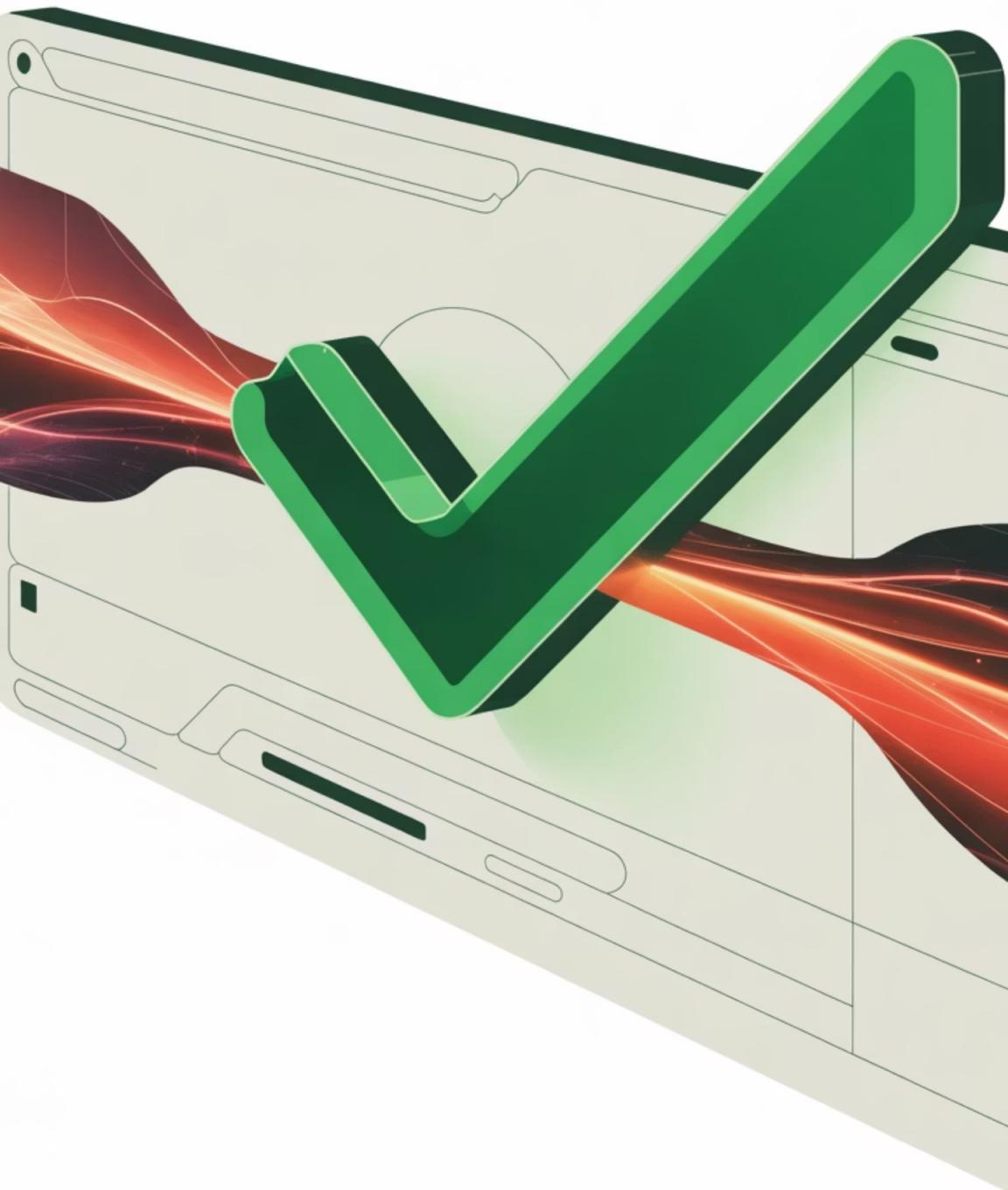


# AI Experiments

To Make Informed Decisions





**AI Says It's OK  
So, Ship It!**



# Quality at Speed

**Pascal Dufour**

**AI development Consultancy**

**Agile Test Consultancy**



# How AI Experiments Support Our Decision-Making

## What is an AI experiment?

- The same as an experiment, now with AI as a deciding factor.

## What is decision making?

- When information leads to a choice.

# rb2

## we develop



### Tech consulting

Onze architecten vertalen jouw businessvraag snel naar een werkbare oplossing.

### Softwareontwikkeling

Onze eigen teams in Portugal en China bouwen onder Nederlandse regie, met slimme inzet van AI om development te versnellen.

### IT-operations & support

We zorgen voor de operationele continuïteit van alles wat we bouwen, van hosting en monitoring tot onderhoud en support.

### Procesoptimalisatie met AI

Ontdek hoe procesoptimalisatie met AI je bedrijf sneller en slimmer laat groeien.

# "Insurance" Company

**DELA**  
voor elkaar

- Active Insurances**  
6M
- Funerals Handled**  
50K+
- Cremations**  
50K+
- Total Employees**  
3000
- IT Staff**  
250



# Organizational Design Differences

## Insurance Company

### Scaled Agile Framework (SAFe) / Scrum Hybrid

- PI planning
- IT teams for IT
- Business IT teams  
("feature teams")
- ...

## Development Company

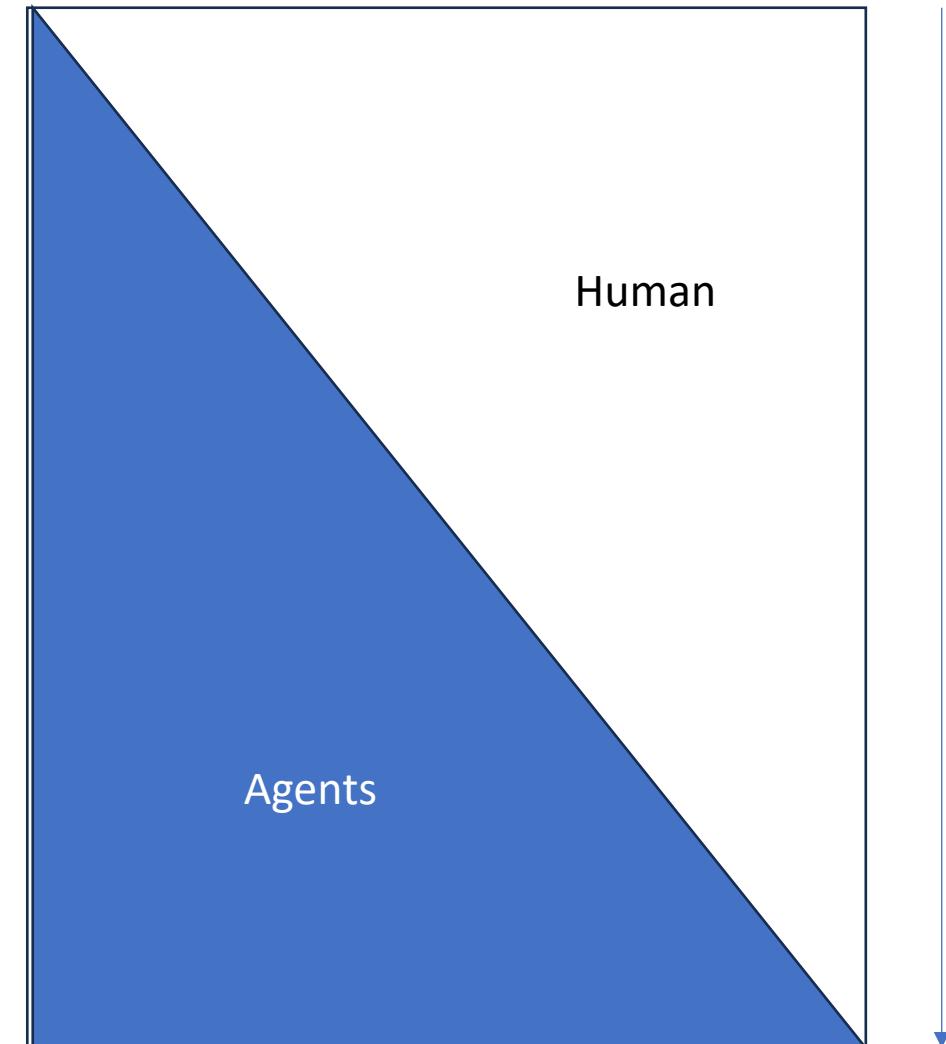
### Project-Based / Scrum

- AI team
- Feature teams
- DevOps
- ...

# AI Transitions in the Dev World

## Scope

- Line-Level**  
Auto-completion in the IDE for single-line suggestions and basic syntax help
- File-Level**  
Small snippets of code generation within a single file context
- Multi-File**  
Multiple file editing with cross-reference understanding and refactoring capabilities
- Application**  
Basic application generation with architectural decision support
- Landscapes**  
Full system landscape understanding and complex integration management



# A year of Gemini progress

amazon-bedrock/eu.anthropic.cl    Claude Sonnet 4.5 (EU)  
amazon-bedrock/global.anthropic.cl    Claude Sonnet 4.5 (Global)  
amazon-bedrock/us.anthropic.cl    Claude Sonnet 4.5 (US)  
amazon-bedrock/cohere.command-  
→ **amazon-bedrock/cohere.command-**    Command R  
amazon-bedrock/us.deepseek.r1-    DeepSeek-R1  
amazon-bedrock/deepseek.v3-v1:    DeepSeek-V3.1  
amazon-bedrock/google.gemma-3-    Gemma 3 4B IT  
amazon-bedrock/google.gemma-3-    Google Gemma 3 27B Instruct  
23/689

