

タスクの作成/編集を実装							
達成要件		実装の例					
<p>■「実装の例」を参考に、タスクを追加するモーダルを完成させてください。また、タスクの propsがある場合には編集になるよう実装してください。</p> <p>■ 編集の場合は、Taskコンポーネントのクラス名が task_text_contentsの要素をクリックすると、編集用のモーダルが開くようにしてください。その時に、選択したタスクの情報を propsでモーダルへ渡してください。</p>		<pre>import React, { useContext, useState } from "react"; import { Select } from "../../components/select"; import { DataContext } from "../../pages/home"; import { TaskType } from "../../interfaces/Task"; import { taskRequest } from "../../requests/taskRequest"; import "./style.css";  interface Props {   // 編集の場合は、タスクを受け取る   task?: TaskType;   handleClose: () =&gt; void; }  export const TaskBody = (props: Props) =&gt; {   const date: Date = new Date();    const { data, dispatch } = useContext(DataContext);    // name, genreId, explanation, deadlineDateをstateで管理する    const onClickSubmit = async () =&gt; {     const requestData = {       name: name,       genre_id: genreId,       explanation: explanation,       deadline_date: deadlineDate,       status: {props.task &amp;&amp; props.task.status}    0,     };      if (props.task !== undefined) {       try {         // props.taskがundefinedではない場合、すなわち編集の場合は、requestDataをパラメーターとして、編集するリクエストを行う       } catch (err) {         dispatch({ type: "failure", payload: { error: err.message } });       }     } else {       try {         // requestDataをパラメータとして、新規作成するリクエストを行う       } catch (err) {         dispatch({ type: "failure", payload: { error: err.message } });       }     }     props.handleClose();   }; };</pre> <pre>// inputなどで変更された値を、stateへセットする関数を定義する const handleChangeGenre = (event: any) =&gt; {}; const handleChangeTitle = (event: any) =&gt; {}; const handleChangeExplanation = (event: any) =&gt; {}; const handleChangeDeadlineDate = (event: any) =&gt; {};  return (   &lt;form className="flex direction_column horizontal_center vertical_center"&gt;     &lt;h2 className="input_menu"&gt;タスクを追加&lt;/h2&gt;     &lt;div&gt;       &lt;h4 className="input_title"&gt;ジャンル&lt;/h4&gt;       &lt;div className="task_genre"&gt;         &lt;Select           genres={data.genreData}           selectList={handleChangeGenre}           initialValue={genreId}         /&gt;       &lt;/div&gt;       &lt;h4 className="input_title"&gt;タイトル&lt;/h4&gt;       &lt;input type="text" value={name} onChange={handleChangeTitle} /&gt;       &lt;h4 className="input_title"&gt;説明&lt;/h4&gt;       &lt;textarea value={explanation} onChange={handleChangeExplanation} /&gt;       &lt;h4 className="input_title"&gt;期限&lt;/h4&gt;       &lt;input         className="input_date"         type="date"         value={deadlineDate}         onChange={handleChangeDeadlineDate}       /&gt;     &lt;/div&gt;     &lt;input       className="input_submit"       type="button"       value="送信"       onClick={onClickSubmit}     /&gt;   &lt;/form&gt; ); };</pre>					