

Fluxo Caixa

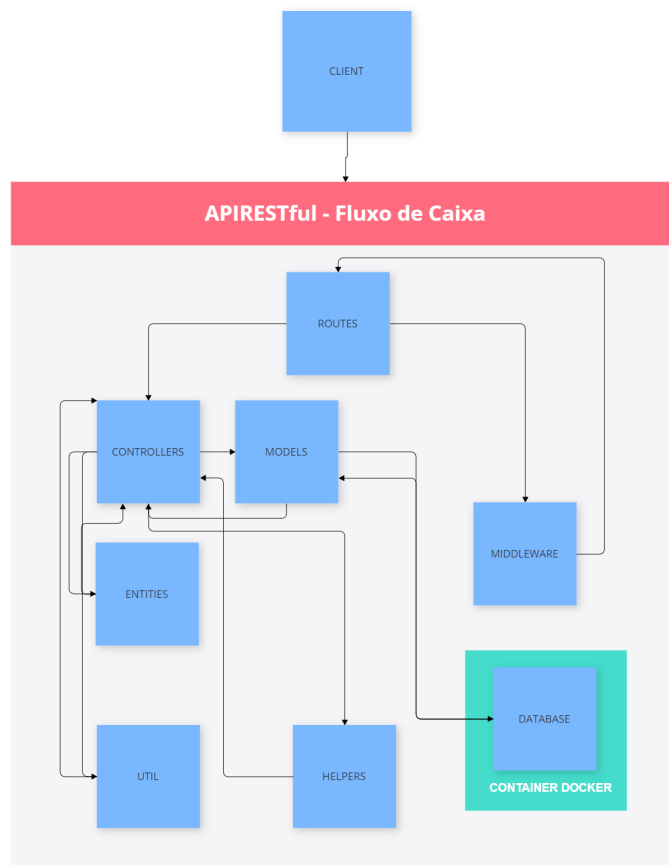
APIRESTful (Representational State Transfer) de fluxo de caixa em Node.js baseada em uma arquitetura em camadas (layers architecture), onde cada camada é responsável por uma funcionalidade específica da aplicação.

A pasta "models" contém as definições de dados usados na aplicação, enquanto a pasta "controllers" contém a lógica de negócios que manipula esses dados. A pasta "database" lida com a comunicação com o banco de dados.

As pastas "helpers", "middleware", "entities" e "util" são utilitárias e auxiliares para as outras camadas. O arquivo "app.js" contém a configuração inicial do aplicativo, o arquivo "server.js" é responsável por iniciar o servidor e o arquivo "router.js" é responsável pelo mapeamento de rotas específicas para os respectivos controladores.

Em resumo, é uma aplicação Node.js com uma arquitetura em camadas, com as camadas de modelo, controle e infraestrutura claramente definidas e pastas auxiliares para funções úteis e de ajuda.

Desenho da arquitetura



Padrões de Arquitetura em Camadas

- Controllers
- Database Service
- Entities
- Helpers
- Middleware
- MModels
- Util

Padrões de projetos utilizados

- Layers Architecture
- Clean Code

Tecnologias e bibliotecas usadas

- Nodejs v14.0
- MySql v5.7
- Docker
- Bcrypt
- Body Parser
- Dotenv
- Express
- Json Web Token
- Moment
- Jest e Supertest

Execução do projeto

Obg: É necessario ter o docker instalado

- Navegar até o diretório \backend
- Executar o comando: npm run dev
- Ter o docker instalado rodar o banco de dados
- Acessar primeiramente a rota <http://localhost:3000/authentication>
- Enviar no body um json com username: "adm" e password: 123
- Após isso vai gerar um token com informações do usuário
- Para acessar as demais rotas <http://localhost:3000/transaction> e <http://localhost:3000/balanceDaily>

- É necessário informar esse token que foi gerado no Headers - x-access-token e também colocar o Content-Type com o valor application/json

Testes

- Foram utilizados testes unitários. A imagem abaixo mostra a cobertura de código dos serviços

```
mateu@mateusdev MINGW64 ~/Documents/projetos/nodejs/test-carrefour-api/backend/src/controllers (main)
$ npm test AuthenticationController.test.js
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.

> teste-carrefour-api@1.0.0 test
> jest "AuthenticationController.test.js"

PASS src/controllers/AuthenticationController.test.js
  POST /authentication
    ✓ should return a token when given valid credentials (161 ms)
    ✓ should return a 401 error when given invalid credentials (12 ms)

Test Suites: 1 passed, 1 total
Tests: 2 passed, 2 total
Snapshots: 0 total
Time: 1.349 s, estimated 2 s
Ran all test suites matching /AuthenticationController.test.js/i.
Jest did not exit one second after the test run has completed.

'This usually means that there are asynchronous operations that weren't stopped in your tests. Consider running Jest with `--detectOpenHandles` to troubleshoot this issue.'
```

```
mateu@mateusdev MINGW64 ~/Documents/projetos/nodejs/test-carrefour-api/backend/src/controllers (main)
$ npm test TransactionsController.test.js
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.

> teste-carrefour-api@1.0.0 test
> jest "TransactionsController.test.js"

PASS src/controllers/TransactionsController.test.js
  POST /transaction
    ✓ should create a new transaction (93 ms)
    ✓ should return an error for invalid data (5 ms)
  GET /balanceDaily
    ✓ should get a consolidated balance sheet (15 ms)
    ✓ should get a consolidated balance sheet with invalid token (7 ms)

Test Suites: 1 passed, 1 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 1.16 s, estimated 2 s
Ran all test suites matching /TransactionsController.test.js/i.
Jest did not exit one second after the test run has completed.
```

Observabilidade

- Nada consta