## Products & Features

VideoFX

Native Image Gen  
in Gemini 2.0 Flash

Jules

Agentspace

Live Video &  
Screenshare  
in Gemini App

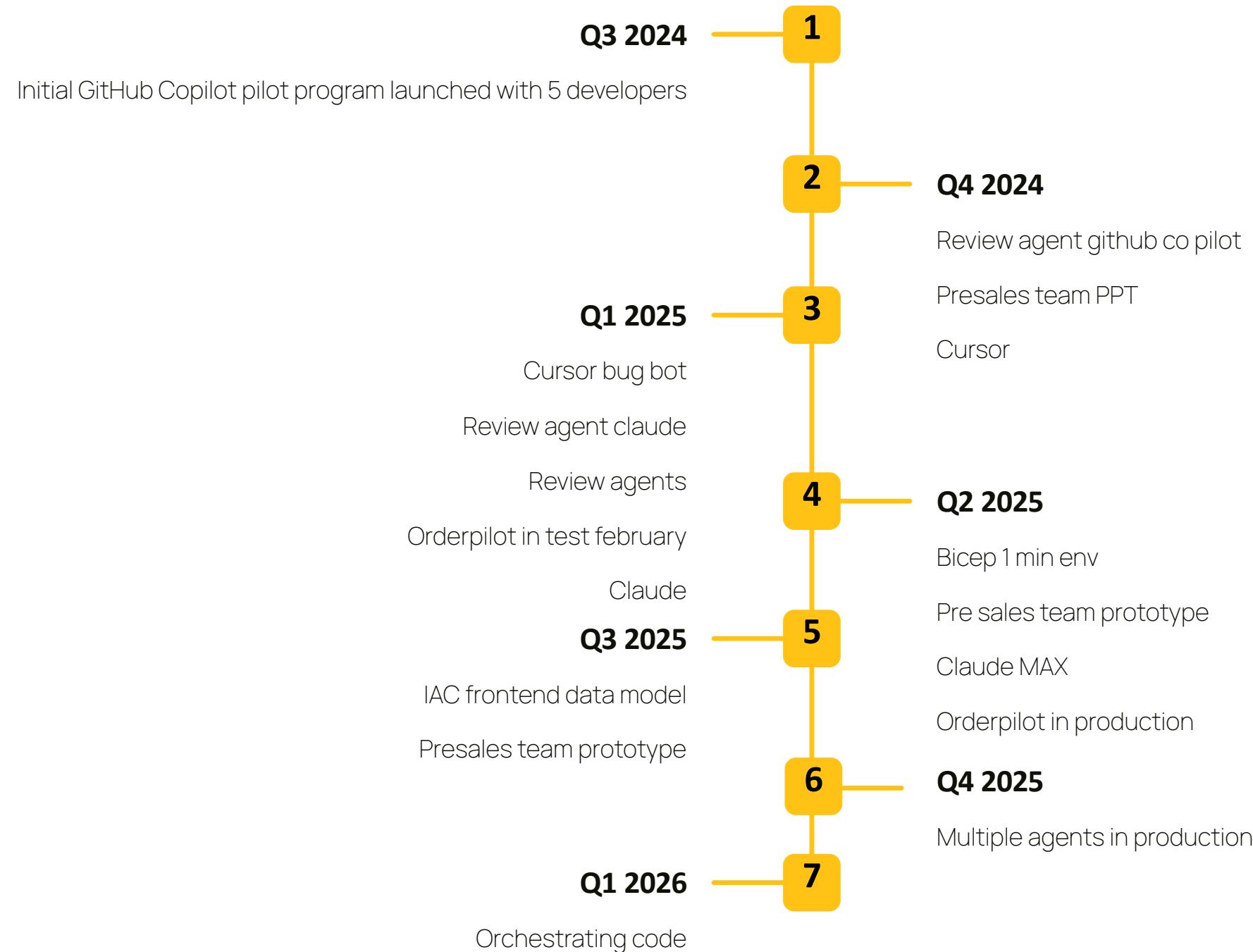
Gemini Canvas

Gemini with  
Personalization



## Timeline: Development Company

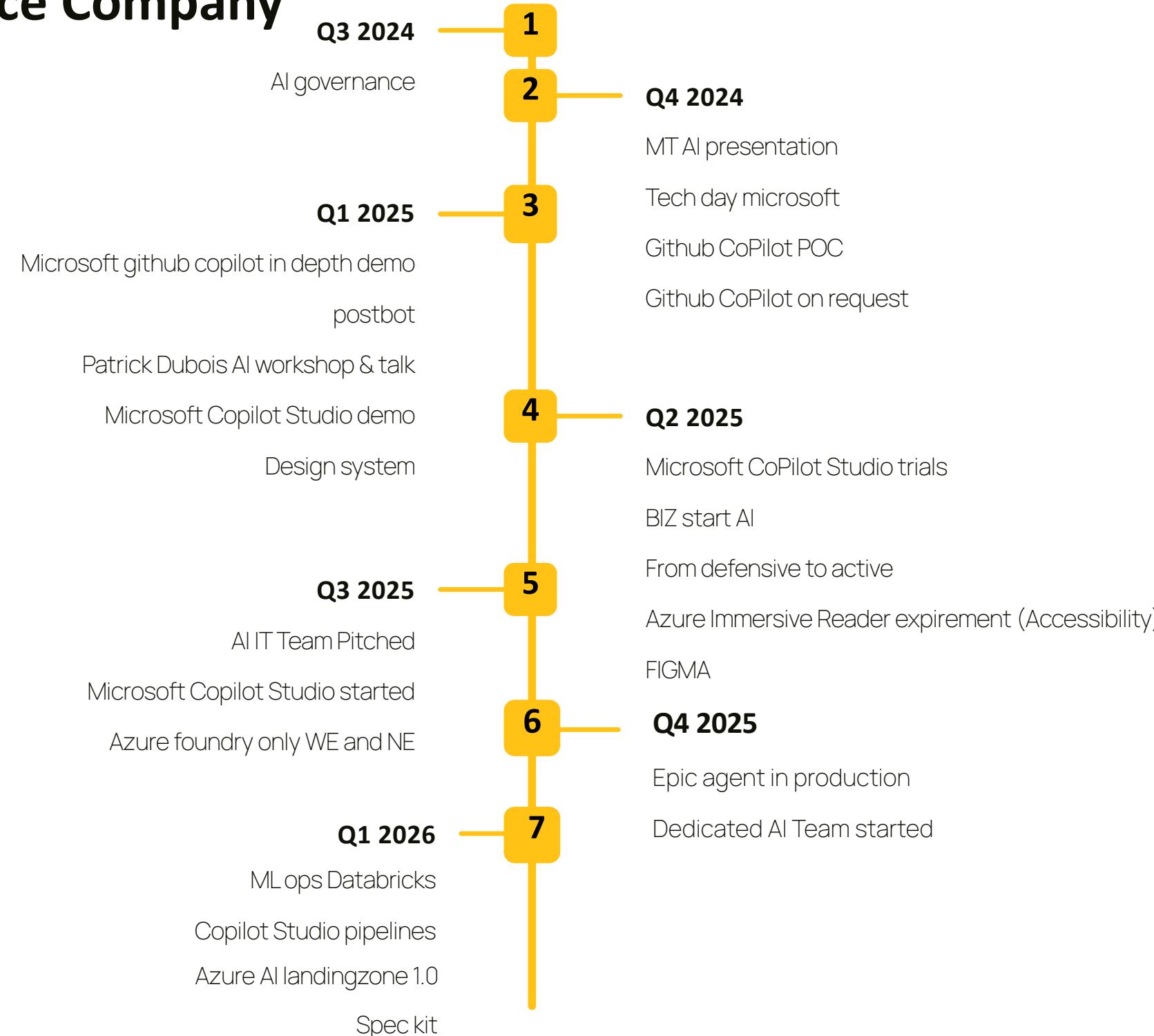
# Timeline: Development Company





## Timeline: Insurance Company

# Timeline: Insurance Company





## Key Timeline Differences

**Development company  
started 3 months earlier  
on Co-Pilot**

**Insurance company gets  
more formal training /  
information sessions**

**AI team formed in  
development company in  
2025 Q2 in insurance  
2025 Q4**

# Experiments Worth Mentioning

## Development Company

- Coding assistants
- **Review bot**
- Prototyping with customers v0, etc.
- Claude code generation
- Multiple Review agents changing code
- Dev agents change code and deploy to production
- Creating Agents for customer workflows
- ...

## Insurance Company

- Azure Copilot, Microsoft Office Copilot,
- Copilot Studio, GitHub Copilot
- **Review bot**
- Release notes
- Vibe coding prototypes
- Document generation
- Design system ( MCP Figma)
- VAT check
- Spec Kit
- Epic AI assistant
- ...



# Insurance Company Deep Dive

Let's explore how the insurance company implemented the AI experiment: the automated review bot.



# Open AI Code Review

[Aidan Cole](#) |  939 installs |  (0) | Free

Integrate Open AI for automated code reviews in Azure DevOps Pull Requests. Enhance code quality and improve review efficiency.

[Get it free](#)

## Simple Idea: A Review Agent

### Step 1: Initial Exploration

#### OpenAI Plugin

OpenAI key needed

- Risky with data?
- Output OK-ish

```
20     self.settings = settings
21     self.azure_devops = AzureDevOpsService(settings)
22     self.openai = OpenAIService(settings)
23     self.git = GitService(settings)
24
25     def run_review(self) -> bool:
26         """Run AI review of the PR."""
27         print("Starting AI review of the PR...")
28
29         try:
30             # Get PR information
31             # Check for duplicate comments if enabled
32             if self.settings.prevent_duplicate_comments:
33                 if self.azure_devops.has_ai_reviewed_pr(self.settings.review_title):
34                     print("AI review already exists in the PR. Skipping review.")
35                     return True
36
37             # Get repository path and diff information
```

# Simple Idea: A Review Agent

## Step 2: Enterprise Solution

### We built an Azure Open AI - Tailored to our insights

- Rights already available
- Cutoff possible (data risk/costs)
- Output OK-ish
- Works in the same repo

## Build Validation 2

Validate code by pre-merging and building pull request changes.

Filter

+

Enabled	Name ↑	Path filter	Trigger	Inheritance
<input checked="" type="checkbox"/> On	<b>Delaonline - Continuous Integration</b> Required		Automatic Expires when code changes	
<input checked="" type="checkbox"/> On	<b>PR Bot pipeline</b> Optional		Automatic Never Expires	

## Status Checks 1

Require other services to post successful status to complete pull requests.

Filter

+

Enabled	Name ↑	Path filter	Inheritance
<input checked="" type="checkbox"/> On	<b>Sonar Cloud Quality Status</b> Required		

# Review Agent Implementation

## Checks

The displayed list of checks may be truncated for optimal performance. For a smoother experience, please maintain a reasonable number of policies (fewer than 100).

X

### Required

 **Delaonline - Continuous Integration**   
Build succeeded

 **Work items must be linked**   
Succeeded

 **Comments must be resolved**   
Succeeded

### Optional

 **PR Bot pipeline**  
Build succeeded

 **[delanl\_online\_libraries] Quality Gate passed**  
Succeeded

 **Quality Gate passed**  
Succeeded

×

 **Online Build Service (delanl)**            <img alt="More options icon" data-bbox="6211 39

# Review Agent in Action

The screenshot shows a web browser window with the URL [dev.azure.com/delanl/PipelineToolbox/\\_git/pt-pr-ai-bot](https://dev.azure.com/delanl/PipelineToolbox/_git/pt-pr-ai-bot). The browser interface includes a header with back, forward, and search buttons, and a sidebar with various icons.

**Contents**

Name	Last change	Commits
pr-ai-bot-template.yml	Tuesday	<a href="#">fc7b82cb</a> Updated pr-ai-bot-template.yml Pascal Dufour
pr-ai-bot.py	Friday	<a href="#">adac301e</a> Another fix for the repo path Danny Scheelings
README.md	Tuesday	<a href="#">0ce20956</a> Readme updated. Repo alias in template changed Danny Scheelings
requirements.txt	Friday	<a href="#">842bf833</a> 1st version of PR AI bot script Danny Scheelings

**PR AI bot for Azure DevOps**

This repository contains a reusable Azure DevOps pipeline that automatically performs an AI-powered code review on each pull request. It uses OpenAI (via Azure OpenAI) to generate a concise review and summary based on the diff.

**Features**

- Inspects changed files in the PR
- Performs AI review using OpenAI
- Automatically posts a summary and/or review comment to the PR
- Reusable across multiple repositories via pipeline templates

**Structure**

```
ai-pr-bot/
├── pr-ai-bot-template.yml  # Reusable pipeline template
├── pr-ai-bot.py            # Python script to invoke AI
└── requirements.txt        # Python libraries
    └── README.md
```

**How to use in another repository**

**1. Define your PR AI bot variables**

```
variables:
  AI_SYSTEM_PROMPT_SUMMARY: |
    You are an AI assistant that summarizes code changes in a pull request.
    Provide a concise summary, using bullet points, of the overall changes and their impact, but keep it short.
  AI_SYSTEM_PROMPT REVIEW: |
    Your task is to act as a code reviewer of a Pull Request
  PREVENT_DUPLICATE_COMMENTS: true
  SCRIPT_OPTION: ''
```

**Online** + Overview Files Updates Commits Conflicts

Online Build Service (delanl) 4h ago

Azure OpenAI Bot - Summary

- Added Accessibility Test:** Introduced `accessibilityDashboard.spec.ts` for dashboard accessibility validation using Axe, ensuring no serious or critical violations.
- Refactored Code:** Removed redundant accessibility checks in `dashboardDup.spec.ts` and updated `dashboardPage.ts` for improved selector handling.
- UI/UX Enhancements:** Updated styles in `MijnDelaHeader.module.scss` and replaced icons with `EditIcon` for consistency. Improved button and navigation styles.
- Design System Updates:** Enhanced `OutlinedCard` and `KeyValueList` components with additional props and modernized styles.
- Dependency Updates:** Integrated new design system styles and components for uniformity.
- Bug Fixes:** Addressed minor issues like `autoComplete` and improved semantic HTML usage.

Write a reply... Reactivate

Online Build Service (delanl) 4h ago

Azure OpenAI Bot - Review

File: `apps/mijn-dela-e2e/src/e2e/pre-deploy/dashboard/accessibilityDashboard.spec.ts`

- Bug:** `loginPage().setAccount` assumes `loginPage` returns a singleton. Ensure it handles state correctly.
- Bug:** `dashboardPage(page).visit` and `dashboardPage(page).verifyVisit` are invoked twice. Optimize reuse.
- Performance:** `new AxeBuilder({ page }).analyze()` could be slow. Consider caching or batching scans.

File: `apps/mijn-dela-e2e/src/e2e/pre-deploy/dashboard/dashboardDup.spec.ts`

- Accessibility test removal may lead to undetected issues; consider retaining or relocating it.

File: `apps/mijn-dela-e2e/src/pages/dashboardPage.ts`

- The change to `aria-labelledby^="Polisnummer"` introduces a starts-with selector. Ensure it doesn't unintentionally match multiple elements or cause false positives.

File: `apps/mijn-dela/components/layouts/MijnDelaHeader.module.scss`

- `$section-elevation` and `var(--dds-background-page)` usage may cause inconsistent styling across themes.
- Removing `font-size: 0.875em` from `.mijnDelaHeader__icon--edit` might break visual consistency.
- Ensure `var(--dds-typography-modest)` and other variables are defined and compatible with `$dela-blue--1_0`.

File: `apps/mijn-dela/components/layouts/MijnDelaHeader.tsx`

- `EditIcon` usage lacks `aria-hidden` where appropriate; may impact accessibility.
- Inline `sx` styles for `Link` could cause performance issues; consider moving to CSS.
- `window.origin` usage might fail in older browsers; ensure compatibility or fallback.
- `nav` element wrapping tabs is correct but ensure proper ARIA roles for accessibility.

File: `libs/design-system/src/components/cards/outlinedCard/OutlinedCard.tsx`

- component prop default value:** Ensure the `component` prop default ( `'section'` ) doesn't conflict with accessibility or styling expectations.
- Dynamic sx prop for CardContent:** Validate `contentBackgroundColor` to prevent invalid CSS values or runtime errors.



# Development Company Deep Dive

# Review Agent Implementation

**Automatic Copilot code review**

Copilot will automatically review your pull requests. [Learn more.](#)

**Select an option ▾**

**Enabled**  
Copilot automatically reviews pull requests you authored from your repositories.

**No policy**  
Copilot won't enforce automated reviews.

# Review Agent Implementation

## Require a pull request before merging

Require all commits be made to a non-target branch and submitted via a pull request before they can be merged.

Hide additional settings ^

## Required approvals

0 

The number of approving reviews that are required before a pull request can be merged.

