Contents

[Install steps 1](#_Toc197437756)

[Architecture 3](#_Toc197437757)

[Client 3](#_Toc197437758)

[1. App.tsx (The Component) 3](#_Toc197437759)

[2. Service (todoService.ts) 3](#_Toc197437760)

[3. Hooks (useTodos.ts) 4](#_Toc197437761)

[4. Model (Todo.ts) 4](#_Toc197437762)

[Server 4](#_Toc197437763)

[Design 5](#_Toc197437764)

[Part 1 6](#_Toc197437765)

[1.a infinite loop bug 6](#_Toc197437766)

[1.b key prop warning 6](#_Toc197437767)

[1.c show content as well 7](#_Toc197437768)

[1.d add status sorting 8](#_Toc197437769)

[1.e add status update functionality 8](#_Toc197437770)

[1.f Add date sorting 9](#_Toc197437771)

[Part 2 11](#_Toc197437772)

[2.a. Pagination 11](#_Toc197437773)

[2.b. Round-trip optimization 11](#_Toc197437774)

[2.c Type filter 12](#_Toc197437775)

[2.d. Local storage 13](#_Toc197437776)

[Config Section 13](#_Toc197437777)

[Bonus Part 13](#_Toc197437778)

[Unit Tests 14](#_Toc197437779)

[Client 14](#_Toc197437780)

# Install steps

**npm install**

PS C:\PERS\Adrian\WORK\TestProjects\git\github\TheLotter\_AdrianMihailescu> npm install

npm error Conflicting peer dependency: react@16.14.0

npm error node\_modules/react

npm error peer react@"^15.3.0 || ^16.0.0" from material-ui-icons@1.0.0-beta.36

npm error node\_modules/material-ui-icons

npm error material-ui-icons@"^1.0.0-beta.36" from the root project

**Fixed**

npm uninstall material-ui-icons

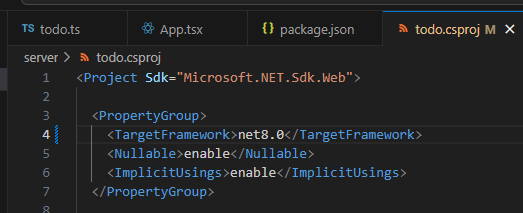
npm install

npm run dev failed. Runned:

npm install concurrently –save-dev

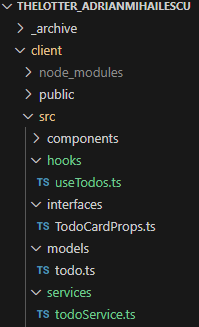
npm install react-scripts –save

**changed target to .net8**

****

# Architecture

## Client



### 1. ****App.tsx (The Component)****

* **Purpose**: This is the main component where the app runs. It's responsible for rendering the UI and handling user interaction (like changing filters or pages).
* **How it works**:
  + **State Management**: It keeps track of important states such as:
    - todos: The list of all todo items.
    - sortOption: The selected sorting method (e.g., active, done).
    - typeFilter: The selected filter (e.g., all, work, personal).
    - isDateAsc: Whether the todos should be sorted in ascending or descending order by date.
    - currentPage: The current page number for pagination.
  + **Custom Hook**: It triggers the fetching of todos whenever the typeFilter changes (this happens inside the useEffect hook).
  + **Rendering**: The todos are passed to child components like TodoCard (for displaying individual todos) and FilterPanel (for displaying filter options).

### 2. ****Service (todoService.ts)****

* **Purpose**: This file has functions that communicates with the backend server to fetch or update data (in this case, todo items).
* **How it works**:
  + **fetchTodos**: This function makes a request to the API to fetch the list of todos. It uses the typeFilter (like "All", "Active", "Completed") to get only the relevant todos.
  + If the fetch is successful, it returns the list of todos; if not, it throws an error.
  + **Communication**: The service functions are called inside the hook to get the latest data for the UI.

