C∅NF42

Who likes unit testing?

You code, let GenAI DevTools worry about the Test.

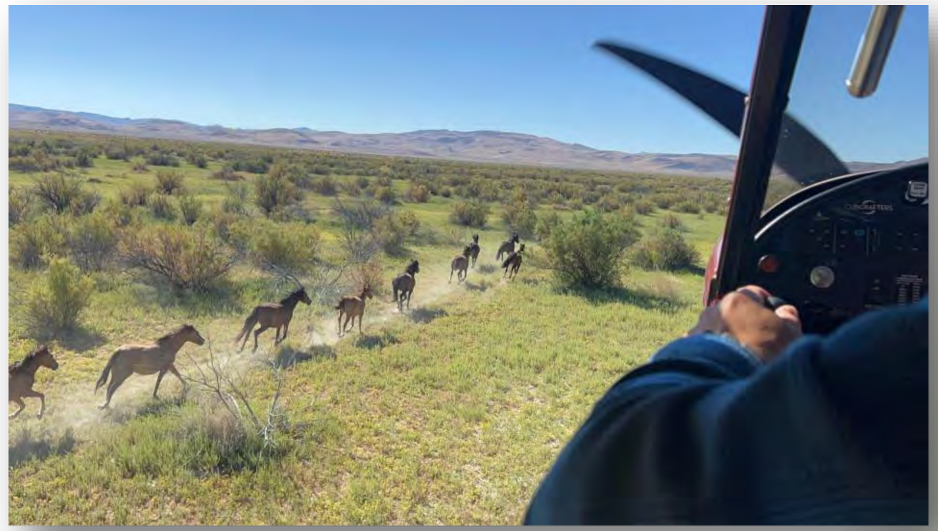
Sharon Barr

About myself

- Lived 17 years in the Bay area, worked at Yahoo, Couchbase, Bina (life science acquired by Roche)

- Former developer
- Founder of Early
  - We generate high quality, working unit tests for your PRs & at scale
  - Chief Test Code Generator - Generated over 500K LoC in year 1

Love traveling and outdoor sports with family and friends



Bush plane in Nevada

Trail Mont Blanc

Tiny Bugs Lead to MASSIVE CRISES

CrowdStrike Outage

What was the issue?

Array out-of-bounds read

Null Pointer Exception

BSOD

Highest cost of bug, ever…

Outcome:

- $**5.4bn** in losses according to report.
- 1 in 4 Fortune 500 companies impacted

UNITED
STAR ALLIANCE
:(
Your device ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.
100% complete
Gates C70-C99
via Shuttle
Terminals A B
United Club (C74)

Delta CEO says CrowdStrike-Microsoft outage cost the airline $500 million

https://www.cnbc.com

Tiny Bugs Lead to MASSIVE CRISES

CrowdStrike Outage

What was the issue?

