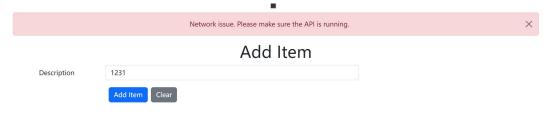
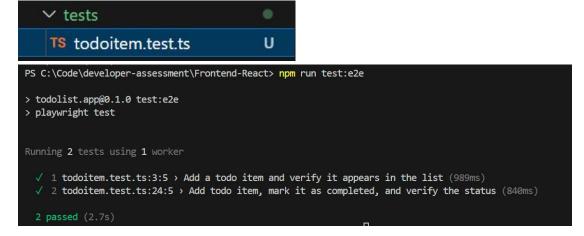
## **Frontend**

- Based on React and use TypeScript.
- Can add and mark todo items as completed;
- Created two components for better maintainability; One for adding todo item, another for displaying todo items.
- Added a `<Spinner/>` and a state to the `Add Item` button for debouncing.
- All the error message will be displayed in the alert.



- Created a generic `apiClient.ts` and `HttpService.ts` to make calling APIs easier and reusable.
- The base API URL defined in `.env`
- Created two end to end tests using Playwright; Use command `npm run test:e2e` to run the tests. More related commands defined in `package.json`.



## **Backend**

- Applied the so-called "Clean Architecture".
- Used `Mapster` for object mapping.
- Used `FluentValidation` for business logic validation.
- Created `AuditLoggingMiddleware.cs` for logging API Request and Response.
- Created unit tests for the controller only, as do not want to go too crazy on unit tests.

i Cat	Duration	Truits	LII
▲ Ø TodoList.Api.UnitTests (8)	150 ms		
■ TodoList.Api.Tests.Controllers (8)	150 ms		
■ TodoltemsControllerTests (8)	150 ms		
GetTodoltem_ReturnsNotFo	76 ms		
GetTodoltem_ReturnsOkRes	52 ms		
GetTodoltems_ReturnsOkRes	. 12 ms		
PostTodoItem_ReturnsBadRe	1 ms		
PostTodoltem_ReturnsCreate	. < 1 ms		
PutTodoItem_ReturnsBadRe	< 1 ms		
PutTodoItem_ReturnsBadRe	9 ms		
PutTodoltem_ReturnsNoCon	< 1 ms		