**DETAILED COLLABORATION MODULE REQUIREMENTS DOCUMENT**

**1. Introduction and Purpose**

**1.1 Background**

The NIRDA Knowledge Management System is an existing platform designed to facilitate knowledge sharing and management within the organization. The system currently includes modules for document management, search functionality, analytics, user management, configuration and many more. These modules are accessible through the main navigation header, which has been developed and implemented.

**1.2 Purpose of This Assignment**

The purpose of this assignment is to develop and integrate the Collaboration Module, which is the final component of the NIRDA Knowledge Management System. This module aims to enhance team collaboration, communication, and project management within the organization.

**1.3 Scope**

The Collaboration Module consists of four main components:

1. Event Management
2. Discussion Forum
3. Collaborative Projects
4. Chat Management

**1.4 Developer Instructions**

As the developer assigned to this project, you are expected to:

1. Implement the Collaboration Module based on the requirements outlined in this document.
2. Maintain consistency with the existing system, particularly in terms of user interface and color scheme.
3. Present finished components to the supervisor at regular intervals during the one-month development period.
4. Deliver the fully integrated and tested module to the supervisor for evaluation at the end of the one-month period.

**1.5 Presentation Schedule**

* Week 1: Present the Event Management component
* Week 2: Present the Discussion Forum component
* Week 3: Present the Collaborative Projects component
* Week 4: Present the Chat Management component
* End of Month: Present the fully integrated Collaboration Module

**1.6 Technical Requirements and Developer Guidelines**

**1.6.1 Technology Stack**

The Collaboration Module should be developed using the following technologies:

* Backend: PHP (latest stable version)
* Database: MySQL
* Frontend: HTML5, CSS3, JavaScript (ES6+)
* Server: XAMPP

***Explanation:*** Using PHP and MySQL ensures compatibility with the existing NIRDA Knowledge Management System. XAMPP provides a convenient local development environment that includes PHP, MySQL, and Apache server.

**1.6.2 Development Environment**

* Text Editor/IDE: Use any preferred text editor or Integrated Development Environment (IDE) that supports PHP, HTML, CSS, and JavaScript development. Popular options include Visual Studio Code, PhpStorm, or Sublime Text and notepad++.

**1.6.3 Coding Standards**

* Follow PSR-12 coding standards for PHP code.
* Use ESLint for JavaScript code linting.
* Ensure all code is well-commented and follows best practices for readability and maintainability.

**1.6.4 Time Management**

* Create a detailed project timeline, breaking down each component into smaller, manageable tasks.
* Allocate time for research, learning, development, testing, and buffer for unforeseen challenges.
* Set daily and weekly goals to ensure steady progress.

**1.6.5 Continuous Learning**

* Conduct thorough research on any unfamiliar concepts or technologies before implementation.
* Utilize online resources such as documentation, tutorials, and video courses to enhance skills and knowledge.
* Allocate dedicated time for learning and experimentation with new technologies or frameworks that may be beneficial for the project.

**1.6.6 Performance and Security**

* Implement proper error handling and logging mechanisms.
* Use prepared statements for all database queries to prevent SQL injection.
* Implement CSRF protection for all forms.
* Ensure all user inputs are properly sanitized and validated.
* Use HTTPS for all communications (configure SSL on the XAMPP server for local development).

**1.6.7 Responsive Design**

* Ensure all new interfaces are responsive and work well on various devices and screen sizes.
* Use a mobile-first approach when designing and implementing the user interface.

**1.6.8 Testing**

* Write unit tests for all critical functions.
* Perform cross-browser testing to ensure compatibility with major browsers (Chrome, Firefox, Safari, Edge).
* Conduct thorough user acceptance testing (UAT) before the final presentation.

By following these guidelines and utilizing the specified technologies, you will ensure that the Collaboration Module integrates seamlessly with the existing system and meets the required quality standards. Remember to manage your time effectively, allocate sufficient time for learning and problem-solving, and maintain open communication with the supervisor throughout the development process. Regular check-ins and progress updates will help identify and address any issues early in the development cycle.

**2. General Design and Consistency Requirements**

**2.1 Color Scheme**

The developer must adhere to the existing color scheme of the NIRDA Knowledge Management System to maintain visual consistency. The primary colors used in the header are:

* Primary Color: #1a237e (deep blue)
* Secondary Color: #2c3e50 (dark grayish blue)
* Accent Color: #00A0DF (bright blue)
* Background Color: #f0f2f5 (light grayish blue)
* Text Color: #333333 (dark gray)

Ensure that all new interfaces and components within the Collaboration Module use this color scheme consistently.

**2.2 Typography**

Use the existing font family 'Roboto' for all text elements to maintain consistency with the rest of the system.

**2.3 Icon Usage**

Utilize Font Awesome icons (version 6.1.1) for all icons within the Collaboration Module, consistent with the existing header.

**2.4 Layout Consistency**

Maintain a similar layout structure to the existing modules, including consistent spacing, padding, and margin values.

