Hey,

It was nice talking to you and as per our conversation

here is a task for you, you will be building your own Backend in Node.js/Nest.js and you can use React.js or Next.js for Frontend

Please go through all the test instructions and let us know if you have any questions.

Test Submission Deadline: Please email me back till when you'll be able to submit this test.

Overview

The To-Do List app allows a user to create, read, update or delete to-do tasks.

A to-do task:

1. has a title

2. has a description

3. has a state of either completed or active

In the given exercise you are expected to build a To-Do list app as described in the document.

All requirements are mandatory except for bonus points.

Bonus Points are there to help you stand out from other applicants.

We will recommend you finish all mandatory tasks before attempting any bonus points.

User Interface

1. Login Screen

Has a login form with fields "Username" and "Password" and a "Submit" button.

Requirements:

1. If the user inputs correct credentials and submit, they are redirected to the "Tasks" screen.

2. If the user inputs incorrect credentials and submit, they stay on the "Login" screen and the error message is displayed on the screen.

Bonus Points: Form Validations: valid email and password. A valid password is at least 6 chars long, has at least one uppercase and one lowercase character.

2. List Tasks Screen

Has a list of all the tasks created by the user, a search bar to search for specific tasks, and an action menu.

Tasks can either be marked as completed or active.

Requirements:(Task List):

1. Each list item is referred to as " tile".

2. Each tile displays the title of the task.

3. Each tile has a checkbox-like element to represent the task's completion state.

i) Unchecked for the active state.

ii) Checked for the completed state.

4. The user can click the checkbox-like element to toggle the task's completion state.

5. When a user clicks on the title of a task, they are redirected to the "Details" screen.

Requirements:(Search Bar):

1. The user can search for a task by typing the task title or excerpt from the task description.

Requirements:(Action Menu):

1. The action menu has an "Add Task" button which redirects the user to the "Add Task" screen.

2. The action menu has a component to filter tasks listed in the list of tasks.

3. The filter component has three filters:

i) All - no filters; display all tasks.

ii) Active - display active tasks.

iii) Completed - display completed tasks.

4. The action menu has an action button to bulk remove all completed tasks from the list with a single click.

Bonus Points: Ability to undo the delete operation.

Bonus Points: Implement pagination and sorting.

Bonus Points: An action button to sync the application store data with the app's backend.

3. Details Screen

Displays a task's details.

Requirements:

1. Display the title of the task.

2. Display the description of the task.

3. An action button to delete the task.

on click, the task is deleted - it's removed from the list of tasks and the user is redirected to the "Tasks" screen.

4. An action button to edit the task.

on click - the user is redirected to the "Edit Task" screen.

Bonus Points: Ability to undo the delete operation.

4. Add Task Screen

Allow the user to create a task.

Requirements:

1. A text field for task title.

2. A text area for task description.

3. An action button to create a task.

On click - creates a new task and the user is redirected to the "List Tasks" screen.

Bonus Points: Form Validations: i) A valid title has a max length of 50 chars. ii) A valid description has a max length of 256 chars.

Bonus Points: Data sanitization before making an API request.

5. Edit Task Screen

Allow the user to edit a task.

Requirements:

1. A text field, pre-populated with the task title.

the text field can be edited to mutate its content.

2. A text area, pre-populated with the task description

the text area can be edited to mutate its content.

3. An action button to save changes.

On click - the task is updated with the new changes and the user is redirected to the "Details" screen.

Bonus Points: Form Validations: i) A valid title has a max length of 50 chars. ii) A valid description has a max length of 120 chars.

Bonus Points: Data sanitization before making an API request.

Cheatsheet:

Applicants are allowed to ignore the quality of UI/UX. You won't be evaluated against it although a good user experience goes a long way.

Further instructions related to API, focus on backend and Api integration with best practices of backend development

Evaluation Criteria (for evaluators)

The applicants will be evaluated according to and in descending order of credits of the following:

1. Accuracy of implementation of features against the provided information in the document.

2. Code quality.

3. Performant code.

4. Best practices like logic splitting and reusing common components, project organization, etc.

5. Implementation patterns in state management.

6. Integration of API Layer.

Bonus Points: Usage of functional components in relevant cases.

if you have any doubt related to task you can contact to me

Thank you

Sandeep