Array out-of-bounds read

↓

Null Pointer Exception

↓

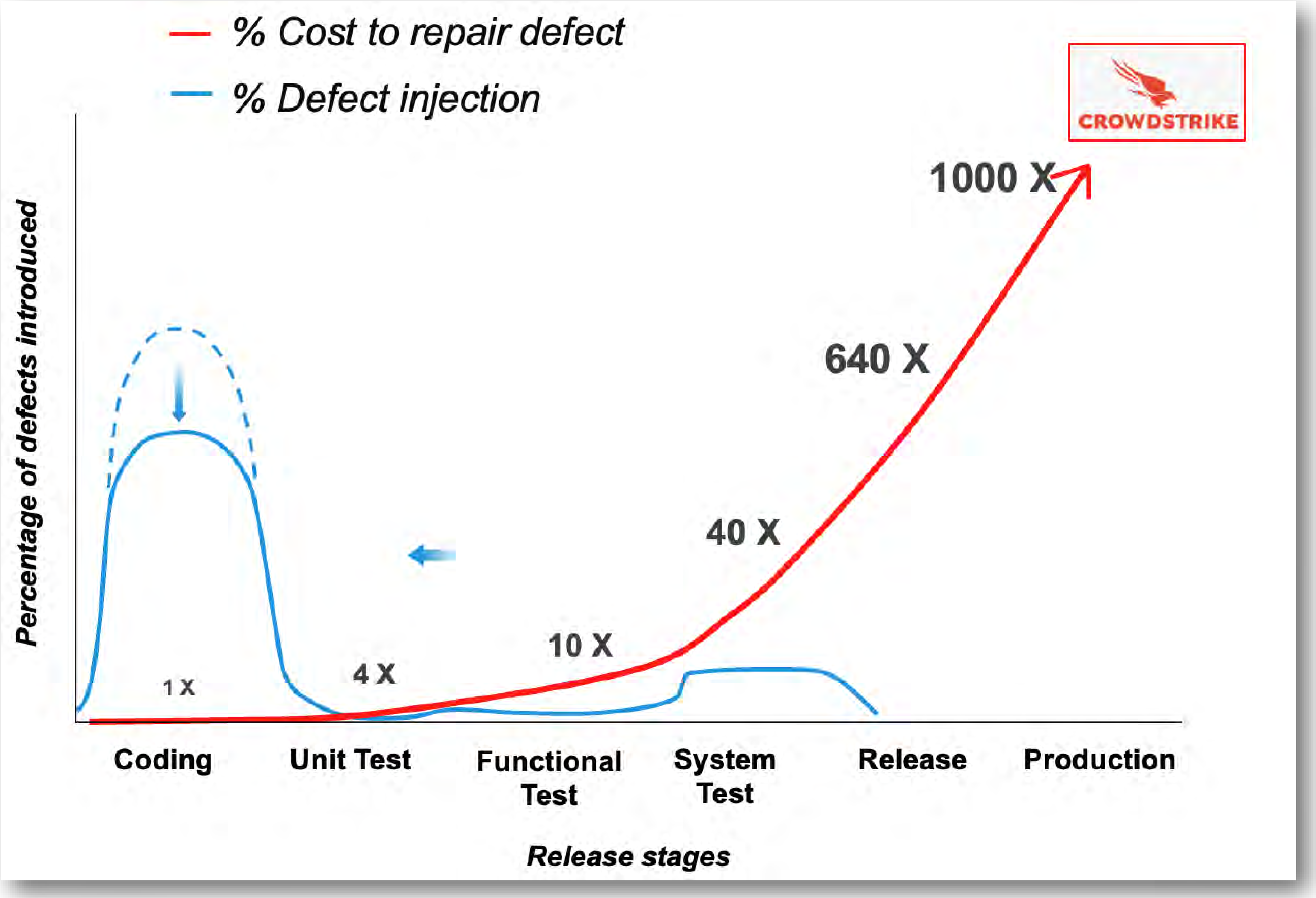
BSOD

↓

Highest cost of bug, ever…

Outcome:

