



High quality and efficient application development collaboration solutions

Michał Rafalski

Product owner at IQVIA / Novartis

8 common issues

01

**Development
takes too much
time and resources**

02

**Architecture
is not optimised**

03

**Customisation
requires
refactoring**

04

**Code
duplicates
in
many projects**

05

**Missing
documentation**

06

**Code is not tested
properly and it is
not maintainable.**

07

**Too short
knowledge transfer
even inside one
team**

08

**Teams don't
know what
others achieved**

I would like to present
4 collaboration solutions to solve
8 common issues

4 collaboration solutions

01

Monorepo

Architecture to share features internally (building library in monorepo).



02

Tests

Providing stability of reusable parts with end-2-end and unit tests.



03

Registry

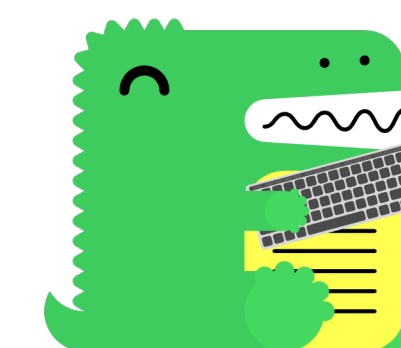
Service to share features with all developers.



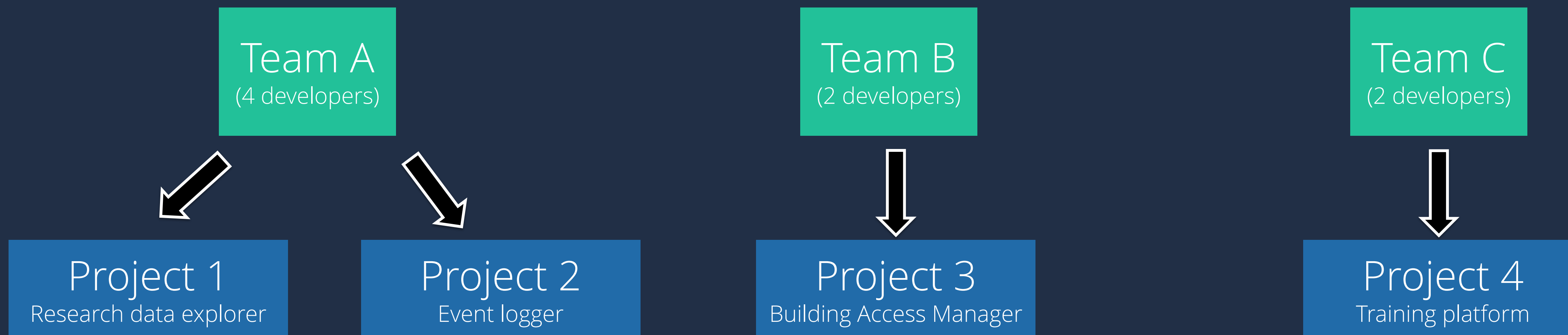
04

Transparency

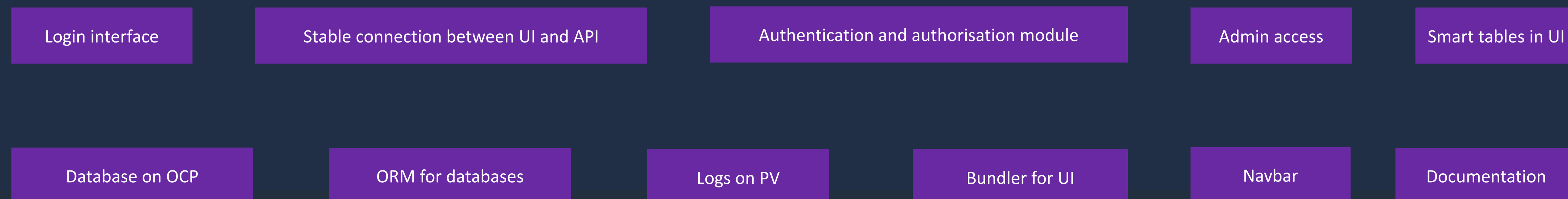
Share a list of features, all decisions, actions, conclusions, and knowledge with Docusaurus.