**3. Detailed Component Requirements**

**3.1 Event Management**

**3.1.1 Schedule Event**

***Requirement:*** Users should be able to create new events with specific details.

***Explanation:*** Implement a form for event creation with the following fields:

* Event title: Text input for the name of the event
* Date and time: Date and time picker for selecting the event's start and end times
* Location: Text input for physical location or URL input for virtual events
* Description: Rich text editor for detailed event description
* Attendees: Multi-select dropdown or search field to invite users from the system
* Recurring options: Dropdown with options for daily, weekly, monthly, or yearly recurrence
* Reminders: Checkboxes for email and in-app notification reminders, with a dropdown for timing (e.g., 15 minutes before, 1 hour before, 1 day before)

***Implementation Notes:***

* Use AJAX for form submission to provide a smooth user experience
* Implement client-side validation for all fields
* Ensure the form is responsive and works well on both desktop and mobile devices

**3.1.2 Manage Events**

***Requirement:*** Users should be able to view and manage their events.

***Explanation:*** Create an event management dashboard with the following features:

* ***List View:*** Display all events in a paginated list, showing key information (title, date, time, location)
* ***Calendar View:*** Implement a monthly calendar view where events are displayed on their respective dates
* ***Event Actions:*** For each event, provide buttons or dropdown options for:
  + Edit: Opens the event creation form pre-filled with the event's current details
  + Delete: Shows a confirmation dialog before permanently removing the event
  + Deactivate: Temporarily hides the event from public view without deleting it
  + View Attendees: Opens a modal showing all invited attendees and their RSVP status
  + Send Reminders: Allows manual triggering of reminder notifications

***Implementation Notes:***

* Use a responsive data table for the list view
* Implement a JavaScript-based calendar library (e.g., FullCalendar) for the calendar view
* Use color-coding to distinguish between active and deactivated events
* Implement real-time updates for attendee RSVPs using WebSockets if possible

**3.2 Discussion Forum**

**3.2.1 Create Forum**

***Requirement:*** Allow users to create new forum categories.

***Explanation:*** Implement a forum creation interface with the following elements:

* Category Name: Text input for the forum category title
* Description: Text area for a brief description of the category's purpose
* Access Permissions: Radio buttons or dropdown for selecting forum visibility:
  + Public: Visible to all users
  + Private: Visible only to selected users or groups
  + Restricted: Visible to specific user groups (e.g. Units, divisions departments, roles)

***Implementation Notes:***

* If "Private" or "Restricted" is selected, display a multi-select field for choosing users or groups
* Implement a preview feature to show how the forum category will appear
* Use AJAX for form submission to provide instant feedback

**3.2.2 Manage Forums**

***Requirement:*** Implement comprehensive forum management functionalities.

***Explanation:*** Create a forum management interface with the following features:

* ***Topic Creation:*** Allow users to create new topics within categories
* ***Content Moderation:*** 
  + Implement an edit function for topics and replies
  + Add a delete option with confirmation for topics and replies
  + Include a "deactivate" toggle to hide topics without deleting them
  + Create a flagging system for users to report inappropriate content
  + Implement an admin interface for reviewing flagged content
* ***Topic Management:*** 
  + Allow admins to pin important topics to the top of the category
  + Implement a search function with filters (by date, user, content)
* ***Notification System:*** 
  + Create a subscription feature for users to follow topics
  + Implement in-app and email notifications for new replies and mentions

***Implementation Notes:***

* Use a threaded view for displaying topics and replies
* Implement a rich text editor (e.g., TinyMCE, CKEditor) for creating and editing posts
* Use infinite scrolling or pagination for long threads
* Implement real-time updates for new posts and replies using WebSockets if possible

**3.3 Collaborative Projects**

**3.3.1 Create Project**

***Requirement:*** Allow users to create new collaborative projects.

***Explanation:*** Develop a project creation interface with the following fields:

* Project Name: Text input for the project title
* Description: Rich text editor for detailed project description
* Start and End Dates: Date pickers for project timeline
* Project Members: Multi-select field to add users, with role assignment (e.g., manager, contributor, viewer)
* Goals and Objectives: Ability to add multiple goals with descriptions

***Implementation Notes:***

* Implement drag-and-drop functionality for easy member role assignment
* Use a wizard-style interface for project creation to break it into manageable steps
* Provide templates for common project types to speed up the creation process

**3.3.2 Manage Projects**

***Requirement:*** Implement comprehensive project management functionalities.

***Explanation:*** Create a project management dashboard with the following features:

* ***Project Overview:*** Display key project information, progress, and recent activities
* ***Task Management:*** 
  + Implement a Kanban board for visual task management
  + Allow creation, editing, and deletion of tasks
  + Include fields for task priority, deadline, and assignee
* ***File Sharing:*** 
  + Create a file repository for project-related documents
  + Implement version control for uploaded files
* ***Project Timeline:*** 
  + Develop a Gantt chart view for visualizing project timeline and dependencies
* ***Reporting:*** 
  + Generate project status reports with key metrics and progress indicators