### 3. ****Hooks (useTodos.ts)****

* **Purpose**: This is a custom hook that contains the logic for fetching, filtering, sorting, and paginating the todos.
* **How it works**:
  + **State Handling**: It manages the todos state (the list of todos).
  + **Fetching**: It calls fetchTodos (from the service) when the typeFilter changes to fetch the updated list of todos.
  + **Filtering and Sorting**: Once the todos are fetched, the hook:
    - Filters them based on the selected sortOption (e.g., Active/Completed).
    - Sorts them based on the isDateAsc value (ascending or descending by date).
  + **Pagination**: It also handles pagination logic by slicing the filtered and sorted todos based on the current page.
  + The hook returns the filteredSortedTodos, paginatedTodos, and other values needed in the App.tsx component.

### 4. ****Model (Todo.ts)****

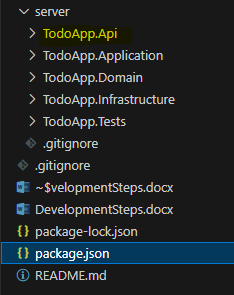
* **Purpose**: This file defines the structure of a todo item, so TypeScript knows how to treat the data.
* **How it works**:
  + The Todo model defines the properties of a todo item, such as:
    - id: A unique identifier for the todo.
    - title: The title of the todo.
    - status: The status (e.g., Active or Done).
    - dueDate: The date the todo is due.
    - creationTime: The date the todo was created.
  + **Usage**: This model ensures that each todo item is structured correctly and helps with type checking in the code.

## Server

Todo.csproj (main api)

Presentation Layer (Controllers, Program.cs)

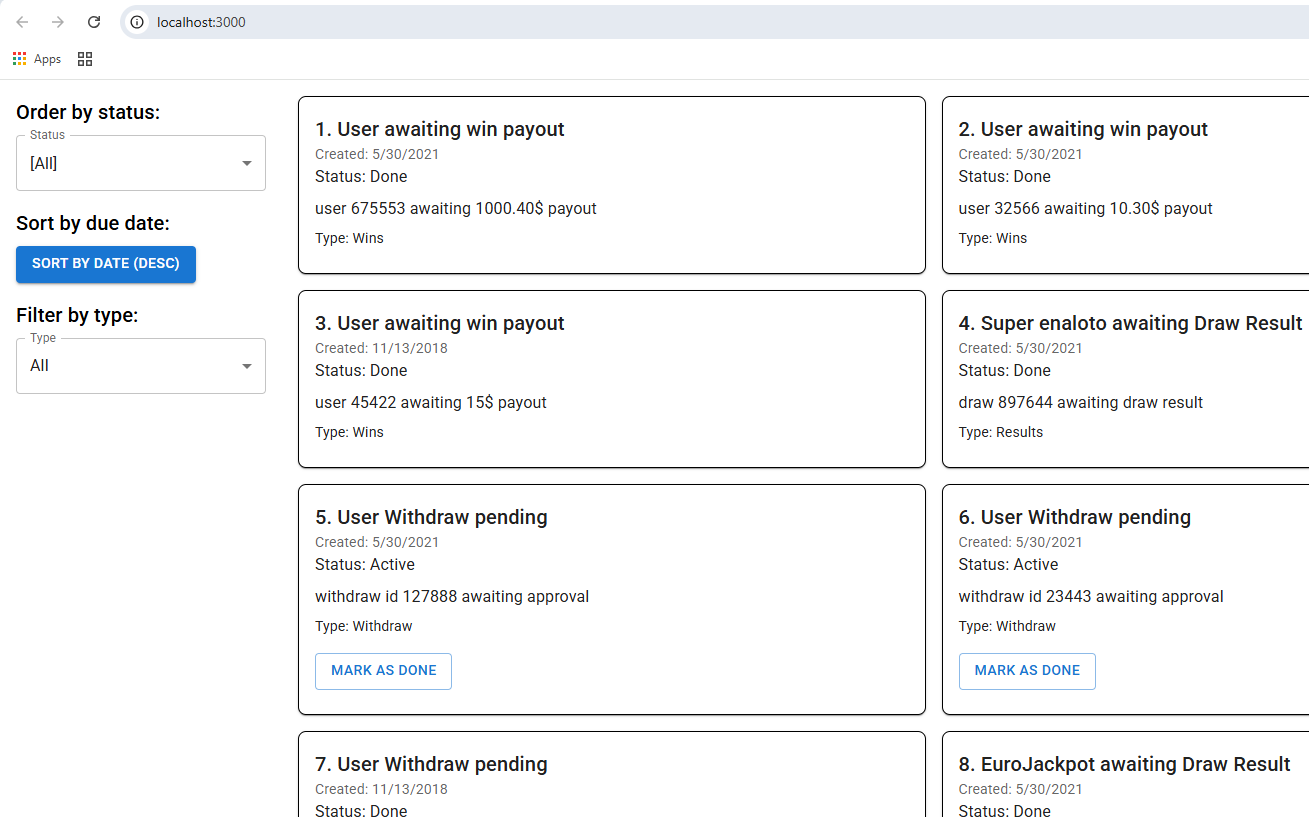
* TodoApp.Application/ → Business Logic Layer (Services, Interfaces)
* TodoApp.Domain/ → Domain Layer (Models, Entities)
* TodoApp.Infrastructure/→ Data Access Layer (Repositories, File IO, Caching)
* TodoApp.Tests/ → Unit Tests (xUnit/Moq)