### Dismiss stale pull request approvals when new commits are pushed

New, reviewable commits pushed will dismiss previous pull request review approvals.

### Require review from specific teams

A collection of reviewers and associated file patterns. Each reviewer has a list of file patterns which determine the files that reviewer is required to review.

### Require review from Code Owners

Require an approving review in pull requests that modify files that have a designated code owner.

### Require approval of the most recent reviewable push

Whether the most recent reviewable push must be approved by someone other than the person who pushed it.

### Require conversation resolution before merging

All conversations on code must be resolved before a pull request can be merged.

### Request pull request review from Copilot

Automatically request review from Copilot for new pull requests, if the author has access to Copilot code review.

## Allowed merge methods

Merge, Squash, Rebase 

When merging pull requests, you can allow any combination of merge commits, squashing, or rebasing. At least one option must be enabled.

QaS

github.com

rb2-bv / MCS-Commerce

Type to search

Code Issues Pull requests Actions Projects Wiki Security Insights

# fix: Fix cart quantity race condition with simplified debounce #1093

Merged boydenhartog merged 2 commits into `main` from `fix-cart-quantity` on Jul 7

Conversation 7 Commits 2 Checks 5 Files changed 1 +42 -30

boydenhartog commented on Jul 7

## Summary

- Fixed race condition in cart quantity updates that caused incorrect values when rapidly clicking
- Simplified state management from complex ref-based approach to cleaner React patterns
- Ensured only the final quantity update is sent to server when making rapid changes

## The Problem

When rapidly updating cart quantities (e.g., clicking from 55 to 50), the final value would be incorrect (e.g., 51) due to:

- Multiple debounced calls executing with intermediate values
- Race conditions between local state updates and server responses
- Complex state synchronization logic causing circular dependencies

## The Solution

Implemented a cleaner approach that:

- Uses standard React `useState` for immediate UI updates
- Leverages existing `debounce` utility (no manual timeout management)
- Tracks server quantity separately to prevent overwriting user input

Reviewers: cursor[bot], +1 more reviewer, gemini-code-assist[bot]

Assignees: No one assigned

Labels: None yet

Projects: None yet

Milestone: No milestone

Development: Successfully merging this pull request may close these issues.

fix: Fix cart quantity race condition with simplified debounce #1093  
boydenhartog merged 2 commits into [main](#) from [fix-cart-quant...](#) on Jul 7

**Merged**

gemini-code-assist (bot) reviewed on Jul 7 [View reviewed changes](#)

gemini-code-assist (bot) left a comment

## Summary of Changes

Hello [@boydenhartog](#), I'm Gemini Code Assist!<sup>[1]</sup>! I'm currently reviewing this pull request and will post my feedback shortly. In the meantime, here's a summary to help you and other reviewers quickly get up to speed!

This pull request addresses a critical race condition in the cart quantity update functionality by refactoring the `EditItemQuantity` component. The primary goal is to ensure accurate and reliable quantity updates, even during rapid user interactions, by simplifying state management and improving the debouncing mechanism for server calls. This leads to a more robust and user-friendly experience for managing cart items.

## Highlights

- Race Condition Fix:** This pull request resolves a race condition in the cart quantity update mechanism that previously led to incorrect final values when users rapidly adjusted item quantities. The fix ensures that only the final, intended quantity is sent to the server.
- Simplified State Management:** The component's state management has been significantly simplified. It transitions from a complex ref-based approach (removing `isUpdatingRef`, `timeoutRef`, and `pendingQuantityRef` as mentioned in the PR description) to a cleaner pattern utilizing React's `useState` for immediate UI updates and a single `useRef` (`lastServerQuantity`) to track the server's last confirmed quantity.
- Streamlined Debouncing:** The custom or implicitly managed debounce logic has been replaced with a more explicit and cleaner use of an existing `debounce` utility. This ensures that server updates for quantity changes are properly debounced, preventing excessive network requests and race conditions.
- Improved UI Responsiveness:** By introducing `localQuantity` managed by `useState`, the UI now provides immediate visual feedback to the user as they change quantities, while the actual server update is handled asynchronously via the debounced call.
- Code Reduction and Clarity:** The changes result in a more concise and readable component, reducing the line count from approximately 100 to 80 lines, and establishing a clearer separation between UI state and server-synced state.

› Using Gemini Code Assist

fix: Fix cart quantity race condition with simplified debounce #1093  
boydenhartog merged 2 commits into `main` from `fix-cart-quant...` on Jul 7

**Merged**

**Code Review**

The pull request simplifies cart quantity updates and resolves a race condition. However, error handling in `debouncedUpdate` is missing, and the input field's value should consistently reflect the local state. Addressing these points will improve robustness and user experience.

`frontend/components/client/EditItemQuantity/EditItemQuantity.tsx`

Comment on lines `+53` to `58`

```
53  +  const debouncedUpdate = useCallback(
54  +    debounce((quantity: string) => {
55  +      startTransition(async () => {
56  +        const numericQuantity = quantity ? Number(quantity) : 1
57  +        await performUpdate(numericQuantity)
58  +      })
59  +    })
60  +  )
```

**gemini-code-assist** (bot) on Jul 7

**High Priority**

The `debouncedUpdate` function doesn't handle potential errors from `performUpdate`. If `performUpdate` fails (e.g., due to a network error), the UI won't reflect the actual cart state, violating the style guide's rule on graceful handling of network failures.<sup>[1]</sup> Additionally, when the input field is cleared, the quantity defaults to 1 instead of 0, which is not the desired behavior.

```
debounce((quantity: string) => {
  startTransition(async () => {
    const numericQuantity = quantity ? Number(quantity) : 0;
    const res = await performUpdate(numericQuantity);
```

fix: Fix cart quantity race condition with simplified debounce #1093  
boydenhartog merged 2 commits into `main` from `fix-cart-quantity` on Jul 7

**Merged**

 cursor bot reviewed on Jul 7 [View reviewed changes](#)

cursor bot left a comment

• **Bug: API Response Validation and State Tracking Flaws**

The `EditItemQuantity` component can display an incorrect quantity due to incomplete API response validation and stale state tracking. The `performUpdate` function only checks for `res?.redirectUrl` and assumes success, failing to validate `res?.success`. If the `updateItemQuantity` API returns an unsuccessful response without a redirect or error, the local UI state (`localQuantity`) becomes inconsistent with the server. Furthermore, `lastServerQuantity.current` is not updated after successful server operations, which prevents the `useEffect` from correctly detecting external quantity changes. This can lead to `localQuantity` being overwritten with stale `defaultQuantity` values, causing user input loss, incorrect UI display, and flawed error recovery.

▶ [frontend/components/client/EditItemQuantity/EditItemQuantity.tsx#L37-L56](#)

[Fix in Cursor](#) • [Fix in Web](#)

---

Was this report helpful? Give feedback by reacting with  or 



 boydenhartog merged commit `bca6e25` into `main` on Jul 7 [View details](#)  
3 checks passed

 boydenhartog deleted the `fix-cart-quantity` branch 2 months ago

fix: Fix cart quantity race condition with simplified debounce #1093  
boydenhartog merged 2 commits into [main](#) from [fix-cart-quantity](#) on Jul 7

**• Bug: API Response Validation and State Tracking Flaws**

The `EditItemQuantity` component can display an incorrect quantity due to incomplete API response validation and stale state tracking. The `performUpdate` function only checks for `res?.redirectUrl` and assumes success, failing to validate `res?.success`. If the `updateItemQuantity` API returns an unsuccessful response without a redirect or error, the local UI state (`localQuantity`) becomes inconsistent with the server. Furthermore, `lastServerQuantity.current` is not updated after successful server operations, which prevents the `useEffect` from correctly detecting external quantity changes. This can lead to `localQuantity` being overwritten with stale `defaultQuantity` values, causing user input loss, incorrect UI display, and flawed error recovery.

▶ [frontend/components/client/EditItemQuantity/EditItemQuantity.tsx#L37-L56](#)

[Fix in Cursor](#) • [Fix in Web](#)

---

Was this report helpful? Give feedback by reacting with  or 



---

  **boydenhartog** merged commit `bca6e25` into [main](#) on Jul 7  
3 checks passed

  **boydenhartog** deleted the `fix-cart-quantity` branch [2 months ago](#)

[View details](#)

---

**Pull request successfully merged and closed**

You're all set — the branch has been merged.

---

 [Add a comment](#)



## CHANGES

Message (⌘ ↲ to com...)



✓ Commit

10

## Changes

- ⬇️ ARCHITECTURE.md U
- {} settings.json .vscode U
- ⬇️ test.zip st-tst-testapp-front U
- ↔️ new.component.html st-t... M
- TS CRMLogin.spec.ts st-tst-t... D

## AGENT REVIEW

Reviewing

Found 1 Potential Issue (Diff with...

↔️ ...es/new/new.component.html  
Form field name changed, ×  
breaking data binding and  
tests

Fix

## ts SystemTeamTestSite.spec.ts M

st-tst-testapp-front &gt; test &gt; pages

```
1 import type { Page } from 'playwright';
2 import { expect } from 'chai';
3 import { HomePage } from './page-objects/home-page';
4
5 export const newPage = async () => {
6   const BASE_URL = 'http://localhost:4200';
7   //Page locators
8   const listPageButton = '#listPage';
9   const pageTitle = '#pageTitle';
10  const usernameTextBox = '#username';
11  const addressTextBox = '#address';
12  const countryTextBox = '#country';
13  const informationTextBox = '#information';
14  const submitButton = '#submit';
15
16  //Page functions
17  const goto = async (url) => {
18    await page.goto(url);
19    await expect(page).to.be.title('CRM Test Site');
20  };
21
22  //Page objects
23  class HomePage {
24    constructor(page) {
25      this.page = page;
26    }
27
28    async login(username, password) {
29      await this.page.fill('#username', username);
30      await this.page.fill('#password', password);
31      await this.page.click('#submit');
32    }
33
34    async verifyLogin() {
35      const title = await this.page.title();
36      expect(title).to.equal('CRM Test Site');
37    }
38  }
39
40  //Page objects
41  class LoginPage extends HomePage {
42    constructor(page) {
43      super(page);
44    }
45
46    async goto() {
47      await this.page.goto(BASE_URL);
48    }
49
50    async verifyPage() {
51      const title = await this.page.title();
52      expect(title).to.equal('CRM Test Site');
53    }
54  }
55
56  //Page objects
57  class ListPage extends HomePage {
58    constructor(page) {
59      super(page);
60    }
61
62    async goto() {
63      await this.page.goto(`${BASE_URL}/list`);
64    }
65
66    async verifyPage() {
67      const title = await this.page.title();
68      expect(title).to.equal('CRM Test Site - List');
69    }
70  }
71
72  //Page objects
73  class AddPage extends HomePage {
74    constructor(page) {
75      super(page);
76    }
77
78    async goto() {
79      await this.page.goto(`${BASE_URL}/add`);
80    }
81
82    async verifyPage() {
83      const title = await this.page.title();
84      expect(title).to.equal('CRM Test Site - Add');
85    }
86  }
87
88  //Page objects
89  class EditPage extends HomePage {
90    constructor(page) {
91      super(page);
92    }
93
94    async goto(id) {
95      await this.page.goto(`${BASE_URL}/edit/${id}`);
96    }
97
98    async verifyPage() {
99      const title = await this.page.title();
100     expect(title).to.equal('CRM Test Site - Edit');
101   }
102 }
```

Problems

Output

Debug Console

pascaldufour@Pascals-MacBook-P



## CHANGES

Message (⌘ ↲ to com...)

✓ Commit

## Changes

10

- ⬇️ ARCHITECTURE.md U
- {} settings.json .vscode U
- ⬇️ test.zip st-tst-testapp-front U
- ↔️ new.component.html st-t... M
- TS CRMLogin.spec.ts st-tst-t... D

## AGENT REVIEW

Reviewing

Found 1 Potential Issue (Diff with...

