**Approach to the Task Management System**

In developing our task management system, we focused on creating a user-friendly and efficient platform that allows users to manage tasks easily. The system provides features such as user registration and authentication, task creation and management, task assignment, filtering, and searching tasks. Here's a summary of our approach:

**1. System Architecture:**

- We used Spring Boot as the primary framework for building the back-end.

-MySQL was chosen as the relational database to store all the data related to tasks, users, and roles.

- We implemented Spring Security for authentication and authorization, ensuring that users have role-based access (Admin or User).

**2. Key Features:**

- **User Authentication & Registration**: Users can register with the system and log in using secure credentials. We implemented role-based access control so that admins and users have different permissions.

- **Task Management:** Users can create, update, delete, and assign tasks. Admins have broader control, such as viewing all users' tasks, while users can manage only their tasks.

- **Filtering and Searching:** Tasks can be filtered based on criteria like status (e.g., pending, completed) and due dates. Users can search for specific tasks quickly.

**3. Dockerization:**

- We containerized the application using Docker. This ensures that the application can run consistently across different environments.

- We used a `Dockerfile` to define the application container and a `docker-compose.yml` file to configure both the application and the MySQL database, allowing seamless setup and deployment.

**4. Testing and CI/CD Pipeline:**

- **Unit and Integration Testing:** We created test cases using JUnit and Mockito to verify the functionality of individual components as well as the integration between them.

- **CI/CD Pipeline:** Using GitHub Actions, we set up a continuous integration pipeline. This pipeline automatically runs tests and builds the application whenever new code is pushed to the repository.

**Assumptions Made:**

**1. Role-Based Access Control:** We assumed that the system requires two distinct user roles: Admin and User. Admins have more control over tasks and user management, while regular users manage their own tasks.

**2. Task Assignment:** It was assumed that only admins can assign tasks to users, while users can create and manage tasks for themselves.

**3. Authentication:** We assumed that all users must register and log in before using the system. This ensures that task data is secure and only accessible by authorized individuals.

**4. Docker Environment:** We assumed that Docker would be available in the deployment environment to run the containerized application and MySQL database.

This approach ensures that the task management system is scalable, secure, and easy to maintain, with a strong focus on efficient task handling and user access management.