Workflow com Claude Code + Cursor

Fluxos de IA no desenvolvimento em 2026

Alan Rocha · Abril / 2026

Cursor · Claude Code · RTK · Caveman · Settings scopes

Agenda

1. Por que isto importa em 2026
2. Cursor — o que é, como funciona, começando, exemplo
3. Claude Code — o que é, como funciona, começando, exemplo
4. Tabela comparativa Cursor × Claude Code
5. Workflow combinado
6. Economia de contexto: **RTK** e **Caveman**
7. Settings scopes — e o poder do **managed**
8. Anti-padrões, recursos e Q&A

Por que isto importa em 2026

- IA deixou de ser autocomplete: hoje é **par de programação agêntico**.
- Duas superfícies dominam o dia a dia:
  - **Cursor** → editor com IA embutida (edição cirúrgica, feedback visual).
  - **Claude Code** → CLI agêntico (tarefas longas, multi-arquivo, automação).
- Usar **a ferramenta certa para cada momento** (e controlar contexto/custo) é o que separa produtividade real de "brincar com IA".

> A questão não é "qual é melhor?", é **quando usar cada uma**.

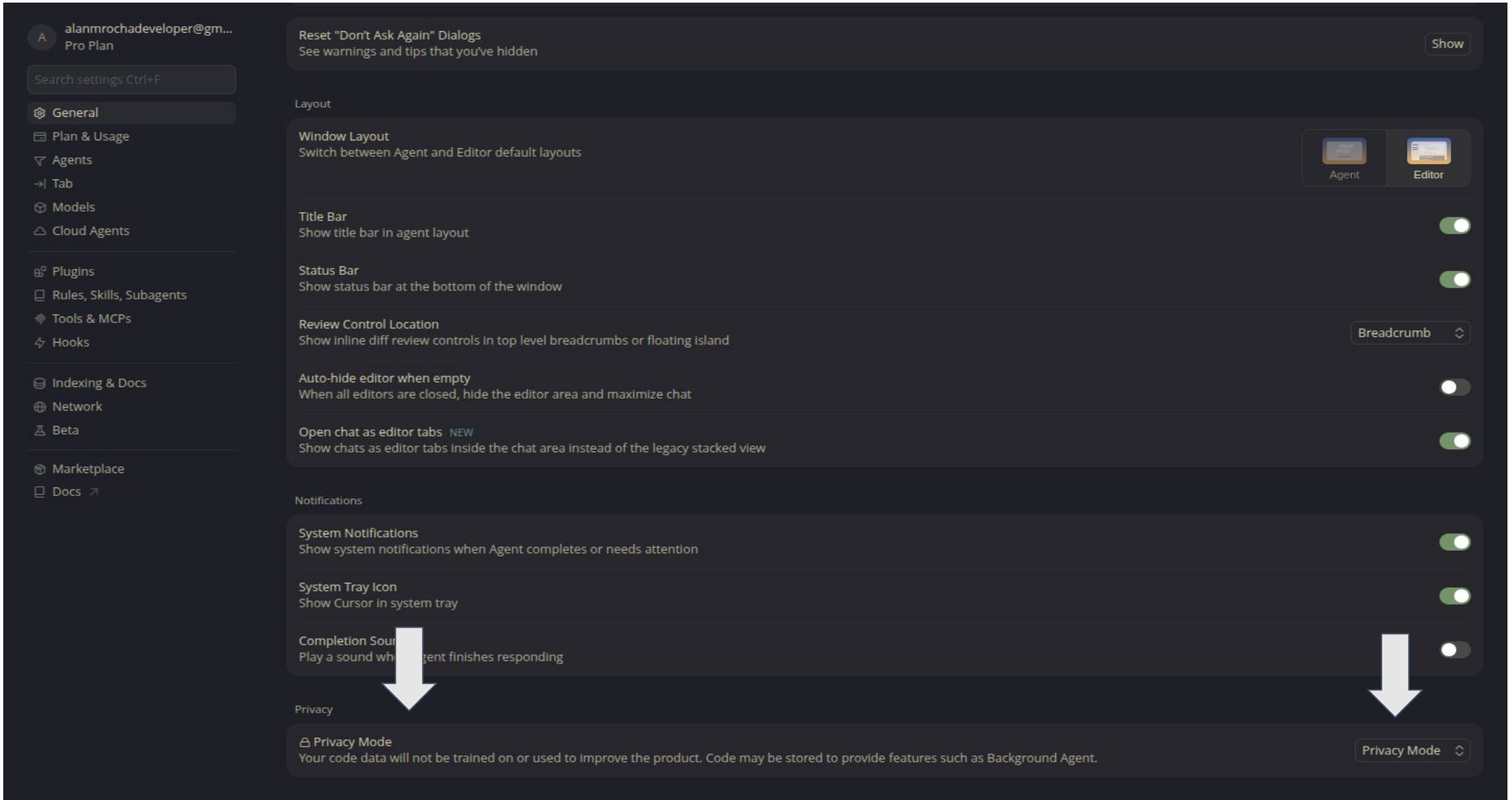
Cursor — o que é

- **IDE fork do VS Code** com IA nativa, feita pela Anysphere.
- Quatro modos de interação:
  - **Tab** — autocomplete preditivo multi-linha.
  - **Chat (Ask)** — perguntas com contexto do arquivo/projeto.
  - **Composer** — edição multi-arquivo via prompt.
  - **Agent** — execução autônoma com acesso a ferramentas.
- Compatível com extensões do VS Code.

Cursor — como funciona

- **Contexto**: `@file`, `@folder`, `@web`, `@docs`, `@git`, imagens.
- **Rules** (`.cursor/rules/*.mdc`) — instruções persistentes por projeto.
- **Modelos**: Claude, GPT, Gemini, modelos próprios (`cursor-small`).
- **Privacy mode**: código não é usado para treino.
- **Indexação de repositório**: embeddings locais para busca semântica.