Story of 3 teams



All projects require:



NX workspace

<https://nx.dev/>

GETTING STARTED

TypeScript, React, Angular, Node and more

Nx has first-class support for many frontend and backend technologies, so its documentation comes in multiple flavours.



TYPESCRIPT



REACT

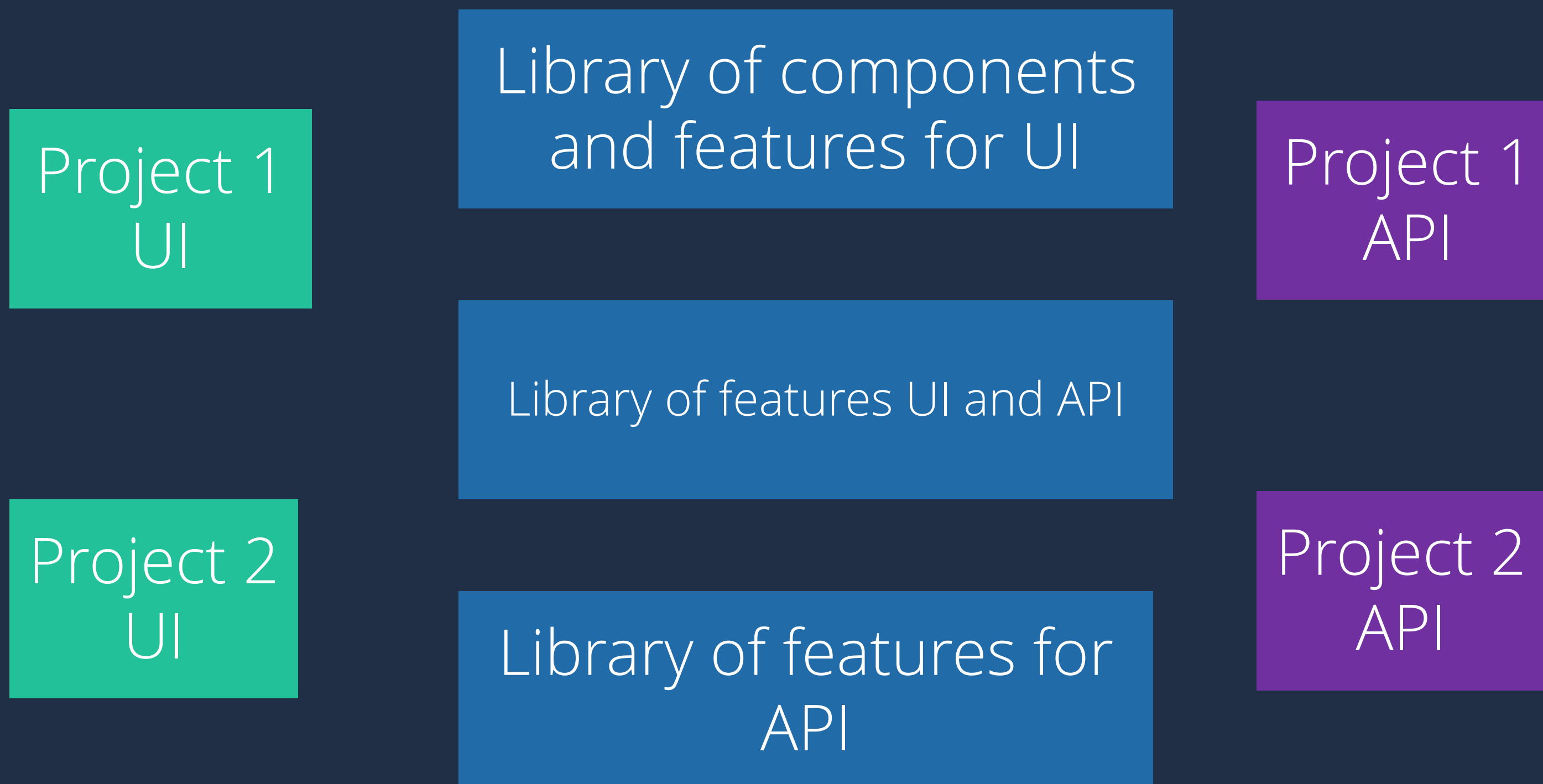


ANGULAR



NODEJS

NX workspace for Team A



1 minute configuration of NX workspace

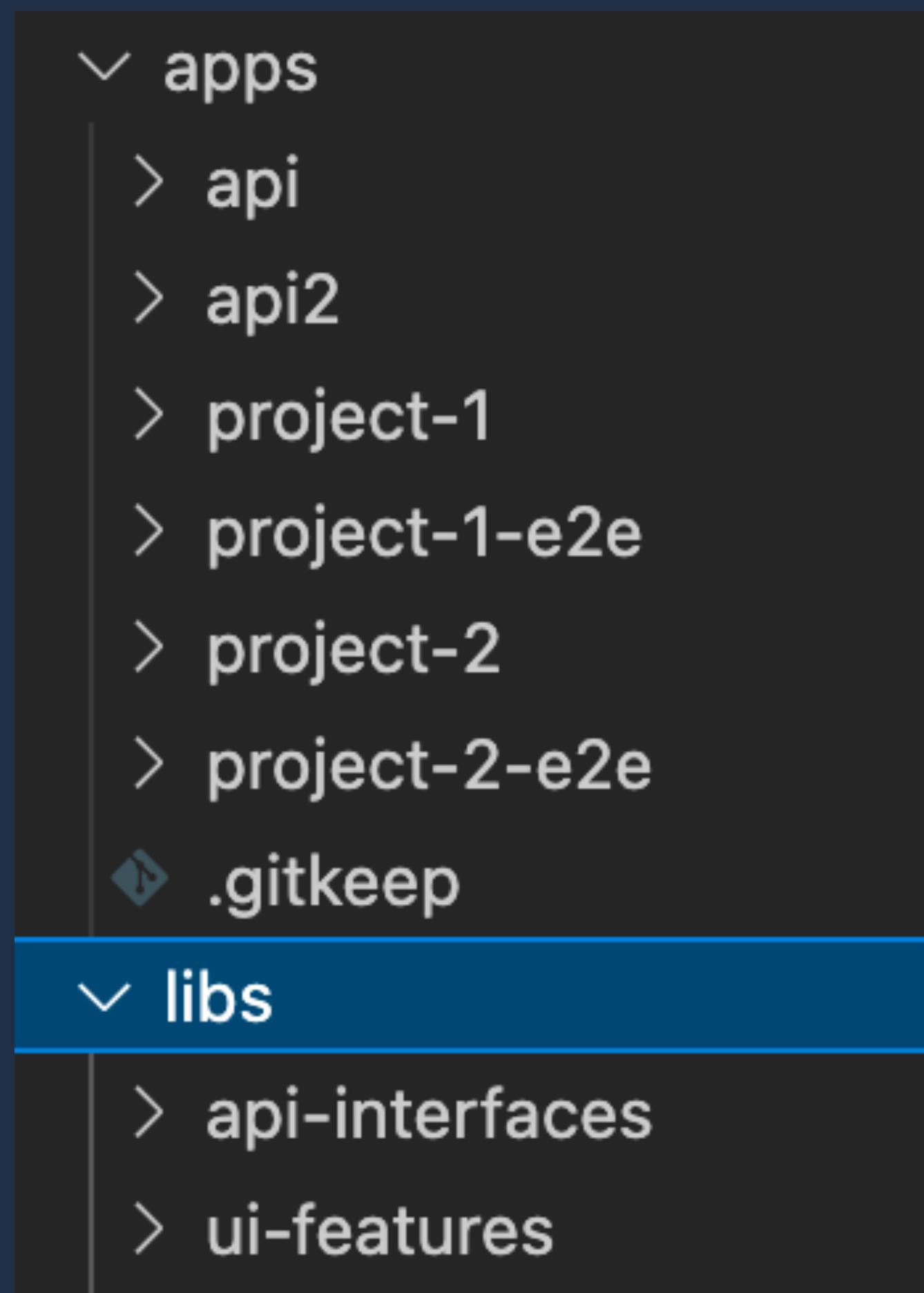
```
(base) C02Z51N3LVCJ:~ rafalmi1$ npx create-nx-workspace@14.1.7
```

```
yarn nx generate @nrwl/react:app project-2
```

