Software Requirements Specification

OS-3 NOTIFICATIONS (WEB)

Team Details

Dr.Neelam Dayal (**Faculty Mentor**)

Shaik Nida Afsheen (21BCS189)

Pateel Rishitha Reddy (21BCS155)

Vandana Manoj Kumar (21BCS130)

Rongali Kishan Koushal (21BCS179)

Sudhanshu Patil(21BCS209) (**Student Mentor)**

1. Introduction

1.1 Introduction about the Fusion

Fusion IIIT stands as a testament to the seamless integration and automation of diverse functions within IIIITDM JABALPUR Crafted with precision using Python 3.8 and powered by the Django Web framework, this initiative is a student-driven endeavor designed to elevate the institute's operational landscape. Encompassing everything from efficient administration management to academic prowess and miscellaneous departmental tasks, Fusion IIIT is a holistic solution that harmonizes the intricacies of campus life.

Imagine it as a digital wizard that takes care of everything, from organizing the administrative stuff to making academics smoother. It's not just limited to the usual tasks Fusion IIIT jumps into various departments and sections, making sure every corner of campus life runs smoothly. In the admin side, it handles the complicated paperwork and processes. For academics, it brings a digital touch, making learning and managing courses easier. But it doesn't stop there, Fusion IIIT is like a friendly companion for all the different parts of the campus, making sure everything works well.
In simpler terms, Fusion IIIT is not just a tool – it's a helpful friend, making life at PDPM IIITDM Jabalpur more organized and enjoyable for everyone.

1.2 Purpose of the module

Enhance user engagement by delivering timely and relevant information through notifications.

Facilitate communication by updating users on events, messages, and interactions within the Django project.

Provide feedback to users, confirming successful actions and alerting them to errors.

1.3 Scope of the module

Notify the User about relevant events, such as interactions with their content and updates from others students and faculty.

For every module there will be different type of notifications according to the unique needs and activities associated with each individual module.

2.User/Actor Description(characteristics):

2.1 Students/Faculty/Staff:

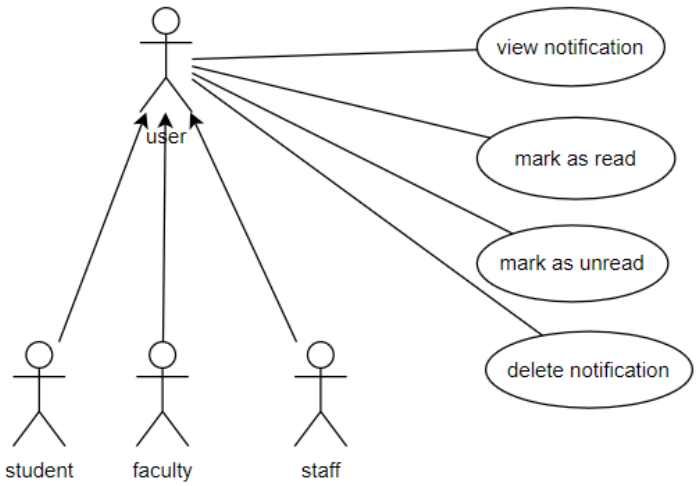
Role: Can generate a notification relevant to the particular module.

Specific Functionalities:

1.The user can open the module by clicking on the module name present in the notification while viewing it.
2.The user can click on mark as read on a new notification.
3.The user can click on mark as unread on a already marked as read notification.
4.The user can delete a notification.

3. Functional Requirements

3.1 Use Case Diagram



Note

Notifiaction panel should be able to retrieve any announcement made or any status change through any of the department and it should be displayed to authorized personnel only

3.2 Use case Description

This section describes each use case Description in the use case diagram in all details.

1)

UC ID	UC#1	
Use case Name	**view notification**	
Description	The " View Notification " use case allows the User to view notifications about their activity from different modules through the Fusion portal.	
Actor	User (Student/Faculty/Staff)	
Precondition	The User is logged in into the portal and has minimum 1 notification.	
Main Flow	M1	The User navigates to the “Notifications” section in the dashboard.
	M2	Then the user gets displayed the list of their notifications.
	M3	The User selects a notification to view.
	M4	The User can now read the content of that notification and can see from which module it came from[A1].

Alternate Flow	4		If the user clicks on the module name in the notification, they should be redirected to that particular module from which the notification came from.
Sub Flow	NIL		
Global Alternate Flow	GA1		If a technical error occurs during the execution of any action (e.g., database failure, server issues), the system displays an error message and logs the incident.

2)

UC ID	UC#2	
Use case Name	**mark as read**	
Description	The "Mark as Read" use case allows the User to mark a new notification as read if they wish not to have that notification as new through the Fusion portal.	
Actor	User (Student/Faculty/Staff)	
Precondition	The User is logged in into the portal and has minimum 1 new notification.	
Main Flow	1	The User navigates to the “Notifications” section in the dashboard.
	2	Then the user gets displayed the list of their notifications.
	3	The User hovers over a new notification to mark as read.
	4	The User marks the notification as read from the options when the notification is hovered. [A1]

Post conditions	That particular new notification no longer remains new as is marked as read		
Alternate Flow	A1	4	The user cancels the mark as read action.
Sub Flow	NIL		
Global Alternate Flow	GA1	If a technical error occurs during the execution of any action (e.g., database failure, server issues), the system displays an error message and logs the incident.	