↔️ ...es/new/new.component.html

Form field name changed, ×  
breaking data binding and  
tests

Fix

## ts SystemTeamTestSite.spec.ts M

st-tst-testapp-front &gt; test &gt; pages

```
1 import type { Page } from 'playwright';
2 import { expect } from 'chai';
3 import { HomePage } from './page-objects/home-page';
4
5 export const newPage = async () => {
6   const BASE_URL = 'http://localhost:4200';
7   //Page locators
8   const listPageButton = '#listPage';
9   const pageTitle = '#pageTitle';
10  const usernameTextBox = '#username';
11  const addressTextBox = '#address';
12  const countryTextBox = '#country';
13  const informationTextBox = '#information';
14  const submitButton = '#submit';
15
16  //Page functions
17  const goto = async (url) => {
18    await page.goto(url);
19    await expect(page).to.be.title('CRM Test Site');
20  };
21
22  //Page objects
23  class HomePage {
24    constructor(page) {
25      this.page = page;
26    }
27
28    async login(username, password) {
29      await this.page.fill('#username', username);
30      await this.page.fill('#password', password);
31      await this.page.click('#submit');
32    }
33
34    async verifyLogin() {
35      const title = await this.page.title();
36      expect(title).to.equal('CRM Test Site');
37    }
38  }
39
40  //Page objects
41  class LoginPage extends HomePage {
42    constructor(page) {
43      super(page);
44    }
45
46    async goto() {
47      await this.page.goto(BASE_URL);
48    }
49
50    async verifyPage() {
51      const title = await this.page.title();
52      expect(title).to.equal('CRM Test Site');
53    }
54  }
55
56  //Page objects
57  class ListPage extends LoginPage {
58    constructor(page) {
59      super(page);
60    }
61
62    async verifyPage() {
63      const title = await this.page.title();
64      expect(title).to.equal('CRM Test Site');
65    }
66  }
67
68  //Page objects
69  class LoginPage extends HomePage {
70    constructor(page) {
71      super(page);
72    }
73
74    async goto() {
75      await this.page.goto(BASE_URL);
76    }
77
78    async verifyPage() {
79      const title = await this.page.title();
80      expect(title).to.equal('CRM Test Site');
81    }
82  }
83
84  //Page objects
85  class ListPage extends LoginPage {
86    constructor(page) {
87      super(page);
88    }
89
90    async verifyPage() {
91      const title = await this.page.title();
92      expect(title).to.equal('CRM Test Site');
93    }
94  }
95
96  //Page objects
97  class LoginPage extends HomePage {
98    constructor(page) {
99      super(page);
100     this.page = page;
101   }
102
103   async goto() {
104     await this.page.goto(BASE_URL);
105   }
106
107   async verifyPage() {
108     const title = await this.page.title();
109     expect(title).to.equal('CRM Test Site');
110   }
111 }
112
113 //Page objects
114 class ListPage extends LoginPage {
115   constructor(page) {
116     super(page);
117   }
118
119   async verifyPage() {
120     const title = await this.page.title();
121     expect(title).to.equal('CRM Test Site');
122   }
123 }
124
125 //Page objects
126 class LoginPage extends HomePage {
127   constructor(page) {
128     super(page);
129   }
130
131   async goto() {
132     await this.page.goto(BASE_URL);
133   }
134
135   async verifyPage() {
136     const title = await this.page.title();
137     expect(title).to.equal('CRM Test Site');
138   }
139 }
140
141 //Page objects
142 class ListPage extends LoginPage {
143   constructor(page) {
144     super(page);
145   }
146
147   async verifyPage() {
148     const title = await this.page.title();
149     expect(title).to.equal('CRM Test Site');
150   }
151 }
152
153 //Page objects
154 class LoginPage extends HomePage {
155   constructor(page) {
156     super(page);
157   }
158
159   async goto() {
160     await this.page.goto(BASE_URL);
161   }
162
163   async verifyPage() {
164     const title = await this.page.title();
165     expect(title).to.equal('CRM Test Site');
166   }
167 }
168
169 //Page objects
170 class ListPage extends LoginPage {
171   constructor(page) {
172     super(page);
173   }
174
175   async verifyPage() {
176     const title = await this.page.title();
177     expect(title).to.equal('CRM Test Site');
178   }
179 }
180
181 //Page objects
182 class LoginPage extends HomePage {
183   constructor(page) {
184     super(page);
185   }
186
187   async goto() {
188     await this.page.goto(BASE_URL);
189   }
190
191   async verifyPage() {
192     const title = await this.page.title();
193     expect(title).to.equal('CRM Test Site');
194   }
195 }
196
197 //Page objects
198 class ListPage extends LoginPage {
199   constructor(page) {
200     super(page);
201   }
202
203   async verifyPage() {
204     const title = await this.page.title();
205     expect(title).to.equal('CRM Test Site');
206   }
207 }
208
209 //Page objects
210 class LoginPage extends HomePage {
211   constructor(page) {
212     super(page);
213   }
214
215   async goto() {
216     await this.page.goto(BASE_URL);
217   }
218
219   async verifyPage() {
220     const title = await this.page.title();
221     expect(title).to.equal('CRM Test Site');
222   }
223 }
224
225 //Page objects
226 class ListPage extends LoginPage {
227   constructor(page) {
228     super(page);
229   }
230
231   async verifyPage() {
232     const title = await this.page.title();
233     expect(title).to.equal('CRM Test Site');
234   }
235 }
236
237 //Page objects
238 class LoginPage extends HomePage {
239   constructor(page) {
240     super(page);
241   }
242
243   async goto() {
244     await this.page.goto(BASE_URL);
245   }
246
247   async verifyPage() {
248     const title = await this.page.title();
249     expect(title).to.equal('CRM Test Site');
250   }
251 }
252
253 //Page objects
254 class ListPage extends LoginPage {
255   constructor(page) {
256     super(page);
257   }
258
259   async verifyPage() {
260     const title = await this.page.title();
261     expect(title).to.equal('CRM Test Site');
262   }
263 }
264
265 //Page objects
266 class LoginPage extends HomePage {
267   constructor(page) {
268     super(page);
269   }
270
271   async goto() {
272     await this.page.goto(BASE_URL);
273   }
274
275   async verifyPage() {
276     const title = await this.page.title();
277     expect(title).to.equal('CRM Test Site');
278   }
279 }
280
281 //Page objects
282 class ListPage extends LoginPage {
283   constructor(page) {
284     super(page);
285   }
286
287   async verifyPage() {
288     const title = await this.page.title();
289     expect(title).to.equal('CRM Test Site');
290   }
291 }
292
293 //Page objects
294 class LoginPage extends HomePage {
295   constructor(page) {
296     super(page);
297   }
298
299   async goto() {
300     await this.page.goto(BASE_URL);
301   }
302
303   async verifyPage() {
304     const title = await this.page.title();
305     expect(title).to.equal('CRM Test Site');
306   }
307 }
308
309 //Page objects
310 class ListPage extends LoginPage {
311   constructor(page) {
312     super(page);
313   }
314
315   async verifyPage() {
316     const title = await this.page.title();
317     expect(title).to.equal('CRM Test Site');
318   }
319 }
320
321 //Page objects
322 class LoginPage extends HomePage {
323   constructor(page) {
324     super(page);
325   }
326
327   async goto() {
328     await this.page.goto(BASE_URL);
329   }
330
331   async verifyPage() {
332     const title = await this.page.title();
333     expect(title).to.equal('CRM Test Site');
334   }
335 }
336
337 //Page objects
338 class ListPage extends LoginPage {
339   constructor(page) {
340     super(page);
341   }
342
343   async verifyPage() {
344     const title = await this.page.title();
345     expect(title).to.equal('CRM Test Site');
346   }
347 }
348
349 //Page objects
350 class LoginPage extends HomePage {
351   constructor(page) {
352     super(page);
353   }
354
355   async goto() {
356     await this.page.goto(BASE_URL);
357   }
358
359   async verifyPage() {
360     const title = await this.page.title();
361     expect(title).to.equal('CRM Test Site');
362   }
363 }
364
365 //Page objects
366 class ListPage extends LoginPage {
367   constructor(page) {
368     super(page);
369   }
370
371   async verifyPage() {
372     const title = await this.page.title();
373     expect(title).to.equal('CRM Test Site');
374   }
375 }
376
377 //Page objects
378 class LoginPage extends HomePage {
379   constructor(page) {
380     super(page);
381   }
382
383   async goto() {
384     await this.page.goto(BASE_URL);
385   }
386
387   async verifyPage() {
388     const title = await this.page.title();
389     expect(title).to.equal('CRM Test Site');
390   }
391 }
392
393 //Page objects
394 class ListPage extends LoginPage {
395   constructor(page) {
396     super(page);
397   }
398
399   async verifyPage() {
400     const title = await this.page.title();
401     expect(title).to.equal('CRM Test Site');
402   }
403 }
404
405 //Page objects
406 class LoginPage extends HomePage {
407   constructor(page) {
408     super(page);
409   }
410
411   async goto() {
412     await this.page.goto(BASE_URL);
413   }
414
415   async verifyPage() {
416     const title = await this.page.title();
417     expect(title).to.equal('CRM Test Site');
418   }
419 }
420
421 //Page objects
422 class ListPage extends LoginPage {
423   constructor(page) {
424     super(page);
425   }
426
427   async verifyPage() {
428     const title = await this.page.title();
429     expect(title).to.equal('CRM Test Site');
430   }
431 }
432
433 //Page objects
434 class LoginPage extends HomePage {
435   constructor(page) {
436     super(page);
437   }
438
439   async goto() {
440     await this.page.goto(BASE_URL);
441   }
442
443   async verifyPage() {
444     const title = await this.page.title();
445     expect(title).to.equal('CRM Test Site');
446   }
447 }
448
449 //Page objects
450 class ListPage extends LoginPage {
451   constructor(page) {
452     super(page);
453   }
454
455   async verifyPage() {
456     const title = await this.page.title();
457     expect(title).to.equal('CRM Test Site');
458   }
459 }
460
461 //Page objects
462 class LoginPage extends HomePage {
463   constructor(page) {
464     super(page);
465   }
466
467   async goto() {
468     await this.page.goto(BASE_URL);
469   }
470
471   async verifyPage() {
472     const title = await this.page.title();
473     expect(title).to.equal('CRM Test Site');
474   }
475 }
476
477 //Page objects
478 class ListPage extends LoginPage {
479   constructor(page) {
480     super(page);
481   }
482
483   async verifyPage() {
484     const title = await this.page.title();
485     expect(title).to.equal('CRM Test Site');
486   }
487 }
488
489 //Page objects
490 class LoginPage extends HomePage {
491   constructor(page) {
492     super(page);
493   }
494
495   async goto() {
496     await this.page.goto(BASE_URL);
497   }
498
499   async verifyPage() {
500     const title = await this.page.title();
501     expect(title).to.equal('CRM Test Site');
502   }
503 }
504
505 //Page objects
506 class ListPage extends LoginPage {
507   constructor(page) {
508     super(page);
509   }
510
511   async verifyPage() {
512     const title = await this.page.title();
513     expect(title).to.equal('CRM Test Site');
514   }
515 }
516
517 //Page objects
518 class LoginPage extends HomePage {
519   constructor(page) {
520     super(page);
521   }
522
523   async goto() {
524     await this.page.goto(BASE_URL);
525   }
526
527   async verifyPage() {
528     const title = await this.page.title();
529     expect(title).to.equal('CRM Test Site');
530   }
531 }
532
533 //Page objects
534 class ListPage extends LoginPage {
535   constructor(page) {
536     super(page);
537   }
538
539   async verifyPage() {
540     const title = await this.page.title();
541     expect(title).to.equal('CRM Test Site');
542   }
543 }
544
545 //Page objects
546 class LoginPage extends HomePage {
547   constructor(page) {
548     super(page);
549   }
550
551   async goto() {
552     await this.page.goto(BASE_URL);
553   }
554
555   async verifyPage() {
556     const title = await this.page.title();
557     expect(title).to.equal('CRM Test Site');
558   }
559 }
560
561 //Page objects
562 class ListPage extends LoginPage {
563   constructor(page) {
564     super(page);
565   }
566
567   async verifyPage() {
568     const title = await this.page.title();
569     expect(title).to.equal('CRM Test Site');
570   }
571 }
572
573 //Page objects
574 class LoginPage extends HomePage {
575   constructor(page) {
576     super(page);
577   }
578
579   async goto() {
580     await this.page.goto(BASE_URL);
581   }
582
583   async verifyPage() {
584     const title = await this.page.title();
585     expect(title).to.equal('CRM Test Site');
586   }
587 }
588
589 //Page objects
590 class ListPage extends LoginPage {
591   constructor(page) {
592     super(page);
593   }
594
595   async verifyPage() {
596     const title = await this.page.title();
597     expect(title).to.equal('CRM Test Site');
598   }
599 }
599
600 //Page objects
601 class LoginPage extends HomePage {
602   constructor(page) {
603     super(page);
604   }
605
606   async goto() {
607     await this.page.goto(BASE_URL);
608   }
609
610   async verifyPage() {
611     const title = await this.page.title();
612     expect(title).to.equal('CRM Test Site');
613   }
614 }
615
616 //Page objects
617 class ListPage extends LoginPage {
618   constructor(page) {
619     super(page);
620   }
621
622   async verifyPage() {
623     const title = await this.page.title();
624     expect(title).to.equal('CRM Test Site');
625   }
626 }
626
627 //Page objects
628 class LoginPage extends HomePage {
629   constructor(page) {
630     super(page);
631   }
632
633   async goto() {
634     await this.page.goto(BASE_URL);
635   }
636
637   async verifyPage() {
638     const title = await this.page.title();
639     expect(title).to.equal('CRM Test Site');
640   }
641 }
642
643 //Page objects
644 class ListPage extends LoginPage {
645   constructor(page) {
646     super(page);
647   }
648
649   async verifyPage() {
650     const title = await this.page.title();
651     expect(title).to.equal('CRM Test Site');
652   }
653 }
653
654 //Page objects
655 class LoginPage extends HomePage {
656   constructor(page) {
657     super(page);
658   }
659
660   async goto() {
661     await this.page.goto(BASE_URL);
662   }
663
664   async verifyPage() {
665     const title = await this.page.title();
666     expect(title).to.equal('CRM Test Site');
667   }
668 }
669
670 //Page objects
671 class ListPage extends LoginPage {
672   constructor(page) {
673     super(page);
674   }
675
676   async verifyPage() {
677     const title = await this.page.title();
678     expect(title).to.equal('CRM Test Site');
679   }
680 }
680
681 //Page objects
682 class LoginPage extends HomePage {
683   constructor(page) {
684     super(page);
685   }
686
687   async goto() {
688     await this.page.goto(BASE_URL);
689   }
690
691   async verifyPage() {
692     const title = await this.page.title();
693     expect(title).to.equal('CRM Test Site');
694   }
695 }
696
697 //Page objects
698 class ListPage extends LoginPage {
699   constructor(page) {
700     super(page);
701   }
702
703   async verifyPage() {
704     const title = await this.page.title();
705     expect(title).to.equal('CRM Test Site');
706   }
707 }
707
708 //Page objects
709 class LoginPage extends HomePage {
710   constructor(page) {
711     super(page);
712   }
713
714   async goto() {
715     await this.page.goto(BASE_URL);
716   }
717
718   async verifyPage() {
719     const title = await this.page.title();
720     expect(title).to.equal('CRM Test Site');
721   }
722 }
723
724 //Page objects
725 class ListPage extends LoginPage {
726   constructor(page) {
727     super(page);
728   }
729
730   async verifyPage() {
731     const title = await this.page.title();
732     expect(title).to.equal('CRM Test Site');
733   }
734 }
734
735 //Page objects
736 class LoginPage extends HomePage {
737   constructor(page) {
738     super(page);
739   }
740
741   async goto() {
742     await this.page.goto(BASE_URL);
743   }
744
745   async verifyPage() {
746     const title = await this.page.title();
747     expect(title).to.equal('CRM Test Site');
748   }
749 }
750
751 //Page objects
752 class ListPage extends LoginPage {
753   constructor(page) {
754     super(page);
755   }
756
757   async verifyPage() {
758     const title = await this.page.title();
759     expect(title).to.equal('CRM Test Site');
760   }
761 }
761
762 //Page objects
763 class LoginPage extends HomePage {
764   constructor(page) {
765     super(page);
766   }
767
768   async goto() {
769     await this.page.goto(BASE_URL);
770   }
771
772   async verifyPage() {
773     const title = await this.page.title();
774     expect(title).to.equal('CRM Test Site');
775   }
776 }
777
778 //Page objects
779 class ListPage extends LoginPage {
780   constructor(page) {
781     super(page);
782   }
783
784   async verifyPage() {
785     const title = await this.page.title();
786     expect(title).to.equal('CRM Test Site');
787   }
788 }
788
789 //Page objects
790 class LoginPage extends HomePage {
791   constructor(page) {
792     super(page);
793   }
794
795   async goto() {
796     await this.page.goto(BASE_URL);
797   }
798
799   async verifyPage() {
800     const title = await this.page.title();
801     expect(title).to.equal('CRM Test Site');
802   }
803 }
804
805 //Page objects
806 class ListPage extends LoginPage {
807   constructor(page) {
808     super(page);
809   }
810
811   async verifyPage() {
812     const title = await this.page.title();
813     expect(title).to.equal('CRM Test Site');
814   }
815 }
815
816 //Page objects
817 class LoginPage extends HomePage {
818   constructor(page) {
819     super(page);
820   }
821
822   async goto() {
823     await this.page.goto(BASE_URL);
824   }
825
826   async verifyPage() {
827     const title = await this.page.title();
828     expect(title).to.equal('CRM Test Site');
829   }
830 }
831
832 //Page objects
833 class ListPage extends LoginPage {
834   constructor(page) {
835     super(page);
836   }
837
838   async verifyPage() {
839     const title = await this.page.title();
840     expect(title).to.equal('CRM Test Site');
841   }
842 }
842
843 //Page objects
844 class LoginPage extends HomePage {
845   constructor(page) {
846     super(page);
847   }
848
849   async goto() {
850     await this.page.goto(BASE_URL);
851   }
852
853   async verifyPage() {
854     const title = await this.page.title();
855     expect(title).to.equal('CRM Test Site');
856   }
857 }
858
859 //Page objects
860 class ListPage extends LoginPage {
861   constructor(page) {
862     super(page);
863   }
864
865   async verifyPage() {
866     const title = await this.page.title();
867     expect(title).to.equal('CRM Test Site');
868   }
869 }
869
870 //Page objects
871 class LoginPage extends HomePage {
872   constructor(page) {
873     super(page);
874   }
875
876   async goto() {
877     await this.page.goto(BASE_URL);
878   }
879
880   async verifyPage() {
881     const title = await this.page.title();
882     expect(title).to.equal('CRM Test Site');
883   }
884 }
885
886 //Page objects
887 class ListPage extends LoginPage {
888   constructor(page) {
889     super(page);
890   }
891
892   async verifyPage() {
893     const title = await this.page.title();
894     expect(title).to.equal('CRM Test Site');
895   }
896 }
896
897 //Page objects
898 class LoginPage extends HomePage {
899   constructor(page) {
900     super(page);
901   }
902
903   async goto() {
904     await this.page.goto(BASE_URL);
905   }
906
907   async verifyPage() {
908     const title = await this.page.title();
909     expect(title).to.equal('CRM Test Site');
910   }
911 }
912
913 //Page objects
914 class ListPage extends LoginPage {
915   constructor(page) {
916     super(page);
917   }
918
919   async verifyPage() {
920     const title = await this.page.title();
921     expect(title).to.equal('CRM Test Site');
922   }
923 }
923
924 //Page objects
925 class LoginPage extends HomePage {
926   constructor(page) {
927     super(page);
928   }
929
930   async goto() {
931     await this.page.goto(BASE_URL);
932   }
933
934   async verifyPage() {
935     const title = await this.page.title();
936     expect(title).to.equal('CRM Test Site');
937   }
938 }
939
940 //Page objects
941 class ListPage extends LoginPage {
942   constructor(page) {
943     super(page);
944   }
945
946   async verifyPage() {
947     const title = await this.page.title();
948     expect(title).to.equal('CRM Test Site');
949   }
950 }
950
951 //Page objects
952 class LoginPage extends HomePage {
953   constructor(page) {
954     super(page);
955   }
956
957   async goto() {
958     await this.page.goto(BASE_URL);
959   }
960
961   async verifyPage() {
962     const title = await this.page.title();
963     expect(title).to.equal('CRM Test Site');
964   }
965 }
966
967 //Page objects
968 class ListPage extends LoginPage {
969   constructor(page) {
970     super(page);
971   }
972
973   async verifyPage() {
974     const title = await this.page.title();
975     expect(title).to.equal('CRM Test Site');
976   }
977 }
977
978 //Page objects
979 class LoginPage extends HomePage {
980   constructor(page) {
981     super(page);
982   }
983
984   async goto() {
985     await this.page.goto(BASE_URL);
986   }
987
988   async verifyPage() {
989     const title = await this.page.title();
990     expect(title).to.equal('CRM Test Site');
991   }
992 }
993
994 //Page objects
995 class ListPage extends LoginPage {
996   constructor(page) {
997     super(page);
998   }
999
1000   async verifyPage() {
1001     const title = await this.page.title();
1002     expect(title).to.equal('CRM Test Site');
1003   }
1004 }
```