Onde fica o privacy mode do cursor:



```
.cursor/rules/
  10-language.mdc      # "responda em pt-BR, código em inglês"
  20-style.mdc         # "sem comentários triviais; named exports"
  30-testing.mdc       # "cada nova função precisa de teste Vitest"
```

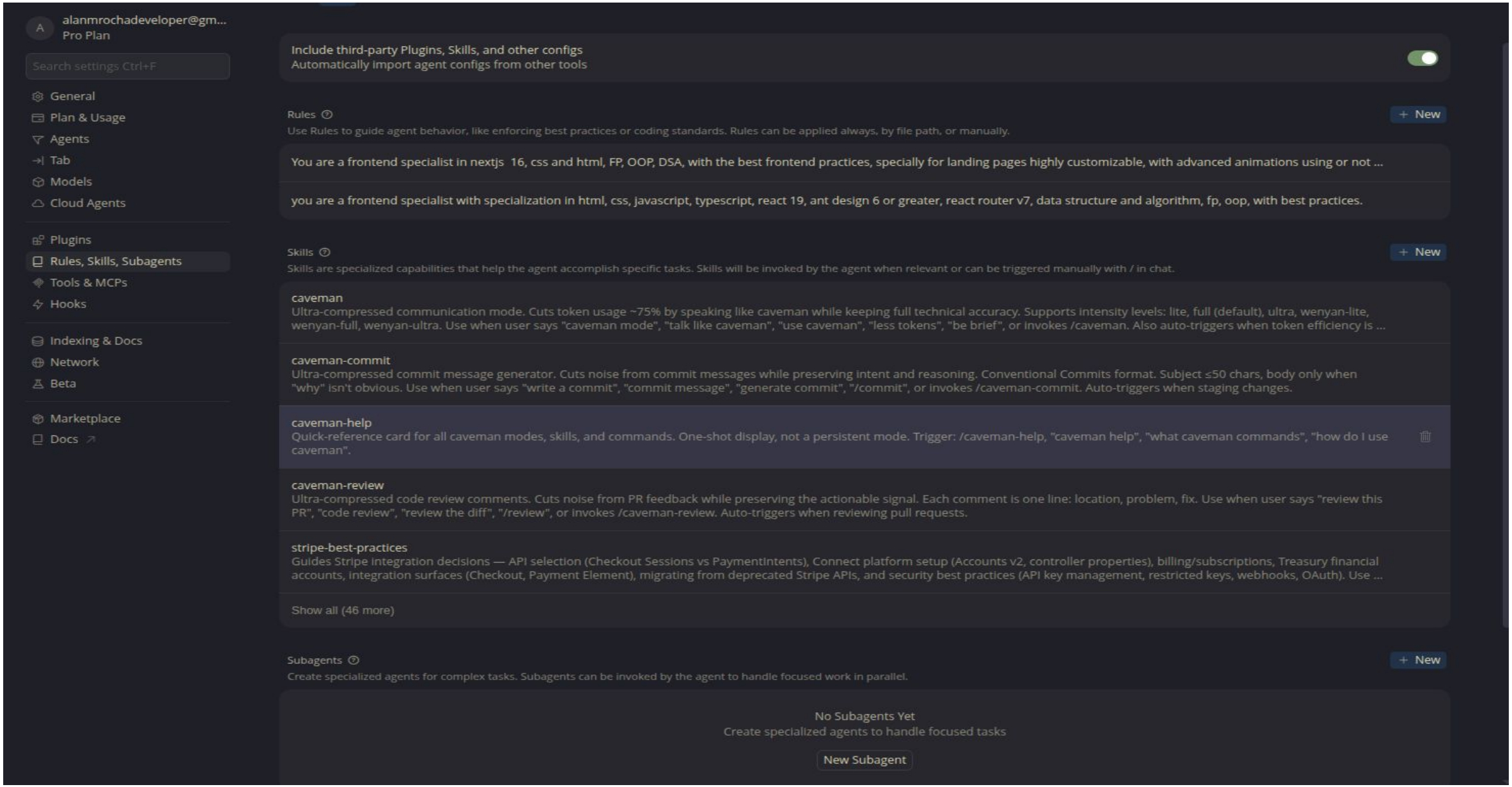
Cursor — começando

1. Baixar em `cursor.com` → instalar → entrar com conta.
2. Abrir um projeto. `Ctrl/Cmd + K` → edit inline. `Ctrl/Cmd + L` → chat.
3. `Ctrl/Cmd + I` → Composer (multi-arquivo).
4. Criar regras do projeto:


```
.cursor/rules/
  10-language.mdc      # "responda em pt-BR, código em inglês"
  20-style.mdc         # "sem comentários triviais; named exports"
  30-testing.mdc       # "cada nova função precisa de teste Vitest"
```


5. Ativar **Privacy mode** em Settings → Beta.

Exemplo da configuração direto no cursor:



Cursor — exemplo prático

Cenário: refatorar `UserCard.tsx` para receber `variant`.