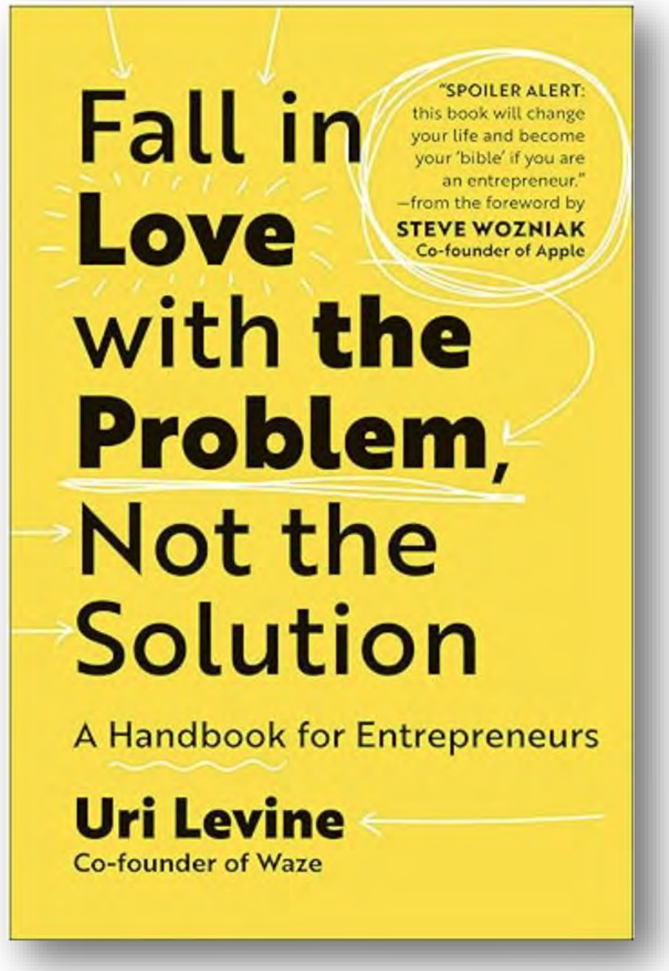
- **$5.4bn** in losses according to report.
- 1 in 4 Fortune 500 companies impacted
- Delta Airlines alone lost **$500M** by



“If I had asked people what they wanted, they would have said faster horses.”

Henry Ford

Fall in love with the problem



Fall in Love with the Problem, Not the Solution

A Handbook for Entrepreneurs

Uri Levine

Co-founder of Waze

"SPOILER ALERT: this book will change your life and become your 'bible' if you are an entrepreneur." —from the foreword by STEVE WOZNIAK Co-founder of Apple