Problems Output Debug Console  
pascaldufour@Pascals-MacBook-P

**Creating code** -> creating behavior.  
Proven by verifications



# Comparing Experiment Portfolios 2025

## Development Company

1. Coding assistants
2. Review bot
3. Customer prototyping – v0 and similar rapid prototype tools
4. Claude code generation
5. Collaborative agents – Multiple agents modifying code
6. Dev agents change code and deploy to production
7. Design system integration – MCP Figma
8. Customer workflow agents

## Insurance Company

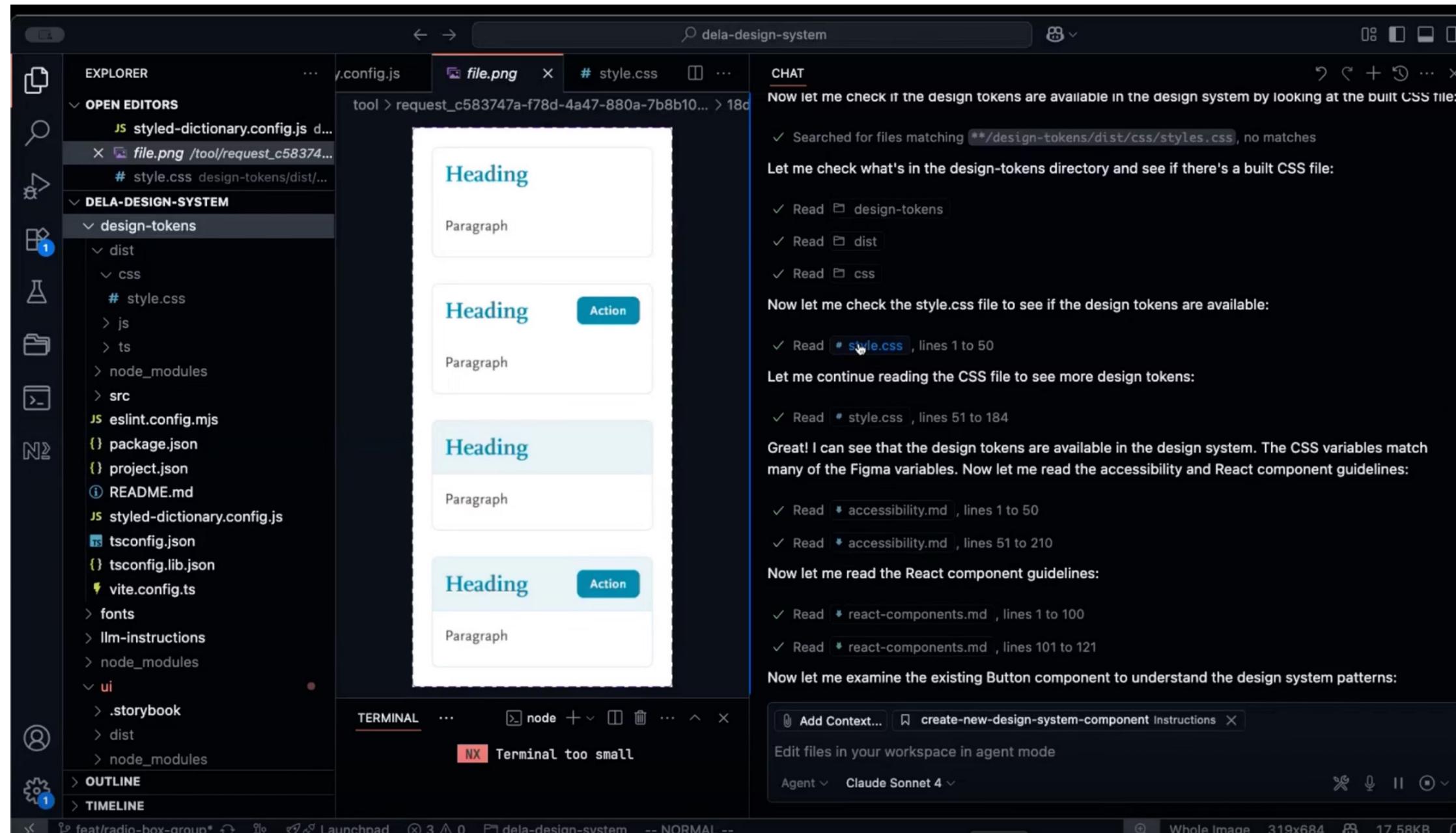
1. Azure Copilot, Microsoft Office Copilot, Copilot Studio, GitHub Copilot
2. Review bot
3. Release notes
4. Vibe coding – Rapid prototype development
5. Document generation
6. Design system – MCP Figma integration
7. VAT checking
8. Epic AI assistant