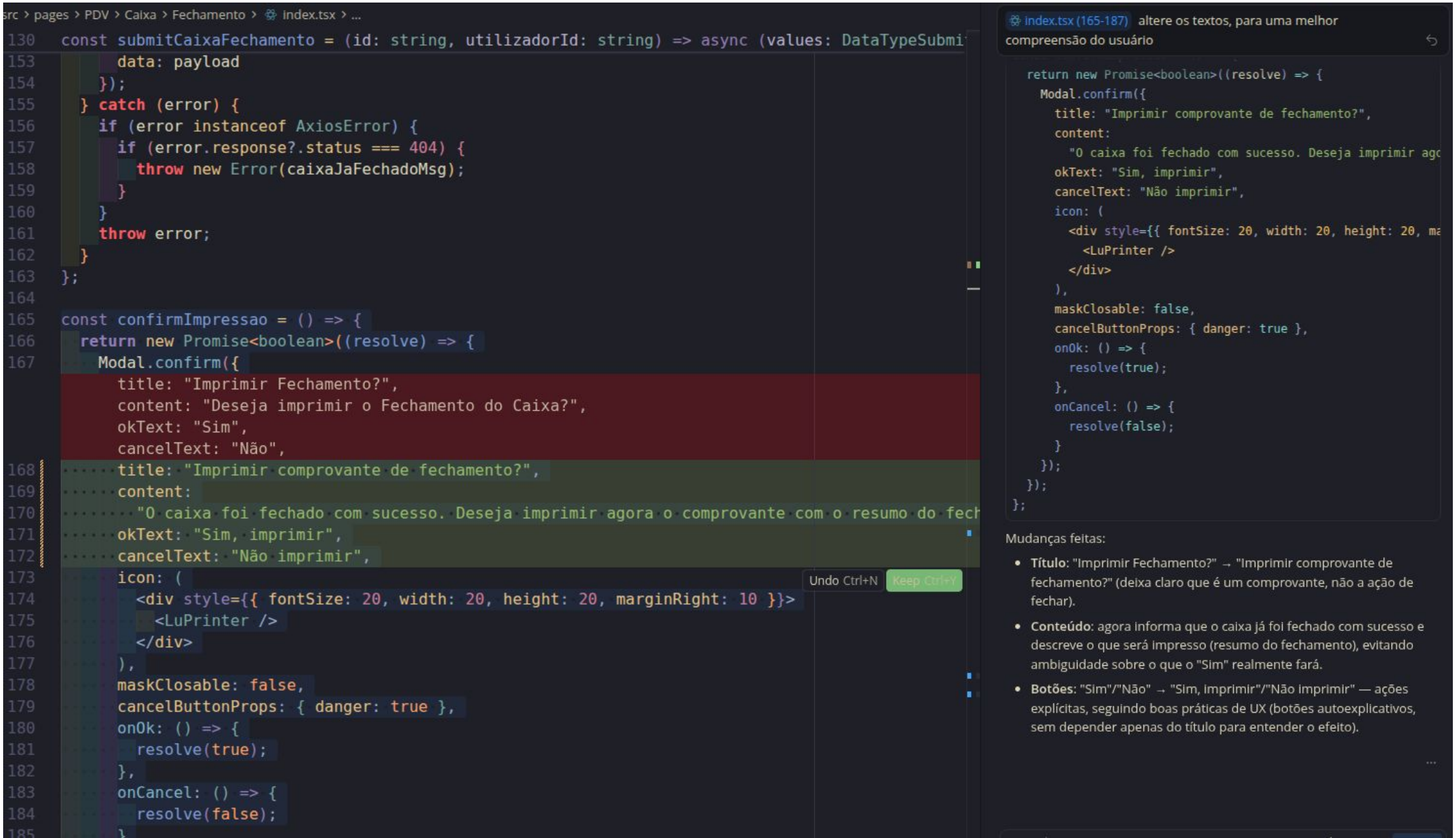
```
Ctrl/Cmd + I  (Composer)

> @UserCard.tsx @UserList.tsx
> Adicione a prop `variant: 'compact' | 'detailed'` em UserCard,
> mantendo 'detailed' como default. Atualize UserList para passar
> 'compact'. Não altere testes sem necessidade.
```

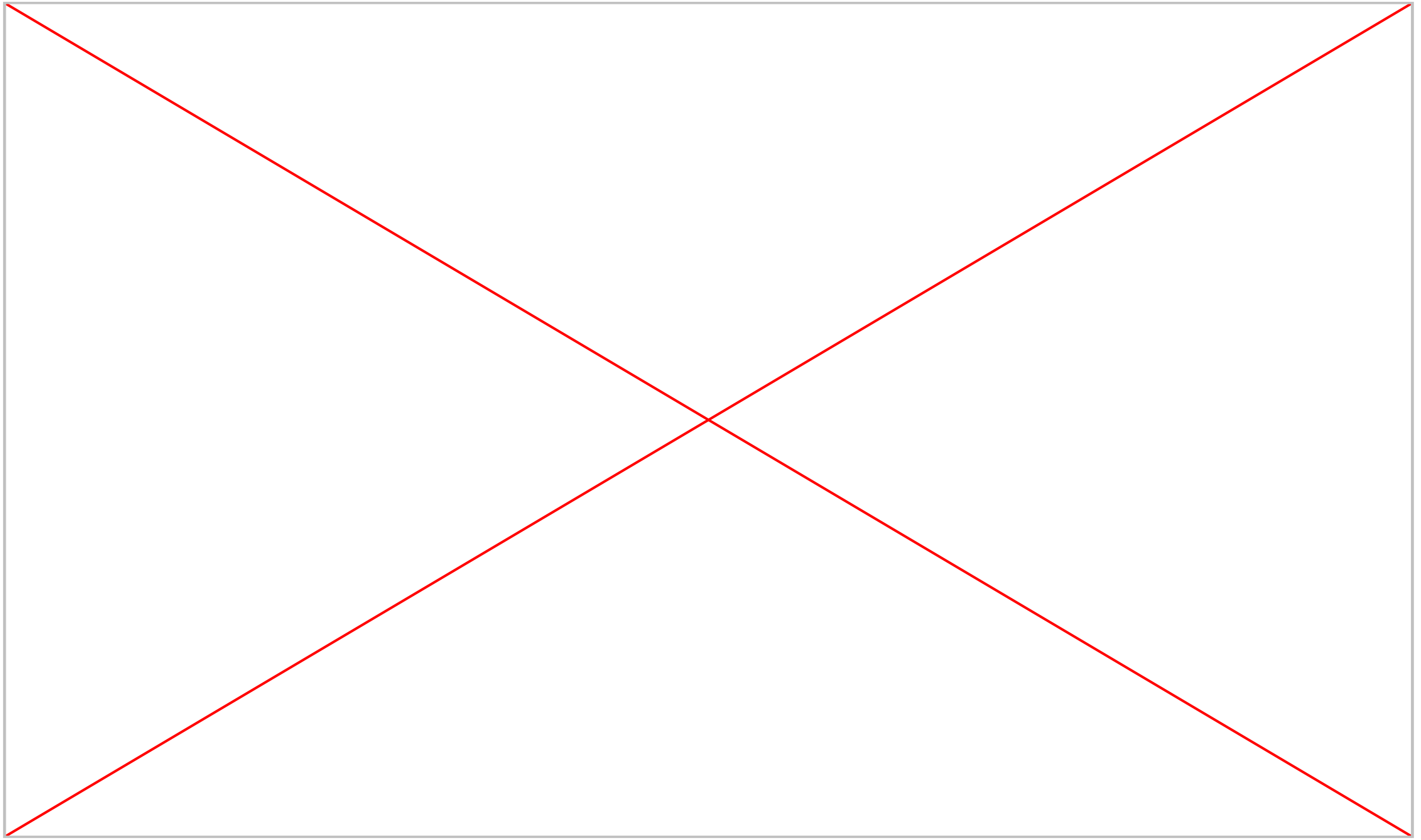
Resultado: diff multi-arquivo com apply/revert por hunk.