***Implementation Notes:***

* Use a drag-and-drop library if possible (e.g., react-beautiful-dnd) for the Kanban board
* Implement a file upload system with progress indicators and preview capabilities
* Use a charting library (e.g., Highcharts) for the Gantt chart and report visualizations
* Implement real-time collaboration features using WebSockets for simultaneous editing

**3.3.3 Project Members Management**

***Requirement:*** Provide tools for managing project team members.

***Explanation:*** Develop a member management interface within each project with these features:

* Invite Members: Allow project managers to send invitations via email or in-app notifications
* Remove Members: Implement a function to remove members with a confirmation dialog
* Role Management:
  + Define clear roles (e.g., project manager, contributor, viewer) with specific permissions
  + Allow changing of member roles with proper authorization
* Activity Tracking: Display member activity logs within the project

***Implementation Notes:***

* Implement role-based access control (RBAC) to manage permissions effectively
* Use a modal interface for member management to avoid page reloads
* Provide bulk actions for efficient management of multiple members

**3.4 Chat Management**

**3.4.1 Contacts**

***Requirement:*** Display and manage user contacts.

***Explanation:*** Create a contacts management interface with these features:

* Contact List: Show all contacts with their online/offline status
* Add/Remove Contacts: Implement functionality to add new contacts and remove existing ones
* Block Contacts: Allow users to block and unblock contacts
* Grouping: Enable users to create custom groups for organizing contacts
* Search: Implement a search function to quickly find contacts

***Implementation Notes:***

* Use WebSockets to update online/offline status in real-time
* Implement lazy loading for large contact lists to improve performance
* Use caching mechanisms to store contact information locally for quick access

**3.4.2 Online Users**

***Requirement:*** Display currently online users and enable quick communication.

***Explanation:*** Develop an "Online Users" section with the following features:

* Online Users List: Display all currently online users
* Status Indicators: Show user status (available, busy, away) with appropriate icons
* Quick Chat: Allow users to initiate a chat directly from the online users list
* Last Seen: For offline users, show their last online time

***Implementation Notes:***

* Implement real-time status updates using WebSockets
* Use a presence system to accurately track user online status
* Ensure the online users list is updated efficiently to handle large numbers of users

**3.4.3 Chat Types**

***Requirement:*** Implement various types of chat functionalities.

***Explanation:*** Develop the following chat types:

* Private Chats: One-on-one messaging between users
* Group Chats: Multi-user conversations
* Starred Conversations: Allow users to mark important chats for quick access

***Implementation Notes:***

* Use a consistent UI across all chat types for a unified experience
* Implement end-to-end encryption for all chat communications
* Allow seamless switching between different chat types within the same interface

**3.4.4 Manage Chats**

***Requirement:*** Provide comprehensive chat management functionalities.

***Explanation:*** Implement the following features for chat management:

* Create Groups: Allow users to create new chat groups with custom names and images
* Edit Groups: Enable editing of group details and member lists
* Message Management:
  + Allow editing and deleting of sent messages
  + Implement message search functionality
* File Sharing: Enable sharing of files and media within chats
* Rich Media: Support emojis, GIFs, and inline media previews
* Chat Settings:
  + Implement read receipts and typing indicators
  + Allow users to mute conversations or set notification preferences

***Implementation Notes:***

* Use a robust real-time messaging system (e.g., Socket.io, Pusher) for instant message delivery
* Implement a caching mechanism for message history to reduce server load
* Use lazy loading for chat history to improve performance with long conversations
* Ensure all file sharing adheres to security best practices and includes virus scanning

**4. Integration and Testing**

**4.1 Integration with Existing System**

* Ensure seamless integration with the existing NIRDA Knowledge Management System
* Use consistent API patterns and data structures
* Implement proper error handling and logging

**4.2 Testing Requirements**

* Develop comprehensive unit tests for all new components
* Perform integration testing with existing system modules
* Conduct thorough user acceptance testing (UAT) with a representative user group
* Perform security testing, including penetration testing and vulnerability assessments

**4.3 Performance Benchmarks**

* Ensure page load times are under 3 seconds
* Optimize real-time features to handle at least 1000 concurrent users
* Aim for 99.9% uptime for all collaboration features

**5. Delivery and Presentation**

**5.1 Incremental Delivery**

* Deliver each component (Event Management, Discussion Forum, Collaborative Projects, Chat Management) as they are completed
* Provide a brief demo and documentation for each delivered component

**6.2 Final Presentation**

* Prepare a comprehensive presentation of the entire Collaboration Module
* Demonstrate the integration with the existing system
* Be prepared to discuss challenges faced and solutions implemented

Remember to maintain open communication with the supervisor throughout the development process, seeking clarification on any ambiguous requirements and providing regular updates on progress and any obstacles encountered.