3)

UC ID	UC#3	
Use case Name	**mark as unread**	
Description	The "Mark as Unread" use case allows the User to mark an already read notification as unread if they wish not to have that notification as read through the Fusion portal.	
Actor	User (Student/Faculty/Staff)	
Precondition	The User is logged in into the portal and has minimum 1 already marked as read notification.	
Main Flow	1	The User navigates to the “Notifications” section in the dashboard.
	2	Then the user gets displayed the list of their notifications.
	3	The User hovers over an already marked as read notification to mark as unread.
	4	The User marks the notification as unread from the options when the notification is hovered. [A1]

Post conditions	That particular already marked as read notification now becomes new.		
Alternate Flow	A1	4	The user cancels the mark as unread action.
Sub Flow	NIL		
Global Alternate Flow	GA1	If a technical error occurs during the execution of any action (e.g., database failure, server issues), the system displays an error message and logs the incident.	

4)

UC ID	UC#4	
Use case Name	**delete notification**	
Description	The "Delete Notification" use case allows the User to delete a notification from the notifications they get displayed through the Fusion portal.	
Actor	User (Student/Faculty/Staff)	
Precondition	The User is logged in into the portal and has minimum 1 notification.	
Main Flow	1	The User navigates to the “Notifications” section in the dashboard.
	2	Then the user gets displayed the list of their notifications.
	3	The User hovers over a notification to delete.
	4	The User deletes the notification from the options when the notification is hovered. [A1]

Post conditions	That particular notification now gets deleted and longer displayed to the User.		
Alternate Flow	A1	1	The user cancels the delete action.
Sub Flow	NIL		
Global Alternate Flow	GA1	If a technical error occurs during the execution of any action (e.g., database failure, server issues), the system displays an error message and logs the incident.	

3.3. Other Functional Requirements

1) All the modules will make use of the **Notification module** for sending notifications and alerts to various actors involved in the module.

2) A module typically should allow the generation of notifications and their visibility for the relevant users.

3) A module should implement the block of code of the notifications module to send the notifications of that specific module smoothly.

3.4 Other constraints

3.4.1 User Interfaces

The user interface should comply with the color scheming and dashboard design of the FUSIONIIIT. Users should be able to navigate from one functionality to other. Inter module navigation should be smooth. All the functionalities should be easy to use and no specific training should be required for the usage of the module.

3.4.2 Tech Stack Used

1) Backend: Django (Python Based Web-Framework).
2) Frontend:(HTML, CSS, JavaScript).

4. Non- Functional Requirements

4.1 Performance:

The system should respond to user interactions quickly. Response time for Generating notification, Marking as read and unread should be less.

4.2 Scalability:

The system should handle a mass of concurrent users. System performance should be evaluated under increasing load conditions.

4.3 Availability:

The system should be available 99.9% of the time.

4.4 Security:

Ensure data confidentiality and integrity. Role-based authorization ensures that users can only perform actions relevant to their designated roles.

5. Module dependencies with other fusion modules

5.1. UI Level

Integration in Fusion :-

At UI level, Notification Module will seamlessly integrates with other modules to send their respective notifications.

After the user sign in into the fusion portal in the dashboard there will be a notification bar, the following functionalities will be available:

- Mark as read
- Mark as unread
- Delete Notification
- Description of the notification and the module name by clicking the module name it will be redirected to that particular module.

5.2. DB Level Dependencies:

1. All the modules will be able to see the information of the notification which is stored in the database.
2. Every module will share the Notification schema with the notifications module.

5.3. Module Level Dependencies:

1. In the **file tracking module**, upon sending a notification, the sender will receive an acknowledgment notification confirming that the file has been sent successfully. Simultaneously, the recipient will be notified that the file has been received.
2. The doctor appointment form within the **health center module** is not working, while the ambulance request function is operational, and ambulance requests are directed to the compounder. After submitting feedback, it is unclear who receives the notifications.
3. Notifications are not functioning in the **visitors hostel module** due to its incomplete functionality. The form details are not being transmitted to the

caretaker responsible for the visitors' hostel.

4. Within the **gymkhana module,** all notifications are operational, except for the club membership form, where the selection of the club is not functioning, rendering it inactive. . The co-convener can send voting poll notifications to specific batches, event notifications to all batches, and session notifications to all batches within the gymkhana module.
5. In the **research procedures module**, the patent form is not working, so it is currently non-functional in generating notifications. Similarly, within the other academic procedures module, both the **leave module** and **assistant - claimship module** forms have notification functions defined, but they are not working. These are currently not active for generating notifications.
6. In the **complaint system module**, once a student submits a complaint, a notification will be forwarded to the designated caretaker assigned to handle that specific complaint.
7. Within the **mess module**, notifications will be accessible upon submission of feedback, requests for changes in the mess menu, and leave or vacation requests from the staff. Additionally, when a student is added to the mess committee, they will receive a notification informing them of their new role.
8. In **Department module** the publication of an announcement by the department head, notifications will be sent to both the department's students and the head of the department. These notifications will be accessible in the announcements section.