Registration Module
   b. User: Course Registration Module, Academic Admin Module, Faculty Module
   c. Potential Fields: course_id, section_id, timeslot_id, classroom_id, instructor_id
5. Payments: (if applicable)
   a. Owner: Fee Management Module
   b. User: Course Registration Module
   c. Potential Fields: payment_id, student_id, amount, payment_date, registration_id
6. Fee:
   a. Owner: Mess Module,Accounts Module
   b. User: Course Registration Module
   c. Potential Fields: student_id, amount, payment_date, registration_id, deadline

5.3 Module Level Dependencies:

1. Academic Information Module:

- Dependency Direction: Course Registration Module depends on Academic Information Module.
- Purpose:
  - To retrieve student academic data (e.g., program, year of study, GPA, completed courses) for eligibility checks and prerequisite enforcement.
  - To update student academic records with registered courses and grades.

2. Notifications Extension:

- Dependency Direction: Course Registration Module depends on Notifications Extension.
- Purpose:
  - To trigger notifications for various events:
    - Registration confirmations and reminders
    - Backlog clearance approvals
    - Course schedule changes
    - Registration deadlines

Additional Module-Level Dependencies (Potential):

- Authentication/User Management Module: For user authentication and authorization.
- Fee Module: For handling course fees and payment processing.
- Exam Module: For recording student grades and generating transcripts.