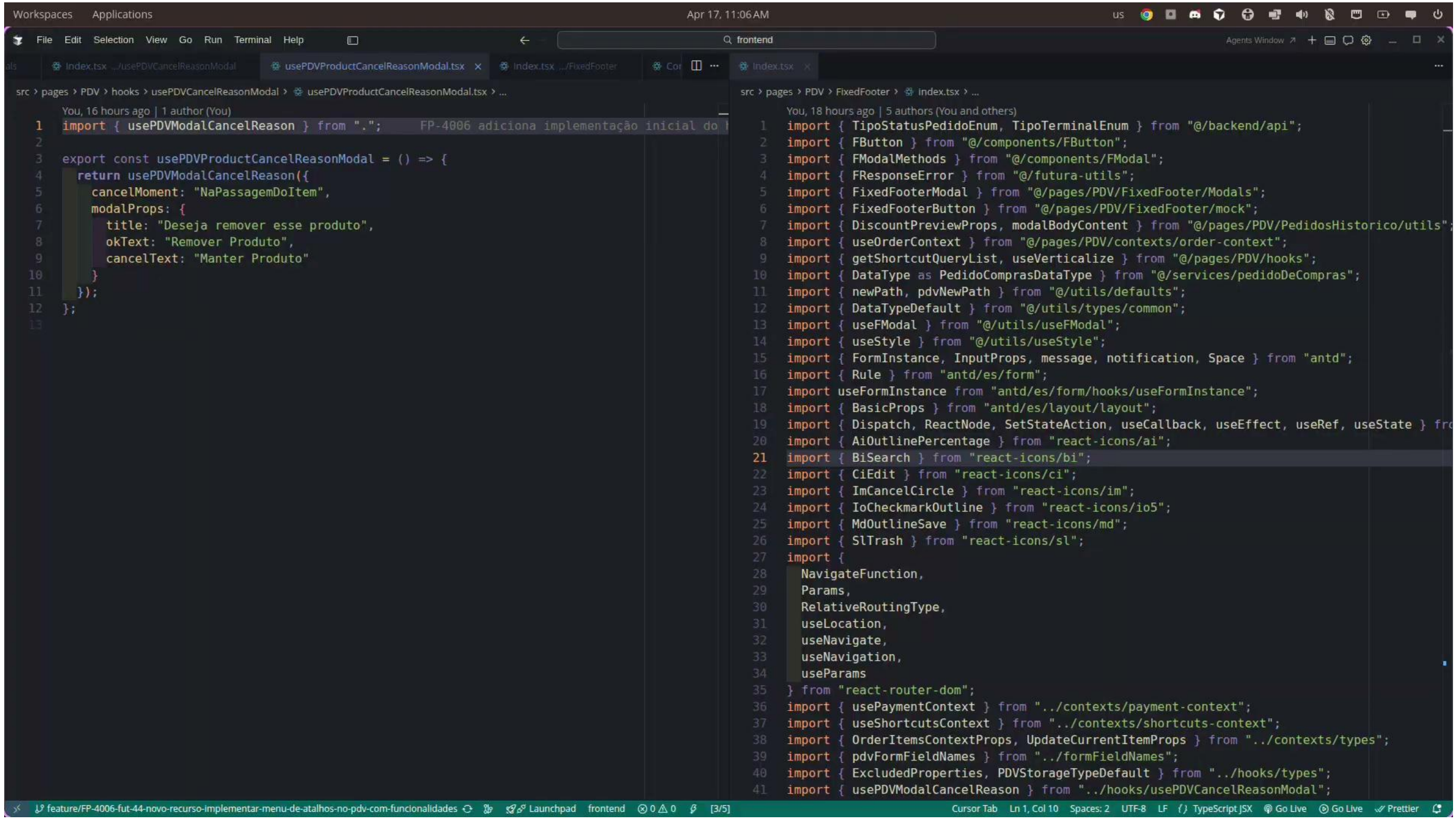
Cursor é **imbatível** quando você já sabe **qual arquivo** quer tocar.