Added references between projects

* TodoApp.Api will reference TodoApp.Application and TodoApp.Infrastructure
* TodoApp.Application will reference TodoApp.Domain
* TodoApp.Infrastructure will reference TodoApp.Domain
* TodoApp.Testw will reference TodoApp.Api and TodoApp.Application

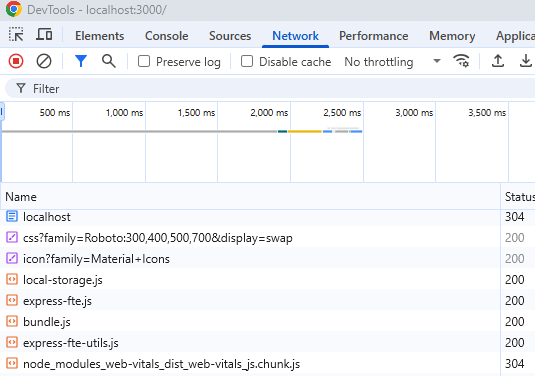
# Design



# Part 1

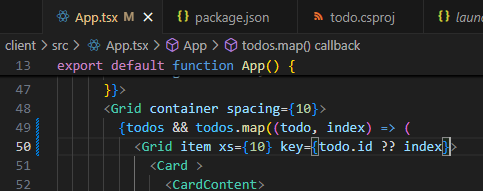
## 1.a infinite loop bug

* useEffect with an empty dependency array ([]) will only run once when the component mounts, instead of re-running on every render.
* Without the dependency array, the effect runs on every render, causing an infinite loop because setTodos triggers a re-render, which calls the effect again.