# Figma Insurance

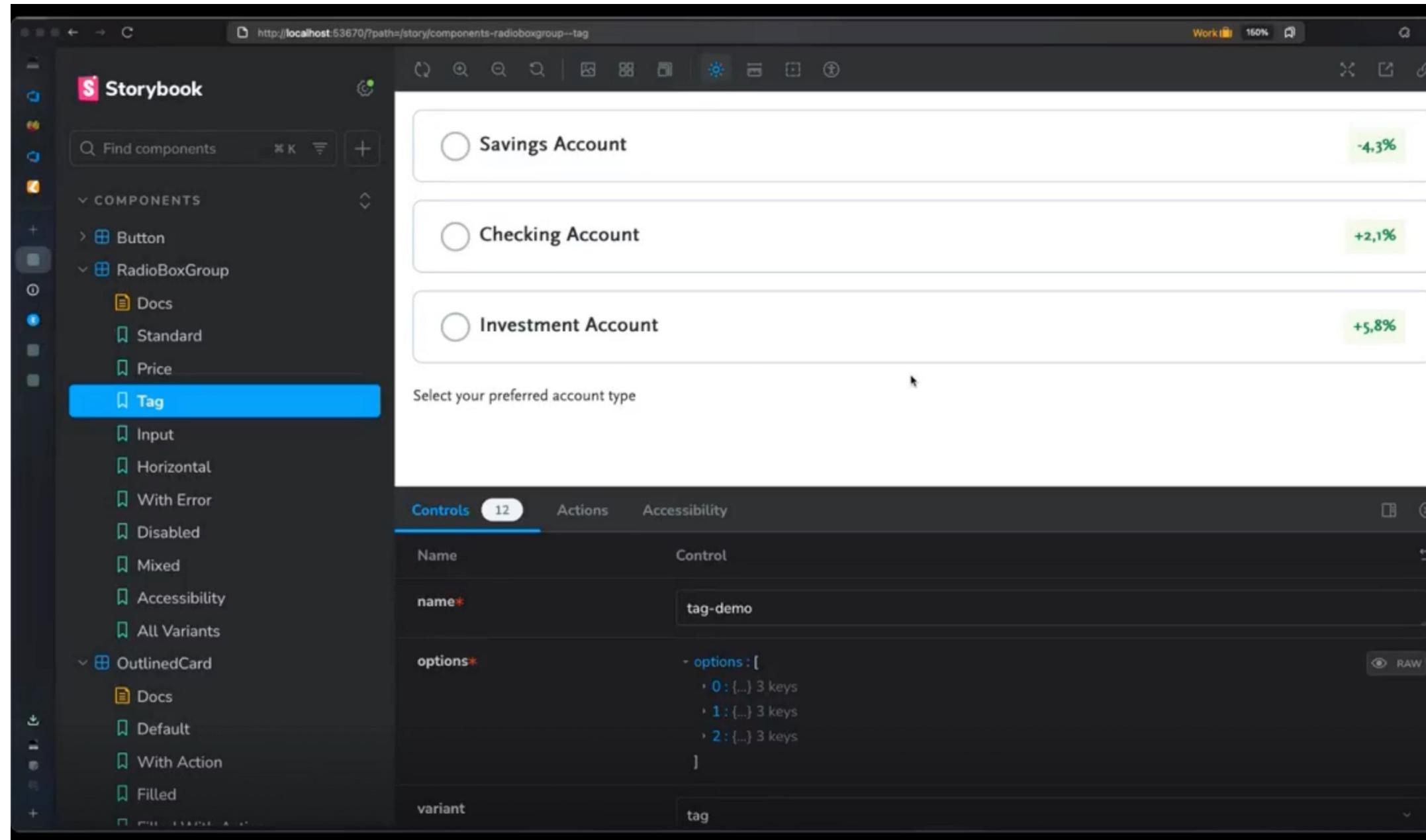
The screenshot displays the Figma interface with the following components:

- Left Sidebar:** Shows the "Base components" section with "Ready for dev" and "Assets" sections, and the "Components" section which includes "Outlined card".
- Central View:** A card titled "Outlined Card" containing:
  - Three green buttons: "Figma", "Frontify", and "Storybook".
  - Text: "Een 'Outlined card' bevat informatie en acties over een enkel onderwerp. De inhoud van de card is flexibel in te richten. Het gebruik van cards zorgt ervoor dat een gebruiker makkelijk informatie kan scannen, lezen en acties kan uitvoeren."
  - Links:
    - [Bekijk de richtlijnen hier](#)
    - [Bekijk in Storybook](#)
- Bottom:** The Figma toolbar with various icons for selection, transform, text, and more.
- Right Sidebar:** The "Outline" panel for the "OutlineCard" component, showing:
  - Design:** Position (X: 24, Y: 24), Rotation (0°), Auto layout, Resizing (W: 319, H: 684), Alignment (Hug), Padding (24), Clip content (checked), Appearance (Opacity 100%, Corner radius 5), Fill (color/purple/100), Stroke (color/purple/100, Inside, Weight 1), Effects, Selection colors.
  - Prototype:** Buttons for preview, play, and test.

# Figma Insurance



# Figma Insurance





## Figma at Development Company

# Experiments to mention

## Development Company Experiments

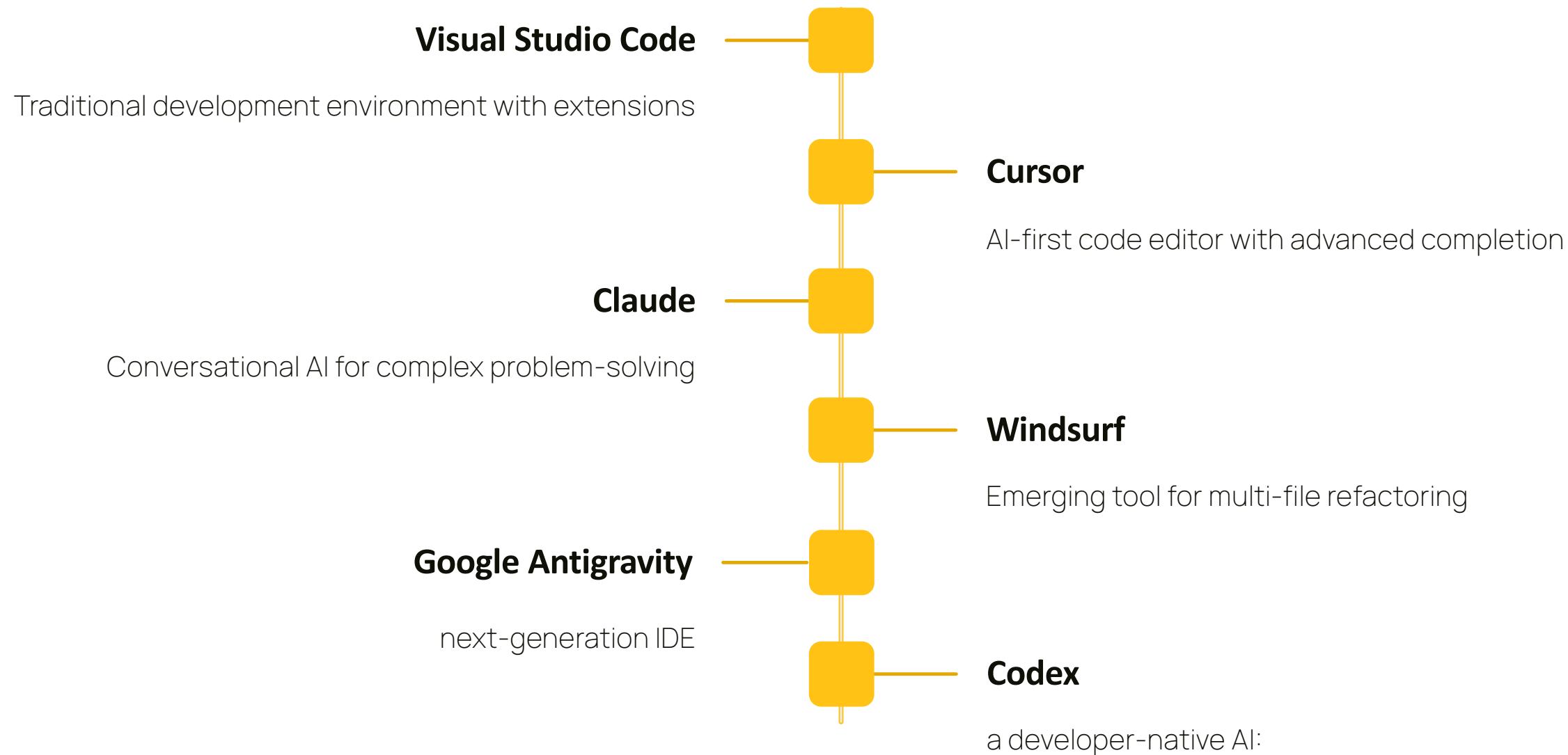
- **Coding assistants**
- Review bot
- Prototyping with customers v0, etc.
- Claude code generation
- Multiple Review agents changing code
- Dev agents change code and deploy to production
- Design system (MCP Figma)
- Creating Agents for customer workflows
- ...

## Insurance Company Experiments

- Azure Copilot, Microsoft Office Copilot, Copilot Studio,
- **GitHub Copilot**
- Review bot
- Release notes
- Vibe coding prototypes
- Document generation
- Design system ( MCP Figma)
- VAT check
- Software testing with AI
- ...

# Tool Evolution at Development Company

## Every "Week" a New Tool



# Insurance Company



## Deliberate Simplicity

- **Visual Studio Code** – Primary development environment
- **With GitHub**

# GitHub Copilot review Hype

When using GitHub CoPilot, I...

[Meer details](#)

Strongly Disagree   Disagree   Neutral   Agree   Strongly Agree

Enjoy coding more



Write better quality code



Commit code suggested by Copilot



Complete tasks faster



Learn from the suggestions Copilot shows me



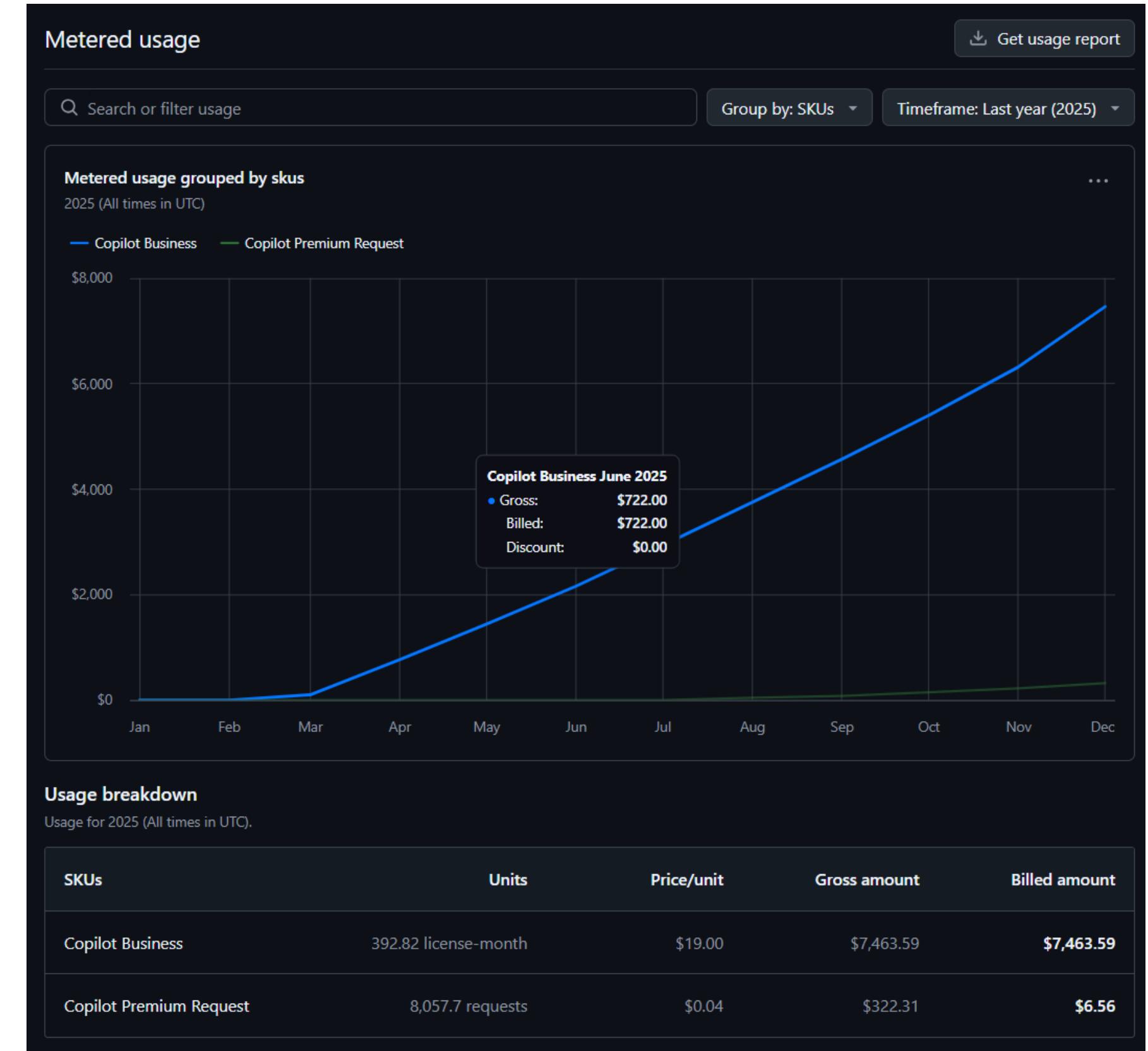
Spend less time searching for information or examples



Spend less mental effort on repetitive tasks

100%   0%   100%

# GitHub Copilot



# GitHub Copilot: Usage Patterns

GitHub Copilot

 GitHub Copilot Business is active for your account

You are assigned a seat as part of a [GitHub Copilot Business](#) subscription managed by [CooperatieDELA](#). Get started by installing the extension in your preferred environment.

[Copilot in your IDE](#) [Copilot in the CLI](#) [Chat in GitHub Mobile](#) [More features](#)

**Usage**

**Premium requests** 9.7%

Please note that there may be a delay in the displayed usage percentage. The premium request entitlement for your plan will reset at the start of next month. You can enable additional premium requests in the [Billing settings](#).