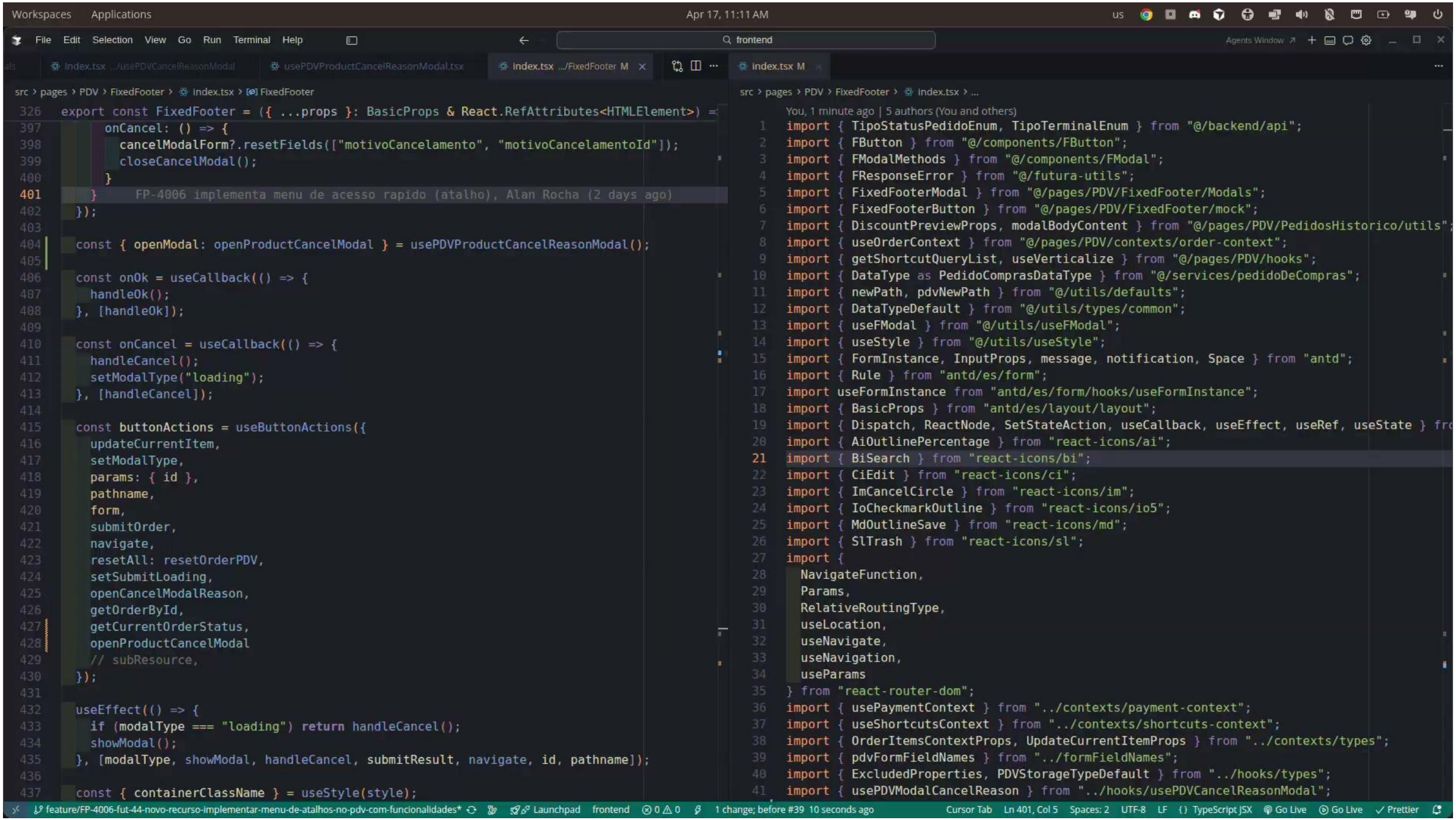
Um exemplo real:



Um exemplo de edição inline com o cursor:



