Testkonzept

**Testkonzept für die Todo-Anwendung**

**1. Akzeptanzkriterien und Testfälle**

**1.1 Der Benutzer kann beim Erstellen einer Aufgabe eine Priorität auswählen (hoch, mittel, niedrig).**

* **Komponente:** InputTodo
* **Testfälle:**
  1. **Initialisierung der Eingabe:**
     + **Ziel:** Sicherstellen, dass das Eingabefeld und die Prioritätsauswahl korrekt angezeigt werden.
     + **Schritte:**
       1. Rendere die InputTodo-Komponente.
       2. Überprüfe, ob das Textfeld und die Dropdown-Liste sichtbar sind.
     + **Erwartung:** Beide Elemente sind sichtbar, die Standardpriorität ist „mittel“.
  2. **Erstellen einer Aufgabe:**
     + **Ziel:** Verifizieren, dass die Aufgabe mit einer ausgewählten Priorität erstellt wird.
     + **Schritte:**
       1. Gib einen Titel ein und wähle eine Priorität (z. B. „hoch“) aus.
       2. Klicke auf die Schaltfläche „Hinzufügen“.
     + **Erwartung:** Die addTodoProps-Funktion wird mit dem Titel und der Priorität aufgerufen.
  3. **Fehlerbehandlung:**
     + **Ziel:** Sicherstellen, dass keine Aufgabe ohne Titel erstellt wird.
     + **Schritte:**
       1. Lasse das Textfeld leer.
       2. Klicke auf „Hinzufügen“.
     + **Erwartung:** Eine Fehlermeldung wird angezeigt.

**1.2 Die Aufgabenliste zeigt die Priorität jeder Aufgabe an.**

* **Komponenten:** TodosList, TodoItem
* **Testfälle:**
  1. **Anzeige der Priorität:**
     + **Ziel:** Sicherstellen, dass jede Aufgabe mit der korrekten Priorität angezeigt wird.
     + **Schritte:**
       1. Übergib eine Liste von Aufgaben mit unterschiedlichen Prioritäten an TodosList.
       2. Überprüfe die value-Eigenschaft der Prioritätsfelder in den gerenderten TodoItem-Komponenten.
     + **Erwartung:** Die Priorität jeder Aufgabe wird korrekt dargestellt.

**1.3 Der Benutzer kann die Priorität einer bestehenden Aufgabe ändern.**

* **Komponenten:** TodoItem, TodoContainer
* **Testfälle:**
  1. **Änderung der Priorität:**
     + **Ziel:** Sicherstellen, dass die Priorität einer Aufgabe geändert werden kann.
     + **Schritte:**
       1. Wähle eine neue Priorität in einem TodoItem.
       2. Überprüfe, ob die changePriority-Funktion mit der korrekten ID und Priorität aufgerufen wird.
     + **Erwartung:** Die Funktion wird korrekt aufgerufen und der Zustand in TodoContainer wird aktualisiert.
  2. **Validierung der Eingabe:**
     + **Ziel:** Verifizieren, dass nur gültige Prioritätswerte akzeptiert werden.
     + **Schritte:**
       1. Gib einen ungültigen Wert (z. B. „abc“) in das Prioritätsfeld ein.
       2. Überprüfe, ob die Eingabe abgelehnt oder korrigiert wird.
     + **Erwartung:** Ungültige Eingaben werden ignoriert.

**1.4 Die Aufgaben können nach Priorität sortiert werden, sodass "hohe" Priorität ganz oben erscheint.**

* **Komponenten:** TodoContainer, TodosList
* **Testfälle:**
  1. **Sortierung der Aufgaben:**
     + **Ziel:** Sicherstellen, dass die Aufgabenliste korrekt sortiert wird.
     + **Schritte:**
       1. Übergib eine unsortierte Liste von Aufgaben an TodosList.
       2. Überprüfe die Reihenfolge der gerenderten Aufgaben basierend auf ihrer Priorität.
     + **Erwartung:** Aufgaben mit der Priorität „hoch“ erscheinen oben, gefolgt von „mittel“ und „niedrig“.

**3. End-to-End-Tests**

1. **Erstellen, Anzeigen und Ändern einer Aufgabe:**
   * **Ziel:** Überprüfen, dass alle Funktionen von der Erstellung bis zur Anzeige und Bearbeitung korrekt funktionieren.
   * **Schritte:**
     1. Erstelle eine Aufgabe mit einer bestimmten Priorität.
     2. Überprüfe, ob die Aufgabe in der Liste angezeigt wird.
     3. Ändere die Priorität und überprüfe die Aktualisierung.
   * **Erwartung:** Alle Aktionen funktionieren nahtlos.
2. **Sortierlogik testen:**
   * **Ziel:** Sicherstellen, dass die Sortierung basierend auf der Priorität korrekt durchgeführt wird.
   * **Schritte:**
     1. Erstelle mehrere Aufgaben mit unterschiedlichen Prioritäten.
     2. Überprüfe die Reihenfolge der Aufgaben in der Liste.
   * **Erwartung:** Aufgaben mit höherer Priorität erscheinen oben in der Liste.

**Testtools und Methodik**

* **Tools:**
  + **Unit-Testing:** Jest, React Testing Library
  + **End-to-End:** Cypress
  + **Komponenten:**

Mit diesem Konzept können die bereitgestellten Akzeptanzkriterien vollständig getestet und alle Anforderungen verifiziert werden.