통합 테스트란 무엇인가?

실무에 바로 적용하는 프런트엔드 테스트

이재성

단위 테스트의 한계

독립된 일부 모듈만을 대상으로 검증

여러 모듈이 조합되었을 때의 동작은 검증할 수 없음

통합 테스트는

✔ 두 개 이상의 모듈이 상호 작용하여 발생하는 상태를 검증

✔ 실제 앱의 비즈니스 로직과 가깝게 기능을 검증할 수 있다

통합 테스트 항목

✔ 특정 상태를 기준으로 동작하는 컴포넌트 조합

✔ API와 함께 상호작용 하는 컴포넌트 조합

단순 UI 렌더링 및 간단한 로직을 실행하는 컴포넌트까지

한번에 효율적으로 검증 가능

ProductCard(단위)

✔ prop 기준으로
가격·상품명이 제대로 렌더링 되는지

✔ 상품을 클릭 했을 때 navigate
모킹을 통해 상세화면으로 이동하는지

✔ 장바구니, 구매 버튼을 눌렀을 때
spy 함수를 통해 각 핸들러가 호출되는지

ProductList(통합)

✔ 상품 리스트 조회 API에 맞게
가격·상품명이 제대로 렌더링되는지

✔ 상품을 클릭 했을 때 navigate
모킹을 통해 상세화면으로 이동하는지

✔ 장바구니·구매 버튼을 눌렀을 때
→ 로그인: 상품 추가 후 장바구니로 이동
→ 비로그인: 로그인 페이지로 이동

✔ 상품 리스트가 더 있는 경우 show more
버튼이 노출되며, 이를 통해 데이터를 더 가져
올 수 있는지

ProductCard(단위)

✔ prop 기준으로
가격·상품명이 제대로 렌더링 되는지

✔ 상품을 클릭 했을 때 navigate
모킹을 통해 상세화면으로 이동하는지

✔ 장바구니, 구매 버튼을 눌렀을 때
spy 함수를 통해 각 핸들러가 호출되는지

ProductList(통합)

✔ 상품 리스트 조회 API에 맞게
가격·상품명이 제대로 렌더링되는지

✔ 상품을 클릭 했을 때 navigate
모킹을 통해 상세화면으로 이동하는지

✔ 장바구니·구매 버튼을 눌렀을 때
→ 로그인: 상품 추가 후 장바구니로 이동
→ 비로그인: 로그인 페이지로 이동

✔ 상품 리스트가 더 있는 경우 show more
버튼이 노출되며, 이를 통해 데이터를 더 가져
올 수 있는지

우리들이 작성할 통합 테스트는..

상태나 데이터를 관리하는 특정 컴포넌트를 기준으로
하위 컴포넌트가 제대로 렌더링 되는지 검증하는 테스트

우리들이 작성할 통합 테스트는..

상태나 데이터를 관리하는 특정 컴포넌트를 기준으로
하위 컴포넌트가 제대로 렌더링 되는지 검증하는 테스트

데이터를 관리하는 로직이 산재 → 통합 테스트 작성이 어려움

앱의 상태를 어디서 어떻게 관리하고 변경할지

구조적인 설계가 중요함

우리들이 작성할 통합 테스트는..

즉, 통합 테스트를 잘 작성하기 위해서는
좋은 설계가 기반이 되어야 한다

정리

통합 테스트의 장점

- 여러 개의 모듈이 동시에 상호 작용 하는 것을 테스트하기 때문에
  단위 테스트에 비해 모킹의 비중이 적으며, 모듈 간에 발생하는 에러를 검증할 수 있다
- 실제 앱이 동작하는 비즈니스 로직에 가깝게 기능을 검증할 수 있다
- 하위 모듈의 단위 테스트에서 검증할 수 있는 부분까지 한 번에 효율적으로
  검증할 수 있다

결국 중요한 건

- 컴포넌트의 상태 및 데이터를 어디서 관리하고 변경할 지 구조적인 설계가 잘 되어야 함
- 좋은 설계를 위한 매개체가 된다!