Task-Management-System

Data Model and Relationships

At its core, the application uses a relational schema with three primary tables:

- Users – Stores user credentials and basic profile data. Each user can have multiple subscriptions.
- TaskItems – Contains details of individual tasks such as title, description, due date, status, and a computed priority value.
- TaskSubscriptions – Acts as the junction table enabling a many-to-many association: a user can subscribe to multiple tasks, and each task can have multiple users subscribed.

This model ensures that task tracking and subscription management are flexible enough to support scenarios where several users interact with the same task simultaneously.

Priority Calculation Service

A dedicated service calculates the priority of a task by merging multiple factors. Although the actual implementation file for the priority calculation is not explicitly provided among the files, the process integrates data from both TaskItems and Categories. The service adjusts the priority based on parameters such as the task's category priority (coming from the Categories entity), user-assigned priority values, and the due date, which contributes to a time sensitivity factor. The calculation often involves normalizing these values—dividing the category and user priority by their respective scales—and applying a logarithmic function based on hours remaining until the due date. This enables tasks nearing their deadline to receive a higher priority score. The resulting weighted number is then scaled to an integer value, which is used for sorting tasks in the user interface. Client-side code in taskhub.jsx demonstrates how tasks are sorted by priority, making the computed value visible and actionable in real time.

Session Handling and Overall Integration

Authentication and authorization are managed through a session mechanism that stores a DTO (including the user ID) both in local storage and in Recoil state. This unified session management guarantees that tasks and subscriptions are securely tied to the correct user. The modular design of the React components—such as the separation between task addition, category management, and dashboard views—ensures that each functionality is encapsulated and easily maintainable, while SignalR provides a robust framework for real-time interactions.

Real-Time Updates via SignalR

The project leverages SignalR on the backend to provide immediate updates across connected clients. When an action—such as task creation or status update—occurs, the server pushes updates instantaneously to all clients subscribed to that task. This real-time mechanism is crucial when multiple users are monitoring or modifying the same task, ensuring that the task status stays consistent throughout the user interfaces.

Task Priority Calculation Service

Though the actual implementation file for the priority calculation service isn't provided, the process is central to the overall business logic. The service calculates task priority based on three main factors:

- Category Priority: Each category is assigned a priority that acts as a weight.
- User-Assigned Priority: Users can select a priority level when creating or updating tasks.
- Due Date Sensitivity: The remaining time until the task's due date is factored in using a logarithmic function. This mechanism amplifies the urgency as deadlines approach.

By normalizing the category and user inputs and coupling these with a time-sensitive weight, the computed priority is scaled to an integer value (typically between 0–100). This priority directly influences task presentation on the dashboard, ensuring urgent tasks are prominently highlighted for all subscribed users.

Session Handling and Overall Integration

Authentication and authorization are managed through a session mechanism that stores a DTO (including the user ID) both in local storage and in Recoil state. This unified session management guarantees that tasks and subscriptions are securely tied to the correct user. The modular design of the React components—such as the separation between task addition, category management, and dashboard views—ensures that each functionality is encapsulated and easily maintainable, while SignalR provides a robust framework for real-time interactions.

In summary, this architecture combines a well-structured database schema, real-time updates via SignalR, modular React components for client interactions, and a dynamic task priority calculation service to create a responsive, user-centric application.

Client-Side Interfaces and Modular Components

Several React components form the client-side experience:

- AddCategoryModal (addcategorymodal.jsx)
  This modal component is designed for adding a new task category. It uses React hooks to manage its state (such as category name and default priority) and employs an API call to fetch and update the list of categories. When a new category is added, the UI list updates accordingly to reflect the change1.
- AddTaskModal (addtaskmodal.jsx)
  The modal for creating a new task gathers details like title, description, due date, user priority, and selected category. Notably, it retrieves the current user's information from Recoil state, which is initially set up during user authentication. This connection between Recoil and local storage (holding the DTO with the user ID) ensures that each task is properly associated with the authenticated user. The component also integrates with the backend to post new tasks, which later trigger SignalR notifications for real-time updates3.
- AddTaskPage (addtaskpage.jsx)
  This page component manages the display and behavior of the task creation modal. It controls navigation such that once the modal is closed, the user is redirected back to a dashboard view. This structure provides a smooth workflow from task creation to task dashboard management2.
- Dashboard (dashboard.jsx)
  Serving as the primary user interface, the dashboard component integrates both task and category management functionalities. It conditionally renders the task and category modals based on user interaction. Users can navigate through the dashboard to monitor tasks in real time, leveraging updates from SignalR, and manage categories—all coordinated through this central hub