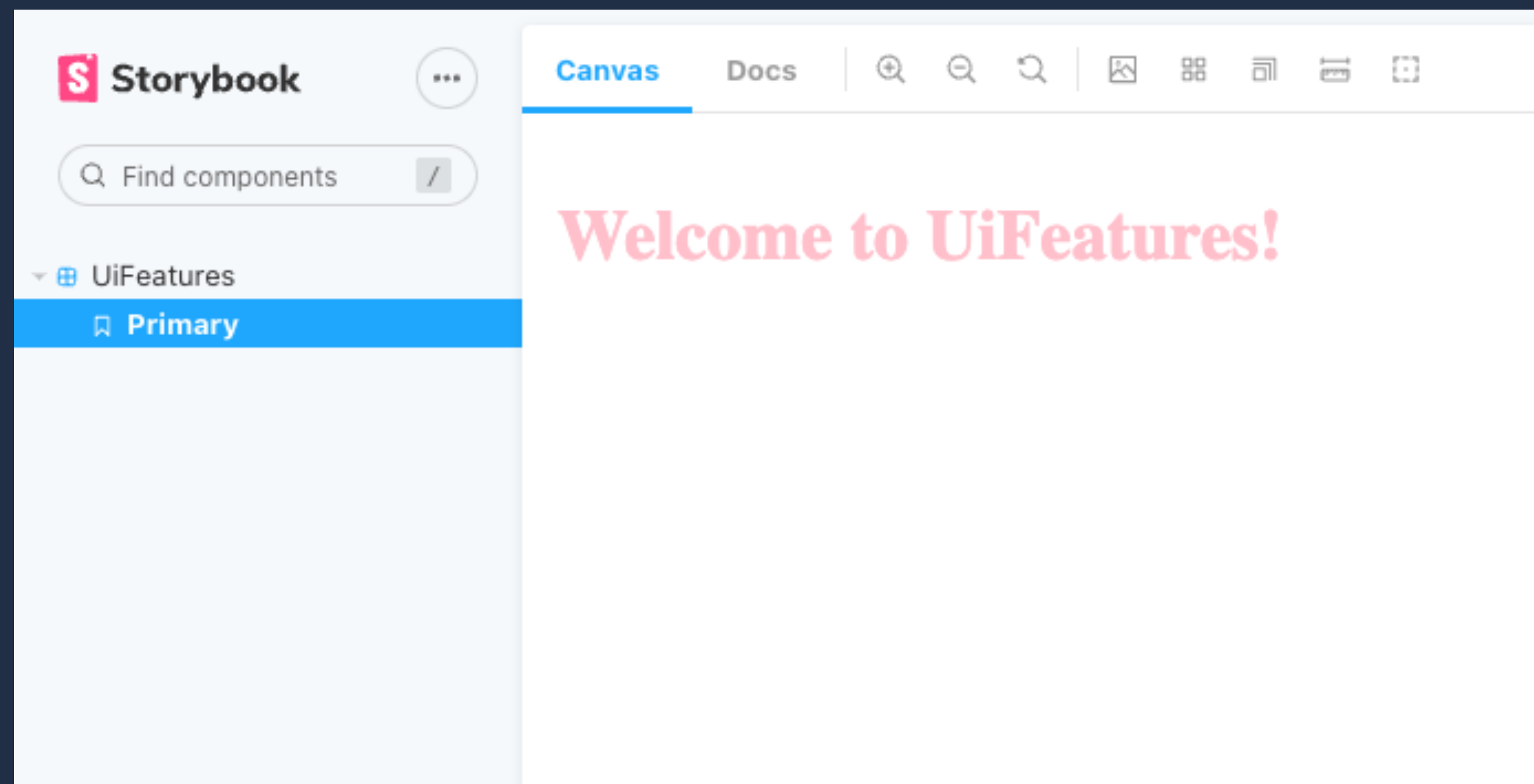
```
yarn nx generate @nrwl/express:app api2
```

```
yarn nx generate @nrwl/react:lib ui-features
```

Prepare for reusable component in NX workspace



```
import { UiFeatures } from '@team-a-workspace/ui-features';
```



```
yarn add --dev @nrwl/storybook  
yarn nx g @nrwl/react:storybook-configuration ui-features  
yarn nx run ui-features:storybook
```

First success: in Team A Project 1 and Project 2 can use the same reusable component!

