Testes de Integração

Técnico em Desenvolvimento de Sistemas

Prof. Daniel Ferreira

Senac Fecomércio Sesc

Ferramentas para Testes de Integração em Node.js

Supertest: Testando Aplicações Web

O Supertest é uma ferramenta essencial para realizar testes de integração em aplicações web construídas com Node.js. Ele permite simular requisições HTTP e testar rotas e middleware de forma eficiente, garantindo o correto funcionamento da sua aplicação.

Jest: Estruturando os Testes

Ao combinar o Supertest com o framework de testes Jest, você pode criar uma estrutura robusta para seus testes de integração. O Jest fornece recursos avançados, como mocks, spies e snapshots, facilitando a escrita e manutenção dos seus testes.

Integrando com Express

Muitas aplicações Node.js utilizam o framework Express para construir seus serviços web. O Supertest se integra perfeitamente com o Express, permitindo que você teste suas rotas e middleware de forma realista, simulando o comportamento da sua aplicação em produção.

É crucial configurar adequadamente o ambiente de teste para que ele reflita o mais próximo possível o ambiente de produção. Isso garante que seus testes de integração sejam confiáveis e representem com precisão o comportamento da sua aplicação em um cenário real.

Exemplo

1

Configurar o Servidor

Antes de escrever os testes de integração, é necessário configurar um servidor de teste para simular o ambiente do aplicativo. Utilizamos a biblioteca Express para criar um servidor e definir as rotas que serão testadas.

2

Testar Endpoints de API

Com o servidor de teste configurado, podemos utilizar a biblioteca Supertest para fazer requisições HTTP aos endpoints da API e verificar se as respostas estão de acordo com o esperado.

3

Validar Respostas e Status

Nos testes, verificamos se as respostas recebidas possuem os dados corretos e se os status HTTP retornados são os esperados, como 200 para sucesso, 400 para erro de validação, entre outros.

4

Limpar Dados de Teste

Para garantir a independência dos testes, é importante limpar os dados de teste antes de cada execução. Isso pode envolver esvaziar arrays, deletar registros do banco de dados ou restaurar o estado inicial do sistema.

Exemplo

Fork: https://github.com/daniellferreira/class-integration-test

```
cd class-integration-test
npm init -y
npm install express
npm install --save-dev jest supertest
```

Seguimos no vscode

Melhores Práticas em Testes de Integração

Preparação do Ambiente

Criar um ambiente de teste isolado, utilizando bancos de dados em memória ou mockados, para garantir a independência e a rapidez dos testes de integração.

Isolamento de Testes

Garantir que cada teste de integração seja independente e não interfira nos outros, evitando dependências e efeitos colaterais indesejados.

Cenários Abrangentes

Criar uma variedade de casos de teste, incluindo entradas válidas e inválidas, para garantir que o sistema seja testado de forma completa e robusta.

Nos testes de integração, é fundamental preparar o ambiente de teste de forma adequada, utilizando bancos de dados em memória ou mockados para tornar os testes mais rápidos. Além disso, devemos isolar os testes de integração para que um não interfira no outro. Por fim, é importante cobrir uma variedade de cenários, incluindo entradas válidas e inválidas, para garantir que o sistema lide corretamente com diferentes situações.

Exercício Prático: API de Livros

Desenvolva testes de integração para uma API de biblioteca que permite gerenciar livros e empréstimos. A API deve permitir adicionar, listar, atualizar e remover livros, bem como realizar empréstimos e devoluções.

Implementar as rotas da API:

- `POST /books` para adicionar um novo livro.
- `GET /books` para listar todos os livros.
- `PUT /books/:id` para atualizar informações de um livro.
- `DELETE /books/:id` para remover um livro.
- `POST /loans` para realizar um empréstimo.
- `POST /returns` para realizar a devolução de um livro.

Escrever testes de integração que verifiquem:

- A adição de um novo livro com dados válidos.
- A validação ao tentar adicionar um livro com dados inválidos.
- A listagem de todos os livros.
- A atualização das informações de um livro existente.
- A remoção de um livro e confirmação de que ele não existe mais.
- A realização de um empréstimo, garantindo que o livro não esteja disponível para outro empréstimo.
- A realização da devolução de um livro, tornando-o disponível novamente.
- O comportamento ao tentar emprestar um livro indisponível ou inexistente.

==> Utilizar um banco de dados em memória ou mock para os testes.