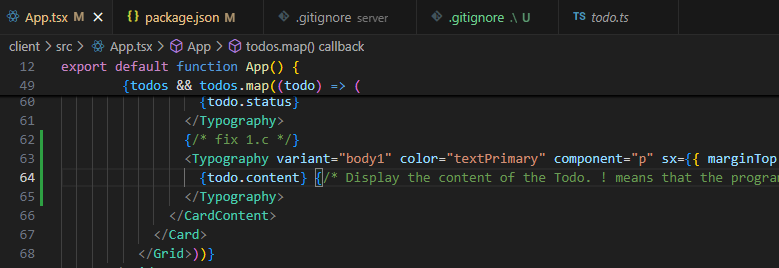
## 1.b key prop warning

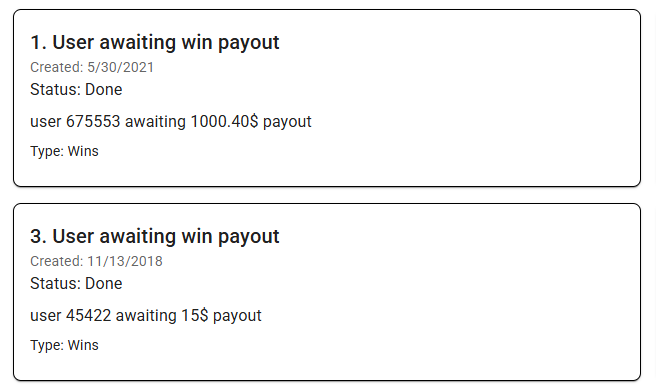
Added a key to each grid item.