Features of Storybook

Canvas

Docs







Welcome to App ↻

Your application name

Username

Password

[Sign In](#)

[Forgot your username or password?](#)

Controls (10)

Actions



Name	Control	
authServerUrl*	Set string	
appName*	Set string	
metricsToken*	Set string	
appVersion	Set string	



Welcome to MDR ↻

Meta Data Repository

Username

Password

Sign In

[Forgot your username or password?](#)



Welcome to Comet ↻

Controlled Terminology Management

Username

Password

Please enter your password

Sign In

[Forgot your username or password?](#)



Welcome to Clinical Data Explorer ↻

Explore GPS datasets in your browser

Username

Password

Sign In

[Forgot your username or password?](#)

Building stable and maintainable components with e2e tests



```
yarn nx run ui-features-e2e:e2e --watch
```

```
cy.get('#username').type("MyUserName");
```

```
cy.get("#username").clear();
```

```
cy.get("div").contains("Please enter your username").should('exist')
```

```
cy.get('#username').type("MyUserName");
```

```
cy.get("div").contains("Please enter your username").should('not.exist')
```

Cypress demo



Cypress is currently automating this browser.

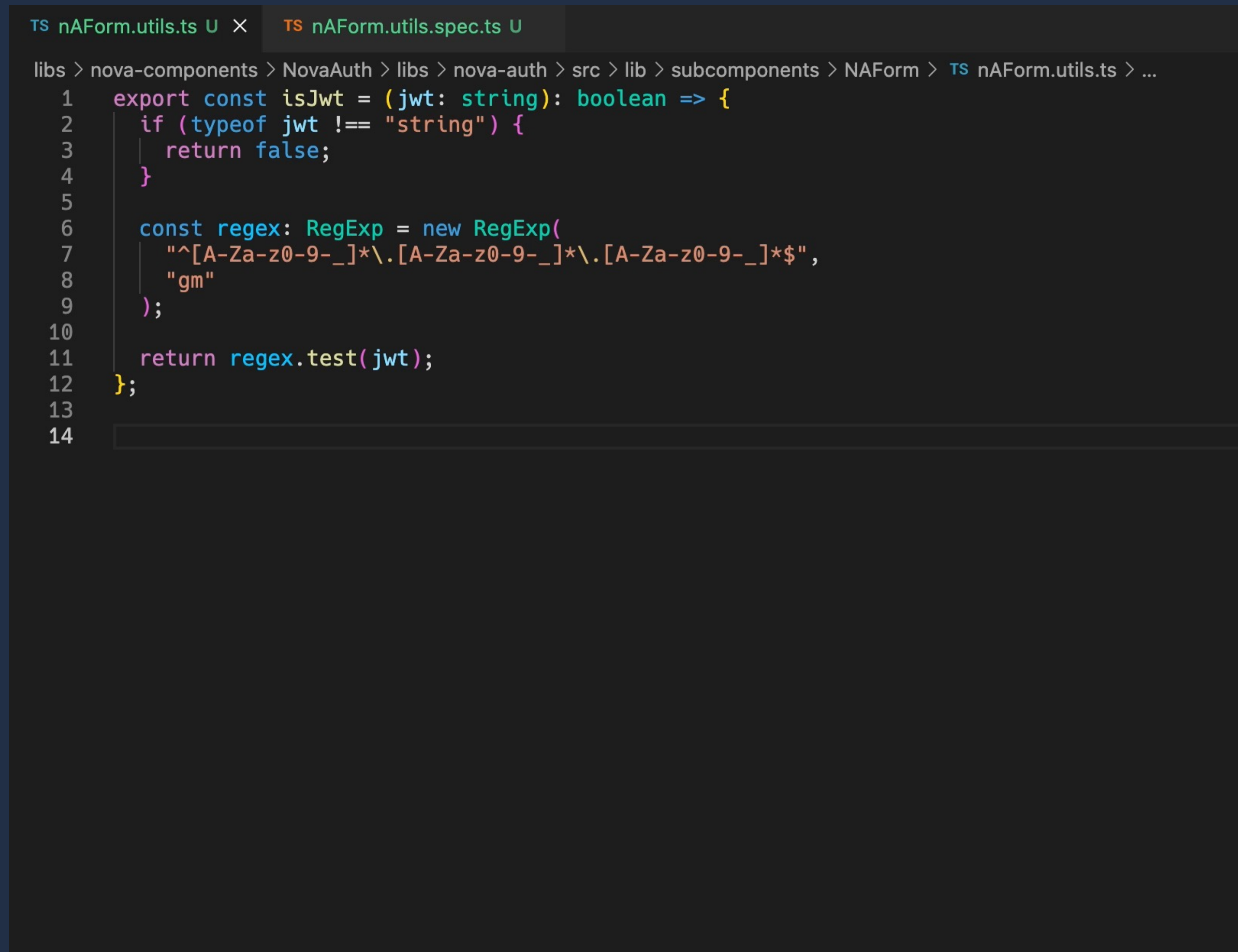
Opening new tabs may interfere with tests and cause failures.