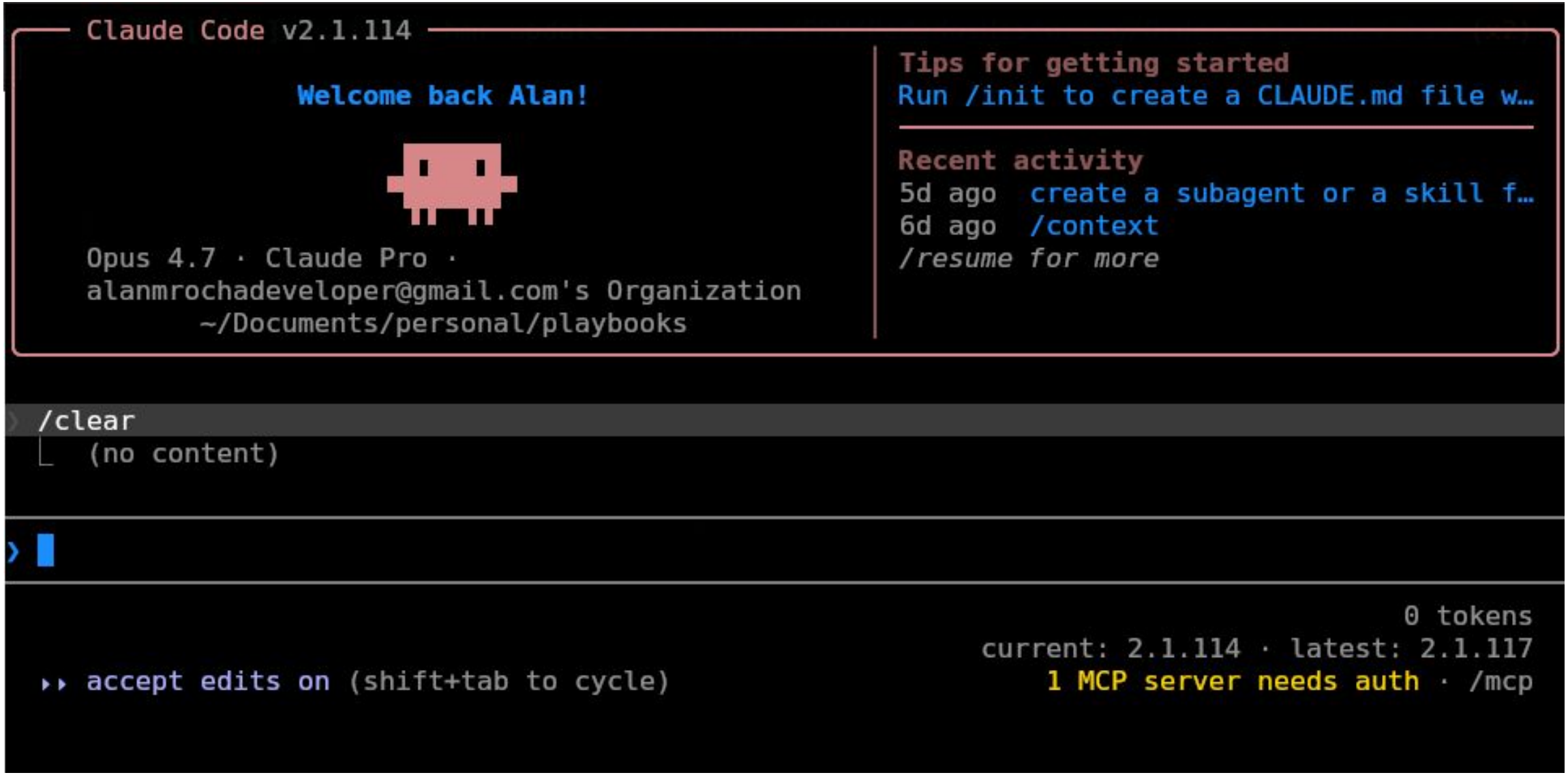

Workspaces
Applications
Apr 17, 11:06 AM
us
File
Edit
Selection
View
Go
Run
Terminal
Help
frontend
Agents Window
Index.tsx .../usePDVCancelReasonModal
usePDVProductCancelReasonModal.tsx
Index.tsx .../FixedFooter
Index.tsx
src › pages › PDV › hooks › usePDVCancelReasonModal › usePDVProductCancelReasonModal.tsx › ...
You, 16 hours ago | 1 author (You)
import { usePDVModalCancelReason } from ".";
FP-4006 adiciona implementação inicial do
export const usePDVProductCancelReasonModal = () => {
return usePDVModalCancelReason({
cancelMoment: "NaPassagemDoItem",
modalProps: {
title: "Deseja remover esse produto",
okText: "Remover Produto",
cancelText: "Manter Produto"
}
});
};
src › pages › PDV › FixedFooter › index.tsx › ...
You, 18 hours ago | 5 authors (You and others)
import { TipoStatusPedidoEnum, TipoTerminalEnum } from "@/backend/api";
import { FButton } from "@/components/FButton";
import { FModalMethods } from "@/components/FModal";
import { FResponseError } from "@/futura-utils";
import { FixedFooterModal } from "@/pages/PDV/FixedFooter/Modals";
import { FixedFooterButton } from "@/pages/PDV/FixedFooter/mock";
import { DiscountPreviewProps, modalBodyContent } from "@/pages/PDV/PedidosHistorico/utils";
import { useOrderContext } from "@/pages/PDV/contexts/order-context";
import { getShortcutQueryList, useVerticalize } from "@/pages/PDV/hooks";
import { DataType as PedidoComprasDataType } from "@/services/pedidoDeCompras";
import { newPath, pdvNewPath } from "@/utils/defaults";
import { DataTypeDefault } from "@/utils/types/common";
import { useFModal } from "@/utils/useFModal";
import { useStyle } from "@/utils/useStyle";
import { FormInstance, InputProps, message, notification, Space } from "antd";
import { Rule } from "antd/es/form";
import useFormInstance from "antd/es/form/hooks/useFormInstance";
import { BasicProps } from "antd/es/layout/layout";
import { Dispatch, ReactNode, SetStateAction, useCallback, useEffect, useRef, useState } fr
import { AiOutlinePercentage } from "react-icons/ai";
import { BiSearch } from "react-icons/bi";
import { CiEdit } from "react-icons/ci";
import { ImCancelCircle } from "react-icons/im";
import { IoCheckmarkOutline } from "react-icons/io5";
import { MdOutlineSave } from "react-icons/md";
import { SlTrash } from "react-icons/sl";
import {
NavigateFunction,
Params,
RelativeRoutingType,
useLocation,
useNavigate,
useNavigation,
useParams
} from "react-router-dom";
import { usePaymentContext } from "../contexts/payment-context";
import { useShortcutsContext } from "../contexts/shortcuts-context";
import { OrderItemsContextProps, UpdateCurrentItemProps } from "../contexts/types";
import { pdvFormFieldNames } from "../formFieldNames";
import { ExcludedProperties, PDVStorageTypeDefault } from "../hooks/types";
import { usePDVModalCancelReason } from "../hooks/usePDVCancelReasonModal";
feature/FP-4006-fut-44-novo-recurso-implementar-menu-de-atalhos-no-pdv-com-funcionalidades
Launchpad
frontend
[3/5]
Cursor Tab
Ln 1, Col 10
Spaces: 2
UTF-8
LF
TypeScript JSX
Go Live
Go Live
Prettier