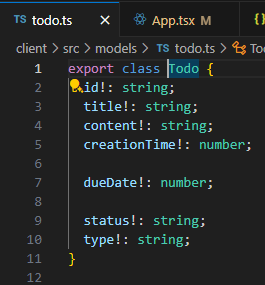


## 1.c show content as well

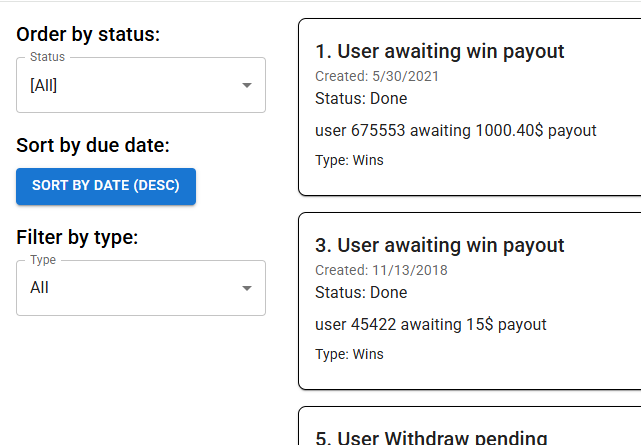




Taken from



## 1.d add status sorting

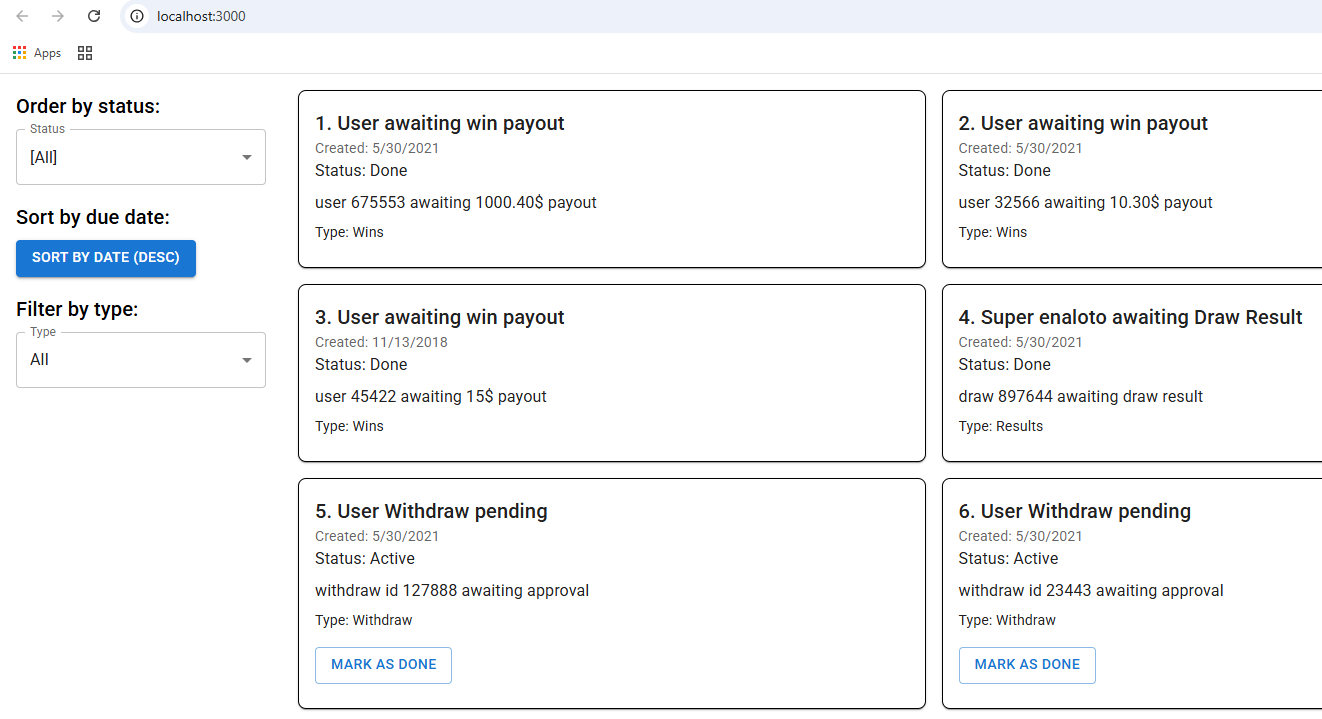


## 1.e add status update functionality

Add a "Mark as Done" button for TODOs that are still "Active".

