







Issue Number#	User Story Title	Description/Associated Tasks	Difficulty Level (1,2,3)	🔒 Priority	🔒 Risk	Time Spent	👤 Assignee	# Sprints
#1	US#001:As a user,I want to log into my account with a designated role (Normal User or Admin), So that I can access the relevant features of the application.	#1.1 Design and implement the login UI:	MED (2)	HIGH	HIGH	5		1
		#1.1.1 Create a form with fields for email and password.	MED (2)	HIGH	MEDIUM	2		1
		#1.1.2 Add a login button.	EASY (1)	HIGH	LOW	1		1
		#1.1.3 Provide appropriate error messages for invalid credentials/existing members.	HARD (3)	MEDIUM	MEDIUM	2		1
		#1.2 Set up authentication logic:	HARD (3)	HIGH	HIGH	9		1
		#1.2.1 Implement a backend authentication service.	HARD (3)	HIGH	HIGH	3		1
		#1.2.2 Verify user credentials against the database.	HARD (3)	HIGH	HIGH	3		1
		#1.2.3 Assign a session or token upon successful login.	MED (2)	HIGH	HIGH	3		1
		#1.3 Implement role-based access control:	HARD (3)	MEDIUM		14		1
		#1.3.1 Store user roles in the database.	HARD (3)	MEDIUM	MEDIUM	3		1
		#1.3.2 Ensure Normal Users and Admins have distinct access levels.	HARD (3)	MEDIUM		3		1
		#1.4 Encrypt and secure passwords:	HARD (3)	MEDIUM	MEDIUM	4		1
		#1.4.1 Use hashing algorithms (e.g., bcrypt) for storing passwords securely.	HARD (3)	MEDIUM	HIGH	4		1
		#1.5 Test login functionality:	HARD (3)	LOW	LOW	8		1
		#1.5.1 Perform unit and integration tests to validate login behavior.	HARD (3)	LOW	LOW	4		1
		#1.5.2 Ensure unauthorized access is prevented.	HARD (3)	LOW	LOW	4		1
#2	US#002: As a new user, I want to create an account with my email, password, and role (Normal User or Admin), So that I can log in and access the platform.	#2.1 Design and implement the registration UI:	MED (2)	HIGH		13		1
		#2.1.1 Create a form with fields for email, password, and role selection.	EASY (1)	HIGH	LOW	3		1
		#2.1.2 Add validation for required fields and incorrect inputs.	MED (2)	HIGH		4		1
		#2.1.3 Provide an appropriate error message for invalid inputs or duplicate accounts.	MED (2)	HIGH	MEDIUM	4		1
		#2.1.4 Display a success message upon successful registration.	EASY (1)	LOW	LOW	2		1
		#2.2 Develop backend logic for user registration:		HIGH	HIGH			1
		#2.2.1 Implement an API endpoint to handle registration requests.	HARD (3)	HIGH				1
		#2.2.2 Store user details in the database.	MED (2)	HIGH	MEDIUM			1
		#2.2.3 Ensure role selection is stored correctly in the database.	MED (2)	MEDIUM	LOW			1
		#2.2.4 Prevent duplicate email registrations by enforcing unique constraints.	HARD (3)	HIGH	HIGH			1
		#2.3 Implement password hashing and security measures:	HARD (3)	HIGH	HIGH			1
		#2.3.1 Use bcrypt or an equivalent hashing algorithm for password encryption.	HARD (3)	HIGH	HIGH			1
		#2.3.2 Ensure that passwords are never stored in plain text.	MED (2)	HIGH				1
		#2.3.3 Implement password strength validation to enforce security standards.	MED (2)	HIGH	MEDIUM			1
		#2.4 Redirect users upon successful registration:	MED (2)	MEDIUM	HIGH			1
		#2.5.1 Upon successful registration, redirect users to the login page.	EASY (1)	MEDIUM	MEDIUM			1
		#2.5.2 Display a confirmation message informing users they have registered successfully.	MED (2)	MEDIUM	HIGH			1
		#2.5 Test registration functionality:	HARD (3)	LOW	LOW			1
		#2.6.1 Perform unit and integration tests for the registration API.	HARD (3)	LOW	LOW			1
		#2.6.2 Validate error handling for incorrect inputs and duplicate accounts.	HARD (3)	LOW	LOW			1
#3	#US003: As an admin, I want to create teams, assign users to specific channels, and moderate discussions so that users can communicate in an organized and controlled environment.	#2.6.3 Ensure database entries are correctly created upon successful registration.	HARD (3)	LOW	LOW			1
		#2.6.4 Test that unverified users cannot log in if email confirmation is enabled.	HARD (3)	LOW	LOW			1
		#3.1 Design and implement the UI for team creation:						1
		#3.1.1 Create a form for admins to input a team name and description.						1
		#3.1.2 Ensure validation for required fields and prevent duplicate team names.						1
		#3.1.3 Display a success message upon successful team creation.						1
		#3.2 Develop backend logic for team creation:						1
		#3.2.1 Implement an API endpoint to handle team creation requests.						1
		#3.2.2 Store team details in the database.						1
		#3.3 Implement user-team mapping:						1
		#3.3.1 Store user-team relationships in the database.						1
		#3.3.2 Allow admins to assign users to specific teams.						1
		#3.3.3 Ensure users can only see the teams they belong to.						1
		#3.4 Implement notification functionality:						1
		#3.4.1 Notify users when they are added to a team.						1