Managed by [CooperatieDELA](#)

**Features**

<b>Editor preview features</b>	 Enabled
You can use preview features in your editor. <a href="#">Learn more</a> .	
<b>Copilot in GitHub.com</b>	 Enabled
You can use Copilot Chat in GitHub.com and Copilot for pull requests. Copilot code review and preview features are only available for paid licenses. <a href="#">Learn more about Copilot in GitHub.com</a> .	
<b>Copilot in the CLI</b>	 Enabled
You can use <a href="#">GitHub Copilot</a> for assistance in terminal.	
<b>Copilot Extensions</b>	 Disabled
GitHub Copilot Extensions are not available for use in your account. You can contact an administrator in <a href="#">CooperatieDELA</a> to request access.	
<b>Copilot in GitHub Desktop</b>	 Disabled
You can use <a href="#">GitHub Copilot</a> for assistance in GitHub Desktop.	
<b>Copilot Chat in the IDE</b>	 Enabled
You can use <a href="#">GitHub Copilot Chat in the IDE</a> to explain a piece of code, propose bug fixes, or generate unit tests in a chat interface from your editor.	
<b>Copilot Chat in GitHub Mobile</b>	 Enabled
You can use <a href="#">Copilot Chat in GitHub Mobile</a> personalized to a codebase.	

**Copilot Usage**

**Code completions** Included

**Chat messages** Included

**Premium requests** 100%

Additional paid premium requests disabled.

Allowance resets August 1, 2025.

**Workspace Index**

Locally indexed

**Code Completions**

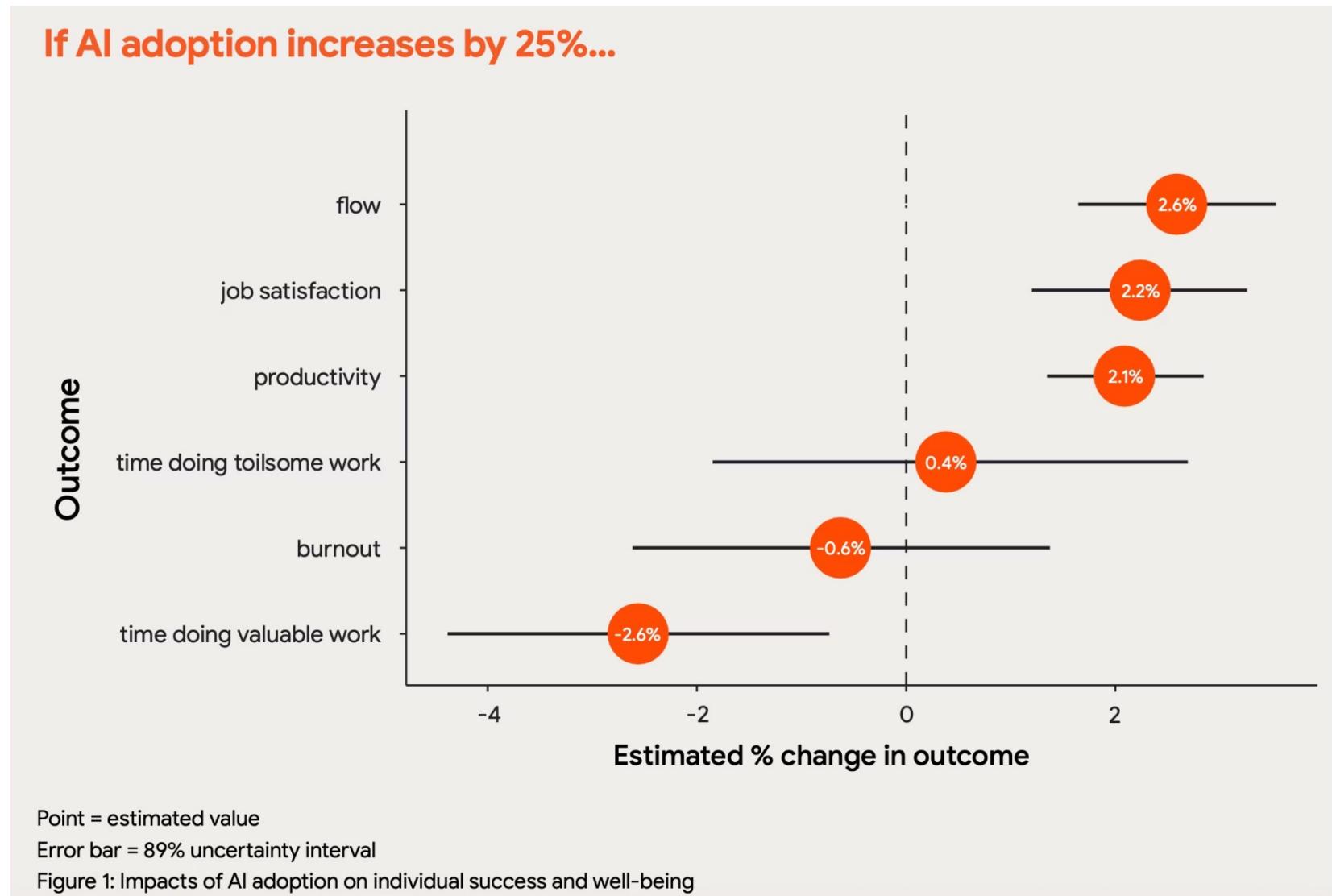
All files 

Next edit suggestions

**Snooze** **Hide completions for 5 min**

# Are We Normal?

## DORA Metrics Comparison (2025)



# Are We Normal?

## McKinsey Research: Superagency in the Workplace

Published January 2026

A BIG NUMBER

40–50%

That's how much faster organizations can [accelerate their timelines for modernizing IT systems, by using gen AI](#) to eliminate much of the company's manual work. McKinsey's Aaron Bawcom and Matt Fitzpatrick and coauthors also estimate that harnessing gen AI can help organizations cut 40 percent of costs derived from their [tech debt](#)—that is, the “tax” a company pays in time and resources needed to solve tech issues.

---

“

---

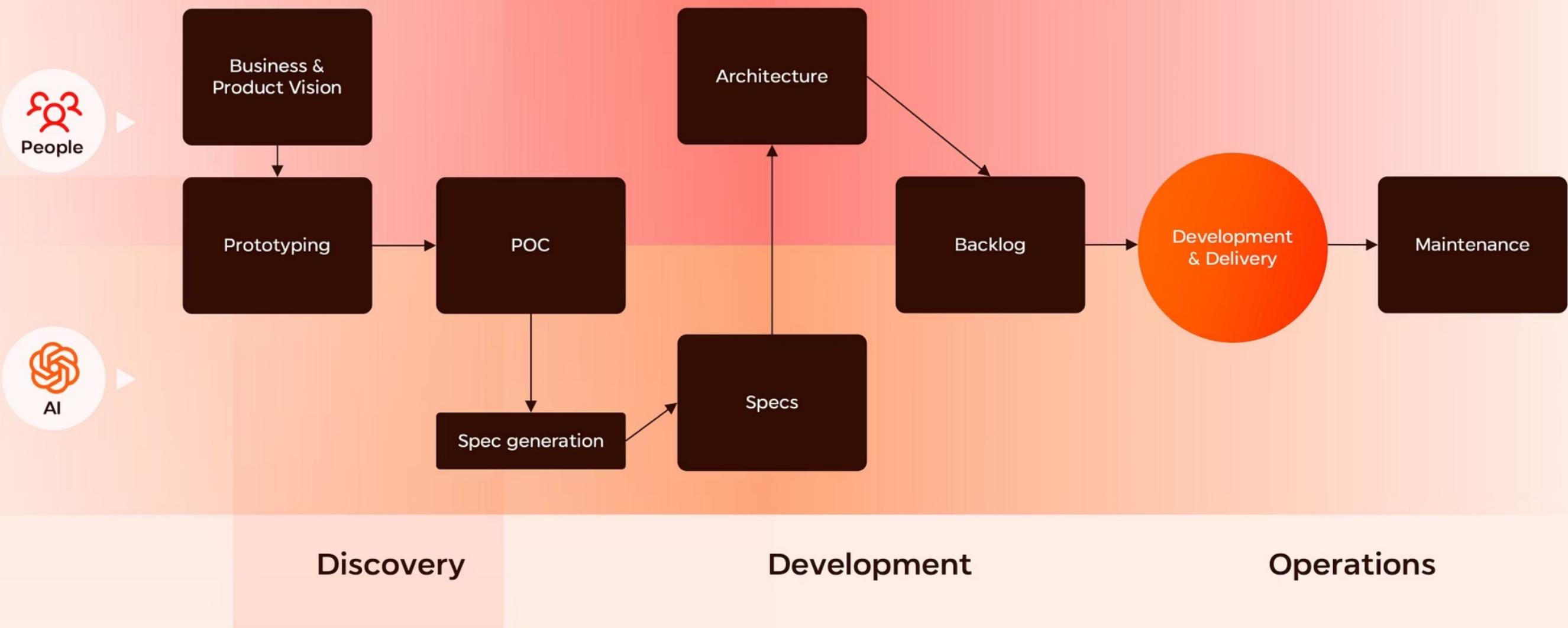
A QUOTE

“Without access to good and relevant data, this new world of possibilities and value will remain out of reach.”

# Results in Organizational Design: Development Company



# Our approach



v0.app

Community Enterprise Pricing iOS FAQ

Sign In Sign Up

Chadders Hero Section

Form Template

Minimalist Portfolio

Dashboard

Portfolio Template

Modern Agency Website

Shopify Ecommerce Template

Newsletter Template

Gabriela Cecchin

ACME

What did you think of our service?

Frontend Developer crafting digital experiences at the intersection of design, technology, and user experience.

I'M JULES, AN INDUSTRIAL DESIGNER WORKING WITH BIG AND SMALL CLIENTS FROM AROUND THE WORLD.

HIGH-IMPACT 3D ANIMATION FOR BRANDS

Refined. Minimal. Never boring.

Verde Leather Lounge Chair

Synecdoche®

PORTFOLIO // 2025

KATACHI

AcMem

QaS

5960 · 800

9516 · 537

2886 · 291

235 · 26

1004 · 233

601 · 89

1218 · 150

921 · 155

1635 · 311

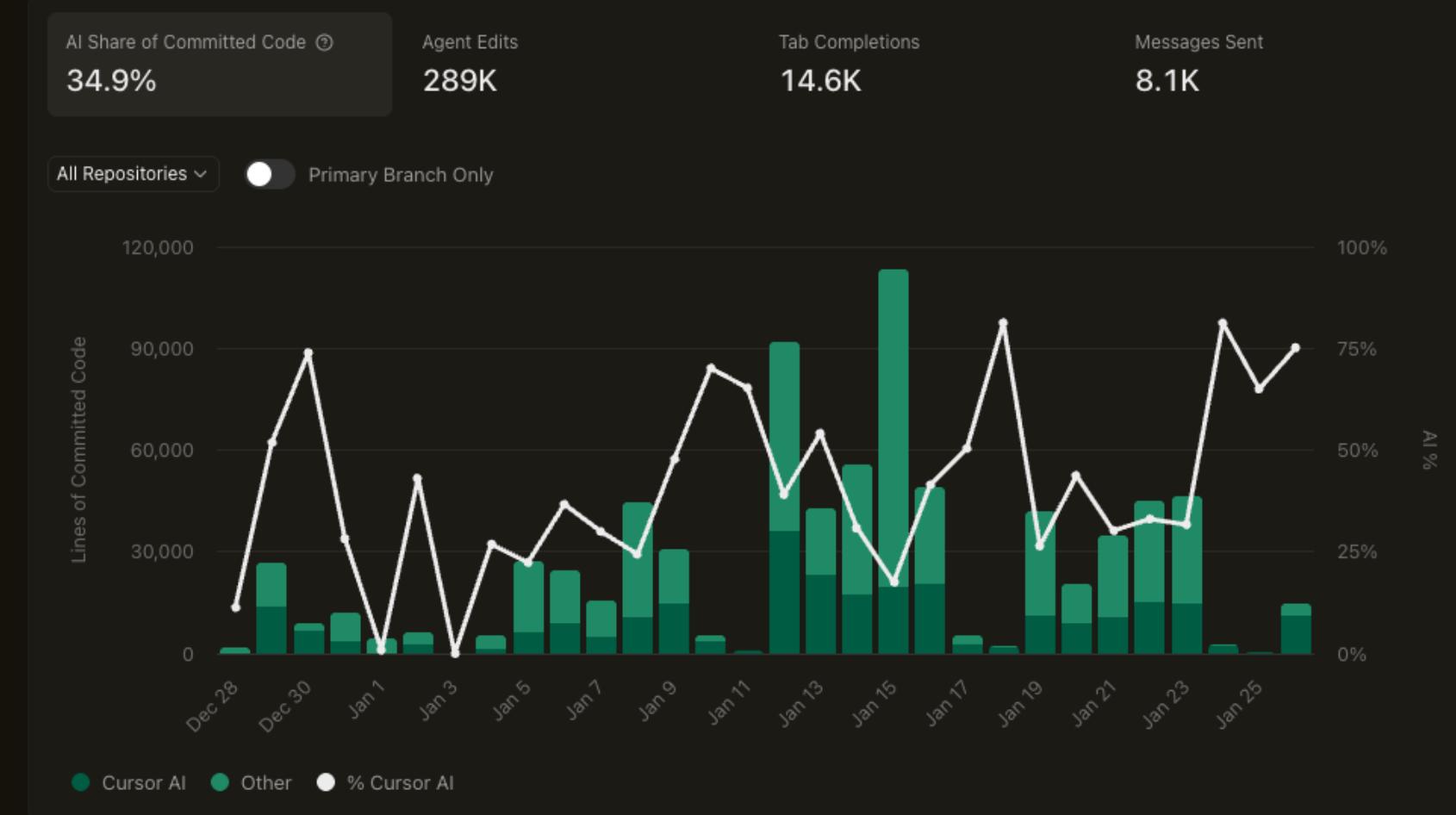
# Move to Orchestrator of agents



# Development Company

16% of code is generated by AI in September

34% of code is generated by AI in January



GitHub [Manage](#) [Connect](#) [Advanced](#)