Please note:

- Any opened tabs will be closed when Cypress is stopped.
- Tests currently running may fail while another tab has focus.
- Cookies and session from other sites will be cleared.

[Read more about browser management](#)

Building stable and maintainable features with unit tests

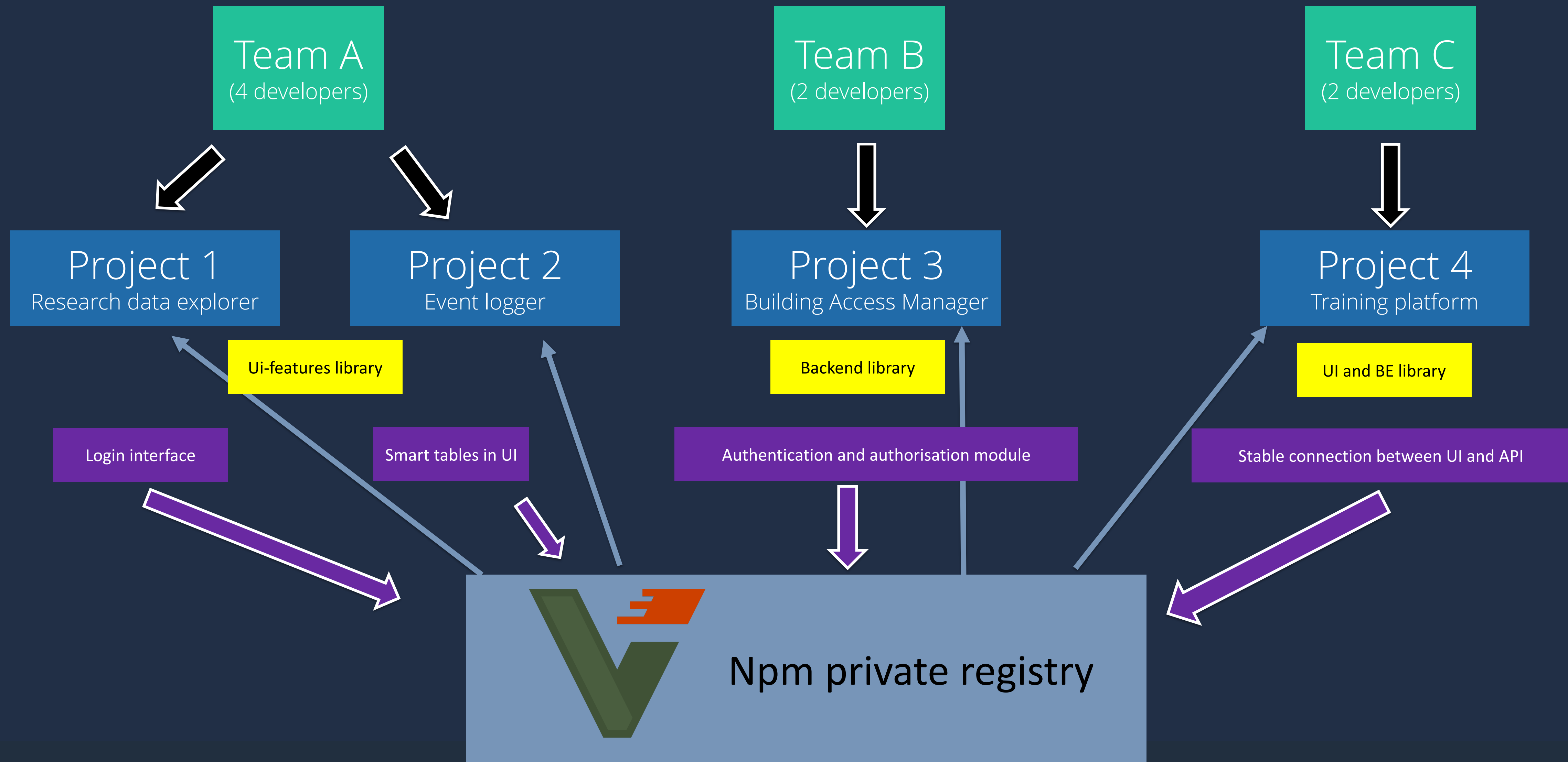


The screenshot shows a code editor with two tabs: 'TS nAForm.utils.ts U' and 'TS nAForm.utils.spec.ts U'. The active tab is 'TS nAForm.utils.spec.ts U'. The breadcrumb navigation path is: 'libs > nova-components > NovaAuth > libs > nova-auth > src > lib > subcomponents > NAForm > TS nAForm.utils.ts > ...'. The code in the editor is as follows:

```
1  export const isJwt = (jwt: string): boolean => {
2    if (typeof jwt !== "string") {
3      return false;
4    }
5
6    const regex: RegExp = new RegExp(
7      "[A-Za-z0-9-._]*\\.[A-Za-z0-9-._]*\\.[A-Za-z0-9-._]*$",
8      "gm"
9    );
10
11    return regex.test(jwt);
12  };
13
14
```

Second success: our features inside the component are stable, well tested and ready for many edge cases.

Publishing reusable features



Publishing reusable features with Verdaccio

The screenshot shows the Verdaccio web interface for the 'login-interface' package. The top navigation bar includes a search bar and icons for help, info, settings, and user. The main content area has tabs for README, DEPENDENCIES, VERSIONS, and UPLINKS. The README tab is active, displaying the package name 'Login Interface' and its description 'Login interface for all web apps'. Below the description is a preview image of a login form with fields for 'Your application name', 'Username', and 'Password', and a 'Sign in' button. To the right of the preview, there is a sidebar with installation instructions for npm, yarn, and pnpm, the latest distribution version (v1.0.3), the license (MIT), and the author (Michal Rafalski).

Verdaccio Search Packages

README DEPENDENCIES VERSIONS UPLINKS

Login Interface

Login interface for all web apps

Welcome to App ↻

Your application name

Username

Password

Sign in Forgot your username or password?

Install

yarn add login-interface

login-interface TS

Login interface for all web apps
Latest v1.0.3

Installation

- npm install login-interface
Install using npm
- yarn add login-interface
Install using yarn
- pnpm install login-interface
Install using pnpm

Latest Distribution

License: MIT

Author

Michal Rafalski

Third success: shareable features from all groups are available for everyone. Team A can use features from team B and team C and vice versa.

Transparency

1. Present your applications to other groups
2. Document your decisions about the technology stack that others can benefit from it
3. List all features from your applications and make it accessible to others
4. Document well your research
5. Write articles to cover unique knowledge about projects you work on
6. Document logic of functions with Mermaid.js that others can understand better the implementation



Docusaurus

```

---
id: greeting
title: Hello
---

## Hello from Docusaurus

Are you ready to create the documentation site for your open source project?

### Headers

will show up on the table of contents on the upper right

So that your users will know what this page is all about without scrolling down or even without reading too m

```

Hello from Docusaurus

Are you ready to create the documentation site for your open source project?