Bugs are introduced whenever new code is developed

- What can we do with new DevTools to solve code quality problem?
- Transforming

  from: "horses" - develop test code manually

  to: "cars" - generate test code automatically

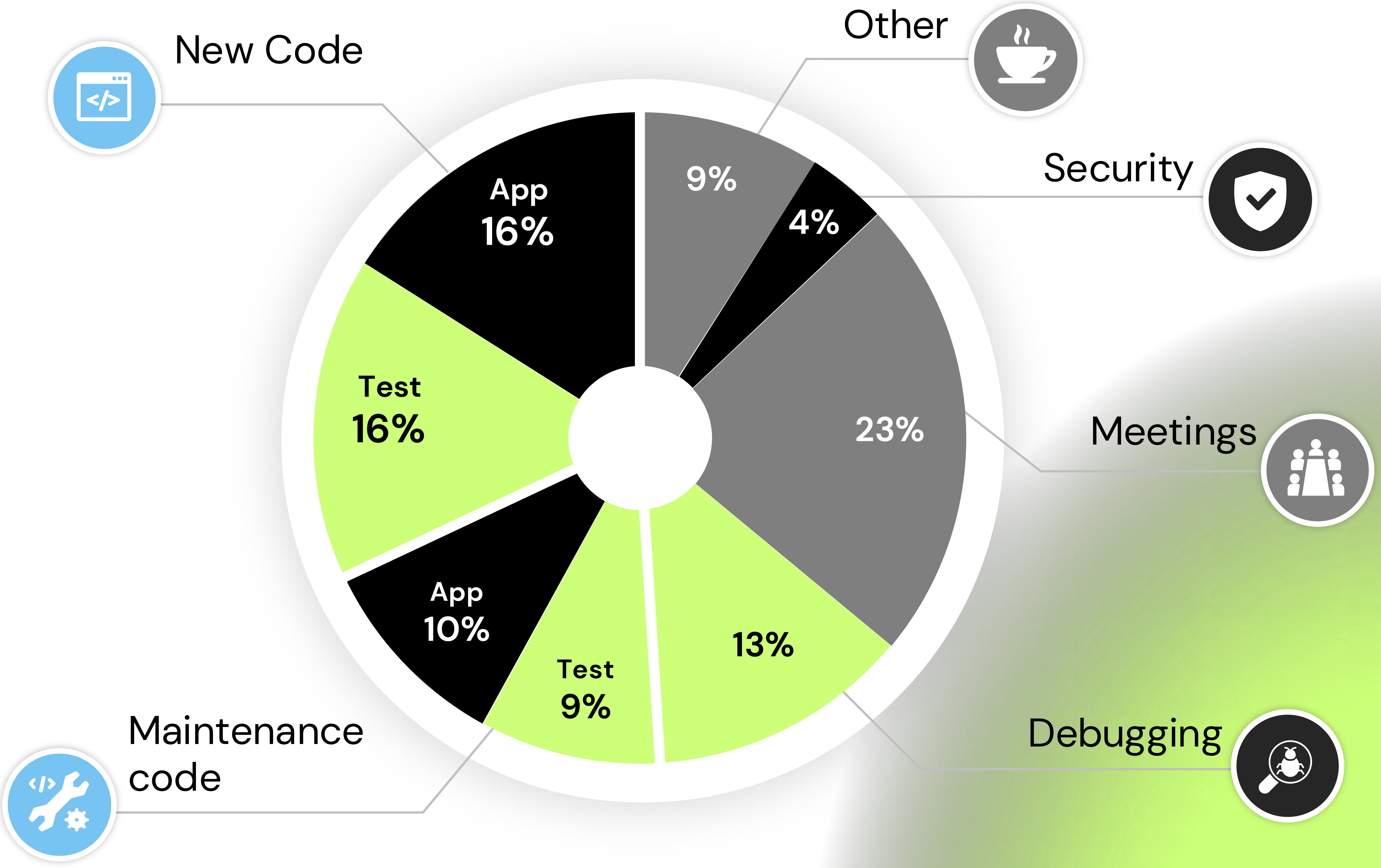
How developers spend their time

Time spent on application code

26%

Time spent on test code and testing

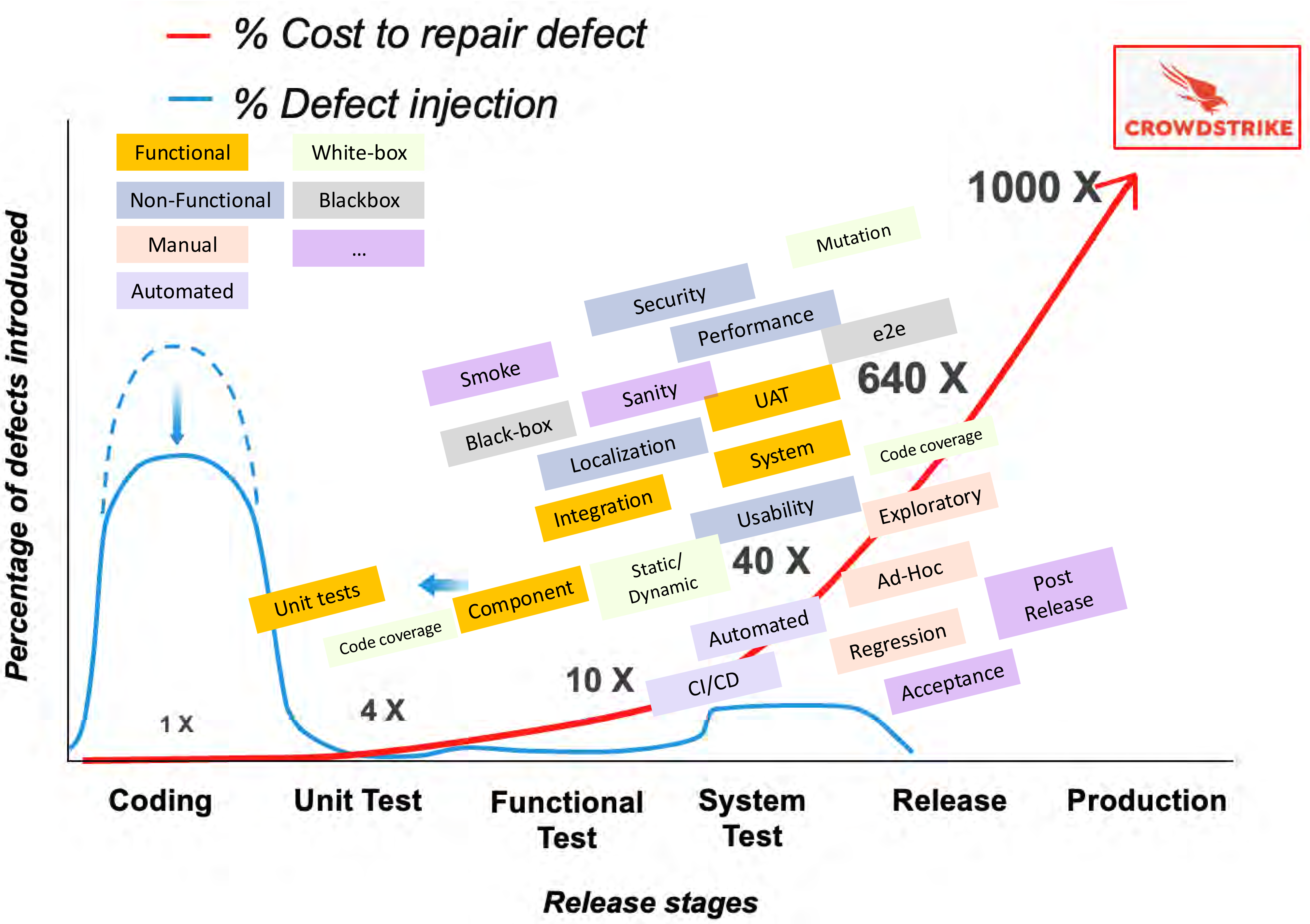
38%



Sources: https://www.microsoft.com/en-us/research/uploads/prod/2019/04/devtime-preprint-TSE19.pdf
https://thenewstack.io/how-much-time-do-developers-spend-actually-writing-code/

Where should I spend my time developing tests?

- ROI on each type of test
- ROI changed as AI tools arise
- What if we could catch most bugs during coding?



AI tools for unit test code generation

Value of Unit tests Generation

- Catch bugs at their lowest cost
- Free up developers’ time
- Speed up release cycles
- Can be used in multiple use cases during development

Assistant DevTools for test code generation

GitHub Copilot

Cursor

codeium

Test Code AI Agent:

ear/y

?