Workspaces
Applications
Apr 17, 11:11 AM
us
File
Edit
Selection
View
Go
Run
Terminal
Help
frontend
Agents Window
Index.tsx .../usePDVCancelReasonModal
usePDVProductCancelReasonModal.tsx
index.tsx .../FixedFooter M
index.tsx M
src › pages › PDV › FixedFooter › index.tsx › FixedFooter
src › pages › PDV › FixedFooter › index.tsx › ...
You, 1 minute ago | 5 authors (You and others)
export const FixedFooter = ({ ...props }: BasicProps & React.RefAttributes<HTMLElement>) =
onCancel: () => {
cancelModalForm?.resetFields(["motivoCancelamento", "motivoCancelamentoId"]);
closeCancelModal();
}
} FP-4006 implementa menu de acesso rapido (atalho), Alan Rocha (2 days ago)
});
const { openModal: openProductCancelModal } = usePDVProductCancelReasonModal();
const onOk = useCallback(() => {
handleOk();
}, [handleOk]);
const onCancel = useCallback(() => {
handleCancel();
setModalType("loading");
}, [handleCancel]);
const buttonActions = useButtonActions({
updateCurrentItem,
setModalType,
params: { id },
pathname,
form,
submitOrder,
navigate,
resetAll: resetOrderPDV,
setSubmitLoading,
openCancelModalReason,
getOrderById,
getCurrentOrderStatus,
openProductCancelModal
// subResource,
});
useEffect(() => {
if (modalType === "loading") return handleCancel();
showModal();
}, [modalType, showModal, handleCancel, submitResult, navigate, id, pathname]);
const { containerClassName } = useStyle(style);
import { TipoStatusPedidoEnum, TipoTerminalEnum } from "@/backend/api";
import { FButton } from "@/components/FButton";
import { FModalMethods } from "@/components/FModal";
import { FResponseError } from "@/futura-utils";
import { FixedFooterModal } from "@/pages/PDV/FixedFooter/Modals";
import { FixedFooterButton } from "@/pages/PDV/FixedFooter/mock";
import { DiscountPreviewProps, modalBodyContent } from "@/pages/PDV/PedidosHistorico/utils";
import { useOrderContext } from "@/pages/PDV/contexts/order-context";
import { getShortcutQueryList, useVerticalize } from "@/pages/PDV/hooks";
import { DataType as PedidoComprasDataType } from "@/services/pedidoDeCompras";
import { newPath, pdvNewPath } from "@/utils/defaults";
import { DataTypeDefault } from "@/utils/types/common";
import { useFModal } from "@/utils/useFModal";
import { useStyle } from "@/utils/useStyle";
import { FormInstance, InputProps, message, notification, Space } from "antd";
import { Rule } from "antd/es/form";
import useFormInstance from "antd/es/form/hooks/useFormInstance";
import { BasicProps } from "antd/es/layout/layout";
import { Dispatch, ReactNode, SetStateAction, useCallback, useEffect, useRef, useState } fr
import { AiOutlinePercentage } from "react-icons/ai";
import { BiSearch } from "react-icons/bi";
import { CiEdit } from "react-icons/ci";
import { ImCancelCircle } from "react-icons/im";
import { IoCheckmarkOutline } from "react-icons/io5";
import { MdOutlineSave } from "react-icons/md";
import { SlTrash } from "react-icons/sl";
import {
NavigateFunction,
Params,
RelativeRoutingType,
useLocation,
useNavigate,
useNavigation,
useParams
} from "react-router-dom";
import { usePaymentContext } from "../contexts/payment-context";
import { useShortcutsContext } from "../contexts/shortcuts-context";
import { OrderItemsContextProps, UpdateCurrentItemProps } from "../contexts/types";
import { pdvFormFieldNames } from "../formFieldNames";
import { ExcludedProperties, PDVStorageTypeDefault } from "../hooks/types";
import { usePDVModalCancelReason } from "../hooks/usePDVCancelReasonModal";
feature/FP-4006-fut-44-novo-recurso-implementar-menu-de-atalhos-no-pdv-com-funcionalidades*
Launchpad
frontend
0 0
1 change; before #39 10 seconds ago
Cursor Tab
Ln 401, Col 5
Spaces: 2
UTF-8
LF
TypeScript JSX
Go Live
Go Live
Prettier

Claude Code — o que é

- **CLI agêntico oficial da Anthropic**, rodando no terminal ou dentro do IDE.
- Opera por **sessões**: lê arquivos, edita, roda comandos, testa.
- Extensível por:
  - **Hooks** (eventos do ciclo de vida).
  - **MCP** (servidores que expõem ferramentas externas).
  - **Sub-agents** e **skills** (especialistas invocáveis).
  - **Plugins** oficiais e comunitários.

Um exemplo de como é CLI no terminal do Claude Code:



Claude Code — como funciona

- `CLAUDE.md` — arquivo de memória lido no início da sessão (user/project).
- **Tools nativas**: `Read`, `Edit`, `Write`, `Bash`, `Grep`, `WebFetch`…
- **Plan mode** (`/plan`) — planeja antes de executar; nada é escrito sem aprovação.
- **Sub-agents** — delegam pesquisa/planejamento sem poluir contexto principal.
- **Hooks** em `settings.json` disparam comandos em eventos (`PreToolUse`, `Stop`…).

Claude Code — CLAUDE.md: hierarquia

Lido no início de toda sessão. Mais específico tem precedência.

Escopo	Caminho	Compartilhado?
Managed (org)	`/Library/Application Support/ClaudeCode/CLAUDE.md` · `/etc/claude-code/CLAUDE.md` · `C:\ProgramData\ClaudeCode\CLAUDE.md`	todos na máquina
User global	`~/.claude/CLAUDE.md`	só você
Project	`./CLAUDE.md` ou `./.claude/CLAUDE.md`	sim (git)
Local	`./CLAUDE.local.md`	não (gitignore)

Imports: dentro de qualquer CLAUDE.md, use `@caminho/arquivo.md` para incluir outro — modulariza instruções sem inflar um só arquivo.