Headers

will show up on the table of contents on the upper right

So that your users will know what this page is all about without scrolling down or even without reading too much.

```

```jsx title="/src/components/HelloCodeTitle.js"
function HelloCodeTitle(props) {
 return <h1>Hello, {props.name}</h1>;
}
```

```

/src/components/HelloCodeTitle.js

```

function HelloCodeTitle(props) {
  return <h1>Hello, {props.name}</h1>;
}

```

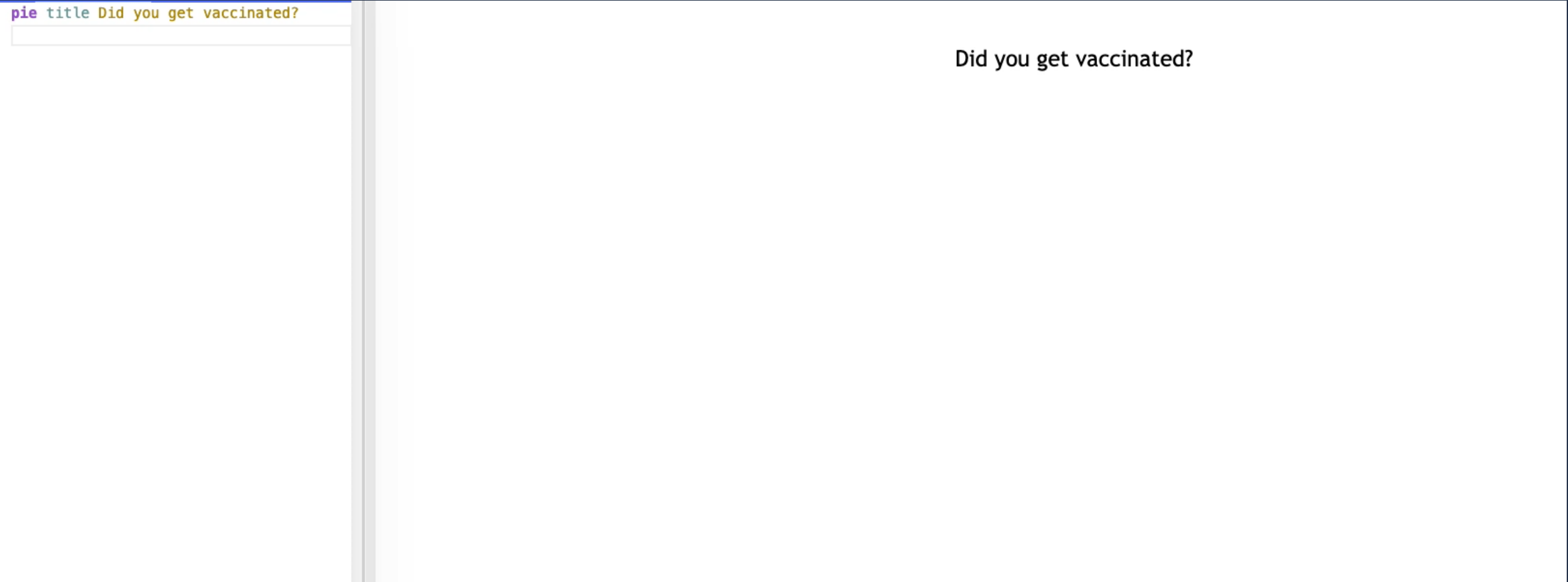
```

$$
I = \int_0^{2\pi} \sin(x) dx
$$

```

$$I = \int_0^{2\pi} \sin(x) dx$$

Mermaid.js



Fourth success: all decisions, actions, conclusions, features, and knowledge are documented.

8 solved common issues

01

**Development
takes too much
time and resources**

02

**Architecture
is not optimised**

03

**Customisation
requires
refactoring**

04

**Code
duplicates
in
many projects**

05

**Missing
documentation**

06

**Code is not tested
properly and it is
not maintainable.**

07

**Too short
knowledge transfer
even inside one
team**

08

**Teams don't
know what
others achieved**



Thank you for your attention

Michał Rafalski
Product owner at IQVIA / Novartis