Connect GitHub for Cloud Agents, Bugbot and enhanced codebase context  
Connected as 'roelofb' to repositories in GitHub organizations: joggroup, rb2-bv

GitLab [Manage](#) [Connect](#)

Connect GitLab for Cloud Agents, Bugbot and enhanced codebase context

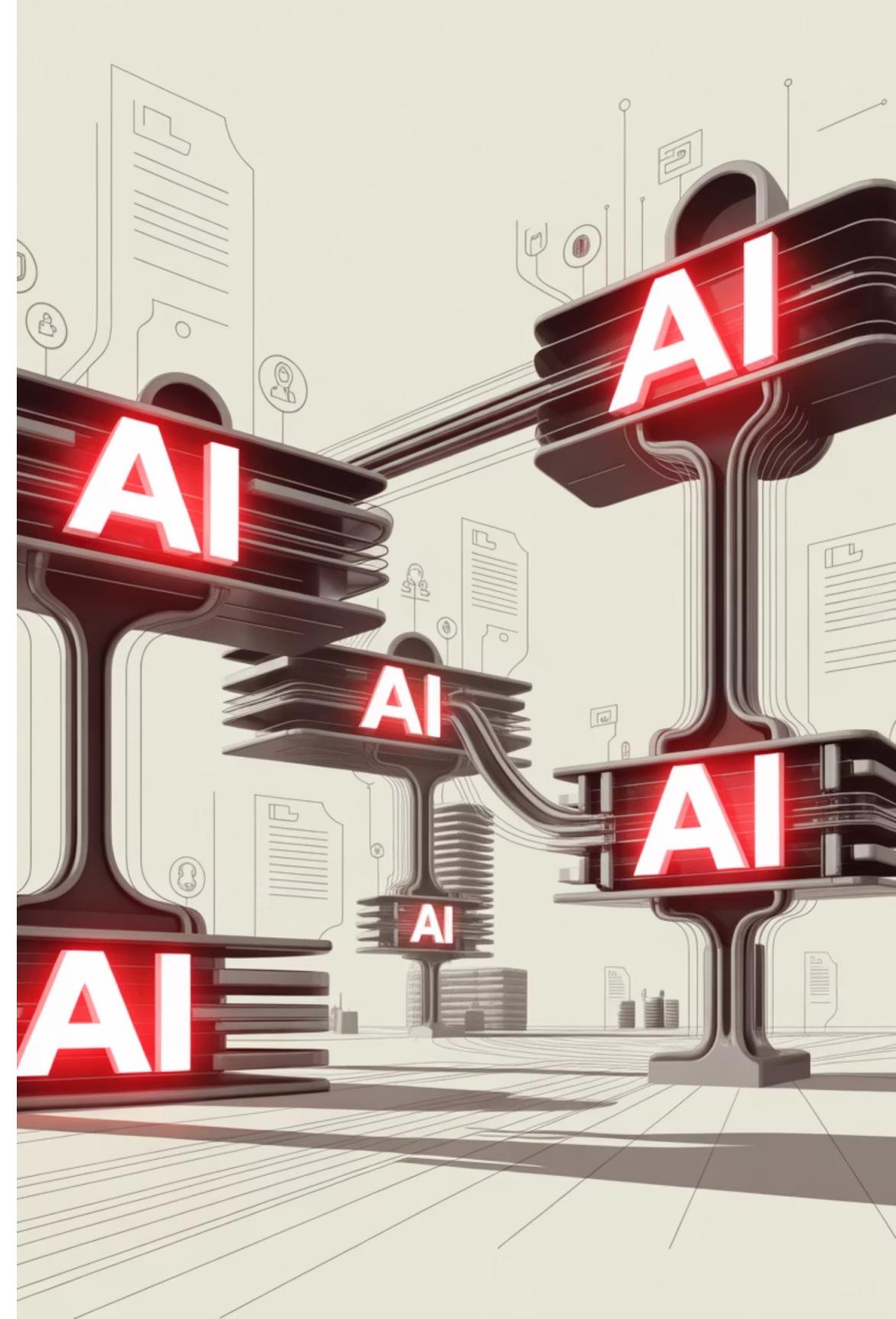
Slack [Manage](#)

Work with Cloud Agents from Slack

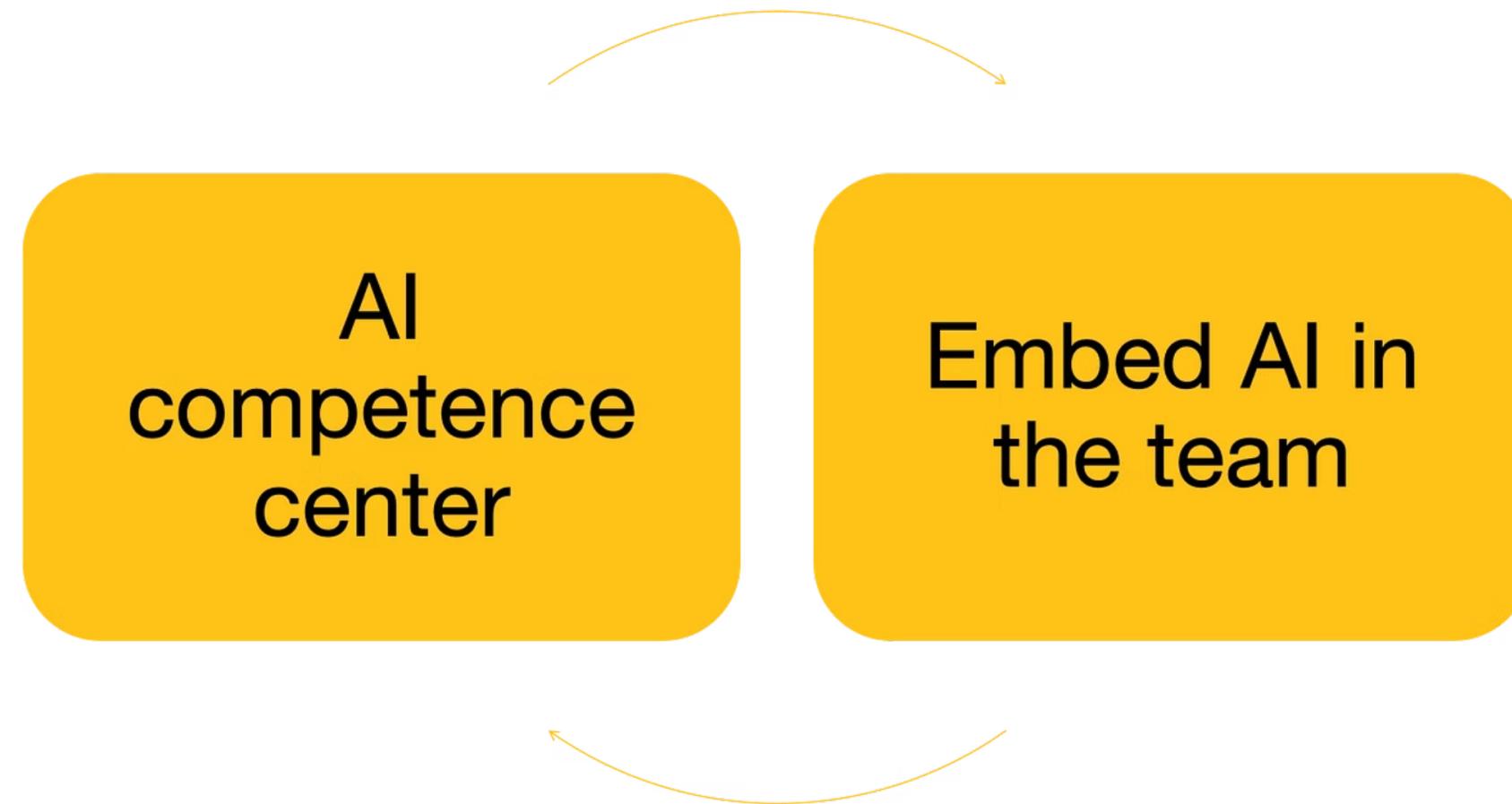
Linear [Manage](#)

Connect a Linear workspace to delegate issues to Cloud Agents  
Connected: rb2

# Results in Organizational Design: Insurance Company



# Organizational Design



# Do We Need More AI Work?



I have to continue to communicate in Dutch, but we will continue with the process.  
**Captured.**

Now we move on to the next field:

## 2. Epic Description

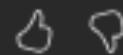
You haven't provided a description yet. This field usually contains:

- **Reason / context:** Why is this important?
- **Purpose:** What do we want to achieve?
- **Scope:** What is and is not covered?

**Proposal (example text):** "The goal of this Epic is to gain insight into the use of AI within the organization. This includes identifying where and how AI solutions are applied, measuring frequency of use, and collecting data to enable future optimizations. The scope focuses on internal AI tools and processes, not external customer applications."

*Please note that this is an illustrative example, not a definitive text. Always test this yourself.*

Do you want to capture this for this field?



9:18 pm

yes please, can you give insight how this epic will help DELA's objectives and goals



**Captured.**

Now the next field:



9:18 pm

yes please, can you give insight how this epic will help DELA's objectives and goals

DELA Epic AI Assistent 9:18 pm



**Captured.**

Now the next field:

### **3. Intended business results & contribution to DELA objectives**

You specifically ask how this Epic contributes to DELA's goals. This field describes:

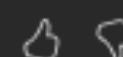
- **Which business goals are supported** (e.g.: efficiency, compliance, innovation).
- **How this Epic contributes to that.**

**Proposal (sample text):** "This Epic supports DELA's strategic objective to use technology responsibly and efficiently. By understanding AI usage, we can:

- Creating transparency about where AI is applied.
- Ensure compliance and ethical use.
- Making data-driven decisions about future investments in AI. This contributes to innovation and the improvement of internal processes, in line with DELA's ambition to use technology responsibly."

*Please note: this is an illustrative example. Always test this yourself.*

Do you want to capture this for this field?



can you explain a bit more regarding making data-driven decisions about future investments in AI. This contributes to innovation and the improvement of internal processes, in line with DELA's ambition to use technology responsibly



DELA Epic AI Assistent 9:19 pm 26



Good question! I will explain this further:

#### **Why data-driven decisions are important:**

- **Insight into usage patterns:** By measuring where and how AI is used, we get hard data on adoption, effectiveness and impact.
- **Substantiated investments:** Instead of investing based on assumptions, we can prioritize AI solutions that demonstrably add value.
- **Resource Optimization:** Data shows which AI applications are little used or do not produce the desired result, so that we use resources more efficiently.
- **Driving innovation:** By understanding which AI solutions contribute the most to process improvement, we can innovate in a more targeted way.
- **Responsible use:** Measuring helps to check whether AI applications meet ethical and compliance requirements, which is in line with DELA's ambition to use technology responsibly.

#### **How this contributes to DELA's goals:**

- **Innovation:** By understanding success factors, we can develop new AI initiatives that really add value.
- **Process improvement:** Data shows where AI accelerates or improves processes, so that we can scale up these successes.
- **Responsibility:** Transparency in AI use supports governance and compliance.

Would you like me to incorporate this extensive explanation in the field "**Intended business results & contribution to DELA objectives**" and present it again as a proposal?

# Experiments

## Do they help?

### Development Company

- **AI Bot (Agent Style)**: YES
- **Figma Components**: MODERATE
- **Customer Prototyping**: YES
- **GitHub Copilot**: YES/MODERATE
- **Workflow Agents**: YES
- **Vibe coding** : YES

### Insurance Company

- **AI Bot (Custom Version)**: MODERATE
- **Figma Components**: YES
- **Copilot Studio**: YES – Document checking at insurance
- **GitHub Copilot**: YES
- **Postbot AI**: YES
- **Epic AI**: YES
- **CoPilot Studio**: maintenance? we don't know yet..

# Key Takeaways

- 1 Early Adopters
- 2 AI Is Genuinely Helpful
- 3 "Vibe Coding" Accelerates POCs ....
- 4 "Standard" Code Is King
- 5 Testing is key for moving fast
- 6 AI Is a Pleaser and it is Addictive (and Costly)
- 7 Agents with SKILLS and Agentic workflows are the current experiments



**Thank you Martijn and  
the team for this journey**

**Questions?**

**Let's Continue the  
Conversation**

**Pascal Dufour**

AI development Consultancy  
Agile Test Consultancy

**Email:** [pascal@qualityatspeed.nl](mailto:pascal@qualityatspeed.nl)