ISSUE NUMBER#	USER STORY TITLE	DESCRIPTION/ASSOCIATED TASKS	DIFFICULTY LEVEL (1,2,3)	 PRIORITY	 RISK	TIME SPENT	 ASSIGNEE	# SPRINT
#4	#US004: As an admin, I want to assign users to specific channels within a team so that they can participate in relevant discussions.	#3.4.2 Ensure notifications appear in the UI.						1
		#3.5 Test team creation functionality:						1
		#3.5.1 Perform unit and integration tests for the team creation API.	HARD (3)	LOW	LOW			1
		#3.5.2 Validate error handling for duplicate team names.	HARD (3)	LOW	LOW			1
		#3.5.3 Verify that users receive notifications when added to a team	HARD (3)	LOW	LOW			1
		#4.1 Design and implement the UI for channel assignment:						1
		#4.1.1 Provide an interface for admins to select users and assign them to channels.						1
		#4.1.2 Ensure validation to prevent duplicate assignments.						1
		#4.1.3 Display a success message upon successful assignment.						1
		#4.2 Develop backend logic for user-channel mapping:						1
		#4.2.1 Implement an API endpoint to handle channel assignment requests.						1
		#4.2.2 Store user-channel relationships in the database.						1
		#4.2.3 Ensure users can only access the channels they are assigned to.						1
		#4.3 Implement notification functionality:						1
		#4.3.1 Notify users when they are assigned to a new channel.						1
		#4.3.2 Ensure notifications appear in the UI.						1
		#4.4 Test channel assignment functionality:	HARD (3)	LOW	LOW			1
		#4.4.1 Perform unit and integration tests for the channel assignment API.	HARD (3)	LOW	LOW			1
		#4.4.2 Ensure only admins can assign users to channels.		LOW	LOW			1
		#4.4.3 Validate error handling for duplicate assignments.		LOW	LOW			1
#5	#US005 -As a user, I want to see the channels I have access to so that I can participate in relevant discussions.	#4.4.4 Verify that users receive notifications when added to a channel.		LOW	LOW			1
		#5.1 Design and implement the UI for channel visibility:						1
		#5.1.1 Display a list of channels available to the logged-in user.						1
		#5.1.2 Ensure the UI dynamically updates when a user gains or loses access to a channel.						1
		#5.1.3 Allow users to click on a channel to open and view messages.						1
		#5.2 Develop backend logic for channel visibility:						1
		#5.2.1 Implement an API endpoint to fetch the list of channels for a user.						1
		#5.2.2 Ensure the system filters channels based on user roles and assignments.						1
		#5.2.3 Restrict access to channels a user is not assigned to.						1
		#5.3 Implement real-time updates for channel visibility:						1
		#5.3.1 Update the UI when a user is added or removed from a channel.						1
		#5.3.2 Ensure changes reflect without requiring a page refresh.						1
		#5.4 Implement the user to be able to. create teams:						1
		#5.4.1 Verify that the user can create team						1
		#5.4.2 Verify that the user can create channels						1
		#5.4.3 Verify that the user can add users to his created channel						1
		#5.5 Test channel visibility functionality:	HARD (3)	LOW	LOW			1
		#5.5.1 Perform unit and integration tests for the channel visibility API.	HARD (3)	LOW	LOW			1
		#5.5.2 Verify that admins see all channels while normal users see only assigned ones.	HARD (3)	LOW	LOW			1
		#5.5.3 Validate real-time updates when a user's access to a channel changes.	HARD (3)	LOW	LOW			1
#6	#US006 -As a user, I want a global chatroom so I can chat with any user/admin that are online	#5.5.4 Ensure unauthorized users cannot access restricted channels.	HARD (3)	LOW	LOW			1
		#6.1 Design and implement the UI for chatroom visibility:						1
		#6.1.1 Display a chat box large enough to view incoming messages						1
		#6.1.2 Ensure chatroom displays the user name, status and ID						1
		#6.1.3 Ensure that user can distinguish between sent and received messages						1
		#6.1.4 Include a session-based storage that stores the chat messages						1
		#6.2 Develop backend logic for chatroom visibility:						1
		#6.2.1 Implement a logic that communicate with the client side in real time.						1
		#6.2.2 Ensure the messages of the user is broadcasted to all users						1
		#6.3 Implement real-time updates for chatroom visibility:						1
		#6.3.1 The messages should be uploaded in the chatroom across any open session instantly						1
		#6.3.2 The messages should not be deleted when user closes or open the global chat tab or r						1

ISSUE NUMBER#	USER STORY TITLE	DESCRIPTION/ASSOCIATED TASKS	DIFFICULTY LEVEL (1,2,3)	 PRIORITY	 RISK	TIME SPENT	 ASSIGNEE	#	SPRINT
#7	#US007 - As a user, I want to have a settings page so I can view my account information and change my password.	#6.4 Test channel visibility functionality:							1
		#6.4.1 Perform unit and integration tests for the chatroom visibility.							1
		#6.4.2 Verify that user-user, admin-admin user-admin interaction is possible							1
		#6.4.3 Verify that user's can enter chat room at any time after login							1
		#6.4.4 Verify that user's refresh does not delete the receiving and sent messages							1
		#7.1 Design the UI of the Settings Page							1
		#7.1.1 Design a left panel containing the Account Settings and Messages							1
		#7.1.2 In the account settings tab, display the username, email							1
		#7.1.3In the account settings tab, add 2 input fields that take input for a new password and a i							1
		#7.1.4Add a back arrow that will be used to go to the previous page							1
		#7.2 Design the Backend Logic Behind the Settings Page	HARD (3)						1
		#7.2.1Implement the back button logic so that it redirects the user to the previous page							1
		#7.2.2Update the database with the new password when a new password is inputted and sav							1