Create a handleStatusUpdate function to call the API and update status.

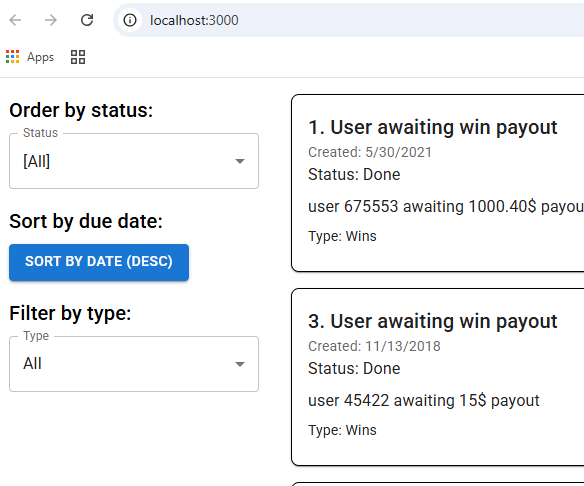
Re-fetch or locally update the todos state to reflect the change and re-sort.



## 1.f Add date sorting

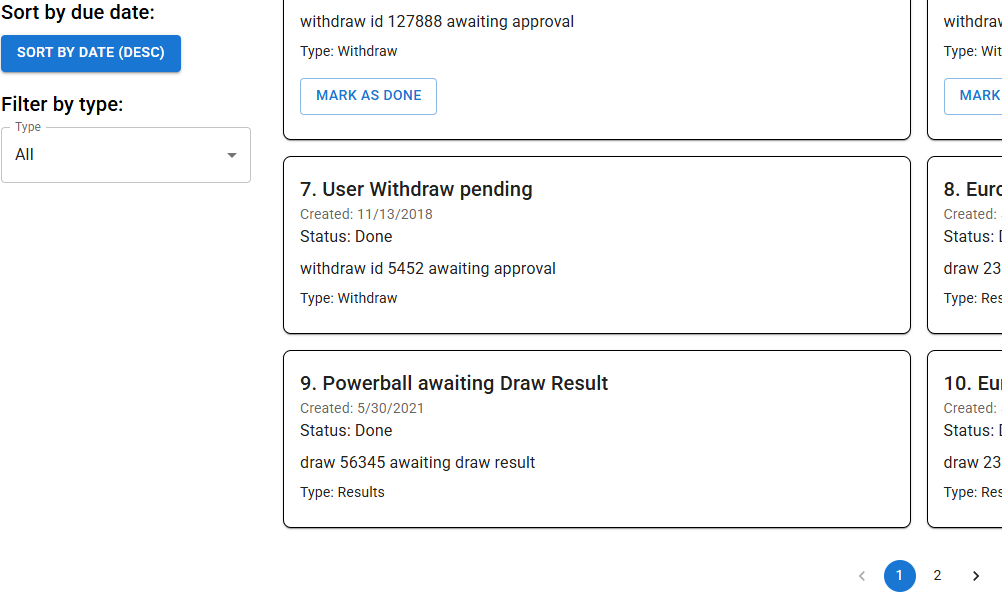
 Add a **"Sort by date"** button.