Claude Code — começando

```
curl -fsSL https://claude.ai/install.sh | bash
$claude  # login na primeira execução
```

Comandos essenciais:

Comando	Efeito
`/init`	gera `CLAUDE.md` inicial do projeto
`/plan`	entra em plan mode (read-only até aprovar)
`/model opus`	troca de modelo (opus / sonnet / haiku)
`/clear`	limpa contexto da sessão
`/hooks`	gerencia hooks

Claude Code — exemplo prático

Cenário: remover acoplamento de `OrderService` com `EmailClient`.

```
$ claude
> /plan
> Quero extrair o envio de email de OrderService para um
> NotificationPort (hexagonal). Manter testes verdes.
> Liste arquivos afetados antes de editar.
```

Claude Code:

1. Explora o repo (`grep`, `read`).
2. Propõe um plano (arquivos + diffs previstos).
3. Após **ExitPlanMode**, aplica, roda testes e reporta.

Ideal para **refactors amplos, tarefas longas e automação em CI**.

Cursor × Claude Code

Dimensão	Cursor	Claude Code
Tipo	IDE com IA embutida	CLI agêntico (+ extensão IDE)
Superfície	Editor visual, diffs inline	Terminal, sessões longas
Interação	Tab, Chat, Composer, Agent	Prompts + tools + hooks + sub-agents
Modelos	Claude, GPT, Gemini, próprios	Claude-first (Opus / Sonnet / Haiku)
Autonomia	Assistência inline + Agent	Execução agêntica multi-step nativa
Contexto	`@file`, `@docs`, Rules, index do repo	`CLAUDE.md`, MCP, memória, sub-agents
Extensibilidade	Rules, MCP, extensões VS Code	Hooks, skills, plugins, sub-agents, MCP
Billing	Assinatura Cursor	Plano Claude / API Anthropic
Melhor para	Edição cirúrgica, UI, debug visual	Refactors amplos, tarefas longas, CI/headless

Workflow combinado

Use Cursor quando:

- Já sabe o arquivo a alterar.
- Quer feedback visual imediato, diff hunk a hunk.
- Está depurando UI / estilos / pequenas correções.

Use Claude Code quando:

- A tarefa toca muitos arquivos e não sabe todos.
- Precisa de **plano antes de executar** (refactor grande, migração).
- Quer rodar em CI/headless, com hooks e política de permissões.

Combo clássico: Claude Code planeja e aplica mudanças amplas → Cursor faz fine-tuning visual no diff.

Economia de contexto — por que importa

- Janela de contexto é **finita** — mais tokens = mais custo + mais latência.
- Sessões longas **degradam qualidade** (atenção dispersa).
- Output verboso ("Certo! Vou agora…") consome limite sem entregar valor.
- Input inchado (CLAUDE.md gigante, arquivos lidos à toa) é o maior vilão.

> Dois aliados no dia a dia: **RTK** (lado CLI) e **Caveman** (lado prompt).

RTK — Rust Token Killer

Proxy CLI que filtra saídas de `git`, `npm`, `ls`, etc. **antes** de chegarem no contexto.

```
rtk --version          # checagem
rtk gain               # analytics de economia (60–90% típico)
rtk gain --history     # histórico por comando
rtk discover           # varre histórico e sugere otimizações
rtk proxy              # executa sem filtro (debug)
```

- **Hook transparente** reescreve `git status` → `rtk git status`.
- Zero mudança de hábito: você continua digitando comandos normais.
- ⚠️ Colisão de nome: `reachingforthejack/rtk` **não é o mesmo** (Rust Type Kit).

https://github.com/rtk-ai <- verdadeiro

Caveman plugin (comunitário)

Skill/plugin de **Julius Brussee** — `github.com/JuliusBrussee/caveman`.
Não é oficial da Anthropic. Reduz verbosidade do modelo.

Modo	Estilo
`/caveman lite`	remove filler ("Claro!", "Vou agora…")
`/caveman full`	respostas em fragmentos
`/caveman ultra`	máxima concisão, estilo telegráfico

Caveman Compress reescreve seu `CLAUDE.md` em estilo comprimido — ataca o lado **input** (maior fatia real de tokens).

Ganho médio em output: ~**65%** em tarefas reais de dev.

Settings scopes no Claude Code

Quatro escopos, precedência `managed > project > local > user`:

Escopo	Caminho	Compartilhado?	Para quê
User	`~/.claude/settings.json`	não	preferências pessoais (modelo, tema)
Project	`.claude/settings.json`	sim (git)	hooks, permissões do time, plugins
Local	`.claude/settings.local.json`	não (gitignore)	overrides por máquina
Managed	diretório do sistema (ver adiante)	imposto por TI	política organizacional, não sobrescrevível

> Um `deny` em **managed** torna a tool indisponível **silenciosamente**.

Managed settings — como criar e blindar

Caminhos oficiais (managed scope):

- macOS: `/Library/Application Support/ClaudeCode/managed-settings.json`
- Linux/WSL: `/etc/claude-code/managed-settings.json`
- Windows: `C:\ProgramData\ClaudeCode\managed-settings.json`

Anti-padrões e dicas finais

- ❌ CLAUDE.md com 500 linhas → use **Caveman Compress**.
- ❌ Rodar `git log` sem `--oneline -10` em cada sessão.
- ❌ Aprovar plano sem ler arquivos afetados.
- ✅ `/plan` **sempre** antes de tarefa multi-arquivo.
- ✅ `rtk gain` semanal para ver o que está economizando.
- ✅ Cursor para hunk-a-hunk, Claude Code para varredura.
- ✅ Hooks para automatizar lint/test após edição.

Recursos + Q&A

Docs & repos

- `docs.anthropic.com/claude-code`
- `cursor.com/docs`
- `github.com/JuliusBrussee/caveman`
- `github.com/anthropics/claude-plugins-official`

Obrigado! 🧠 Dúvidas?