Toggle between **ascending and descending** order.

Reflect the sort in the list of TODOs.  


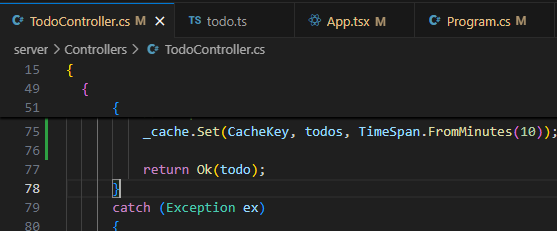
# Part 2

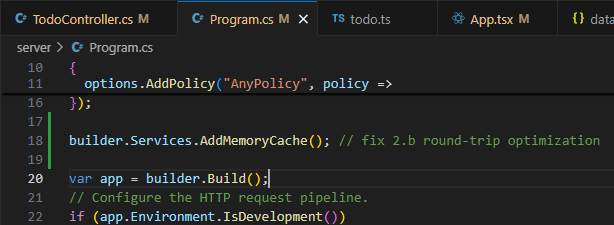
## 2.a. Pagination



## 2.b. Round-trip optimization

Builder add Memory cache

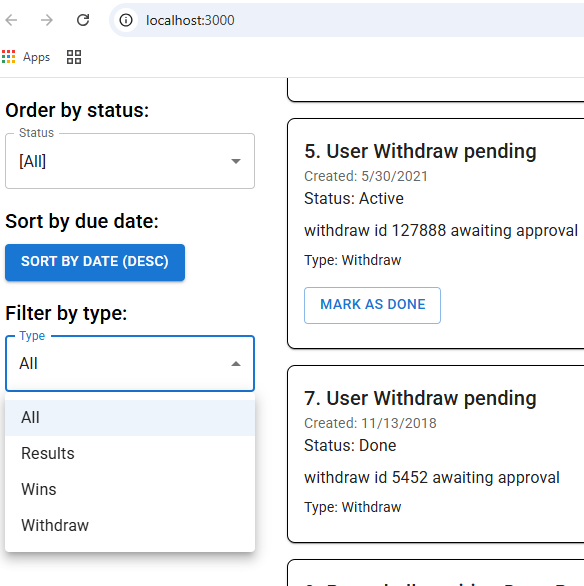




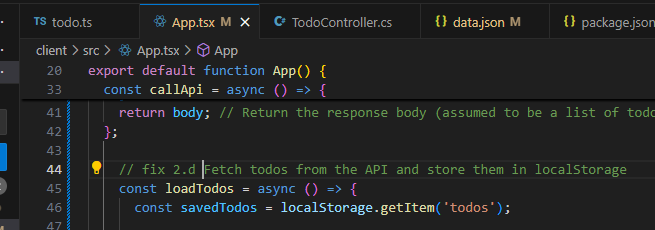
## 2.c Type filter

Backend changes

Frontend changes



## 2.d. Local storage

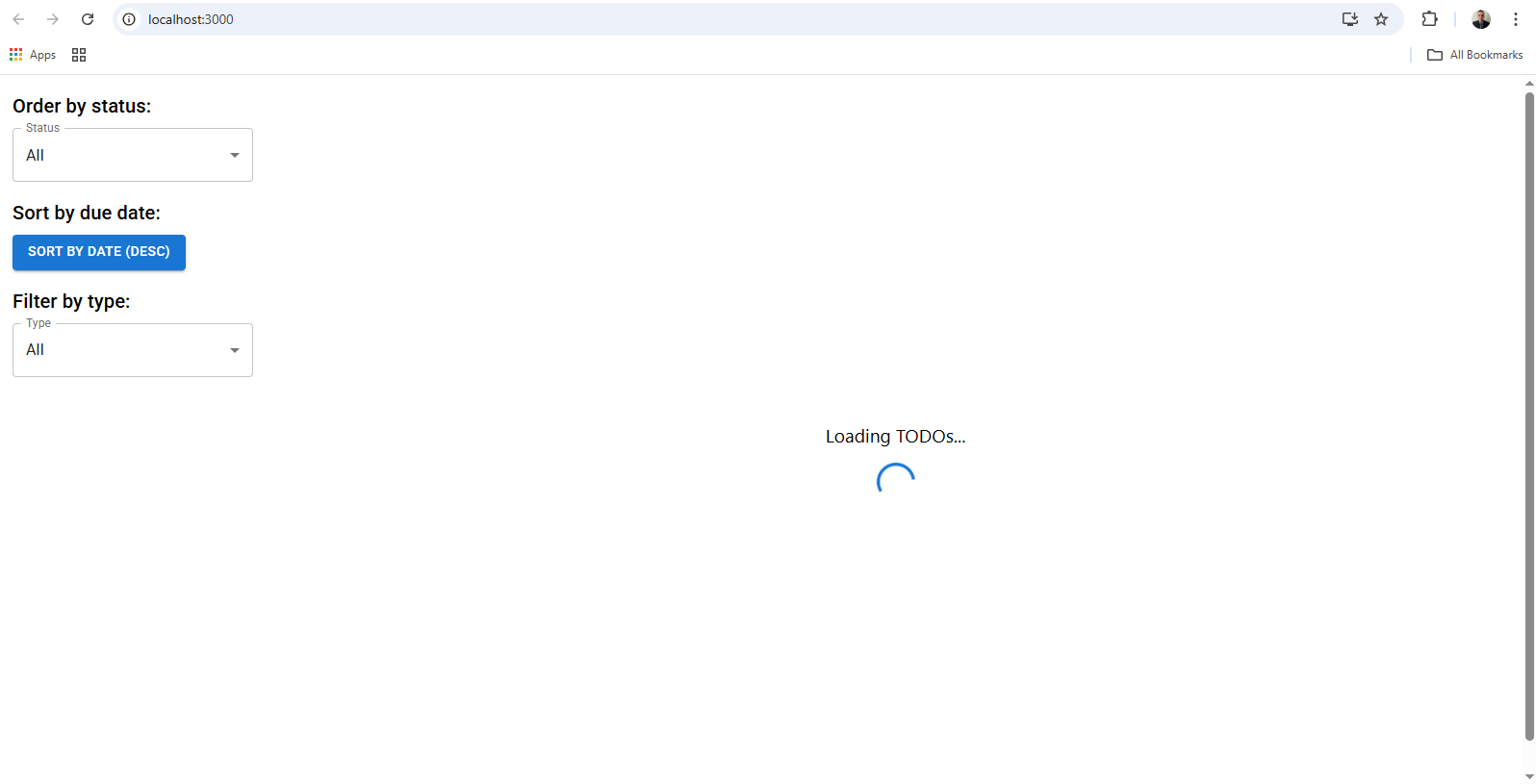


# Config Section

# 

# Bonus Part

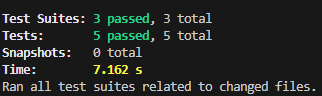
Progress bar for the long-running tasks

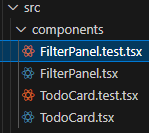


# Unit Tests

## Client

npm install --save-dev @testing-library/react @testing-library/user-event @testing-library/jest-dom jest



Server