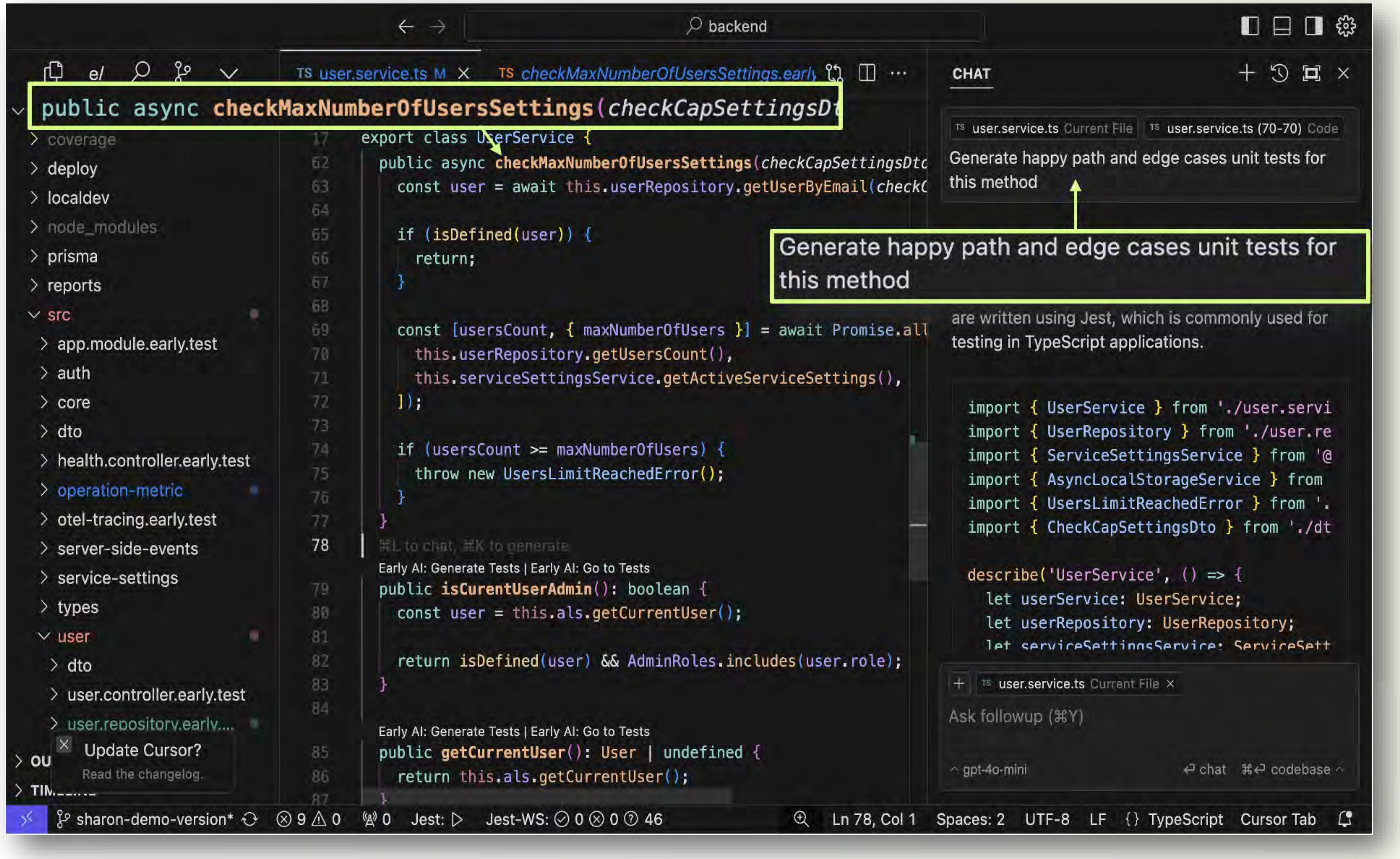
Cognition

poolside

Let’s generate some unit tests

Cursor – AI assistant

Early – AI Agent



```
public async checkMaxNumberOfUsersSettings(checkCapSettingsDt
```

```
export class UserService {
  public async checkMaxNumberOfUsersSettings(checkCapSettingsDtc
    const user = await this.userRepository.getUserByEmail(checkC

    if (isDefined(user)) {
      return;
    }

    const [usersCount, { maxNumberOfUsers }] = await Promise.all
      this.userRepository.getUsersCount(),
      this.serviceSettingsService.getActiveServiceSettings(),
    ]);

    if (usersCount >= maxNumberOfUsers) {
      throw new UsersLimitReachedError();
    }
  }
  ⌘L to chat, ⌘K to generate
  Early AI: Generate Tests | Early AI: Go to Tests
  public isCurentUserAdmin(): boolean {
    const user = this.als.getCurrentUser();

    return isDefined(user) && AdminRoles.includes(user.role);
  }

  Early AI: Generate Tests | Early AI: Go to Tests
  public getCurrentUser(): User | undefined {
    return this.als.getCurrentUser();
```

user.service.ts Current File | user.service.ts (70-70) Code

Generate happy path and edge cases unit tests for this method

Generate happy path and edge cases unit tests for this method

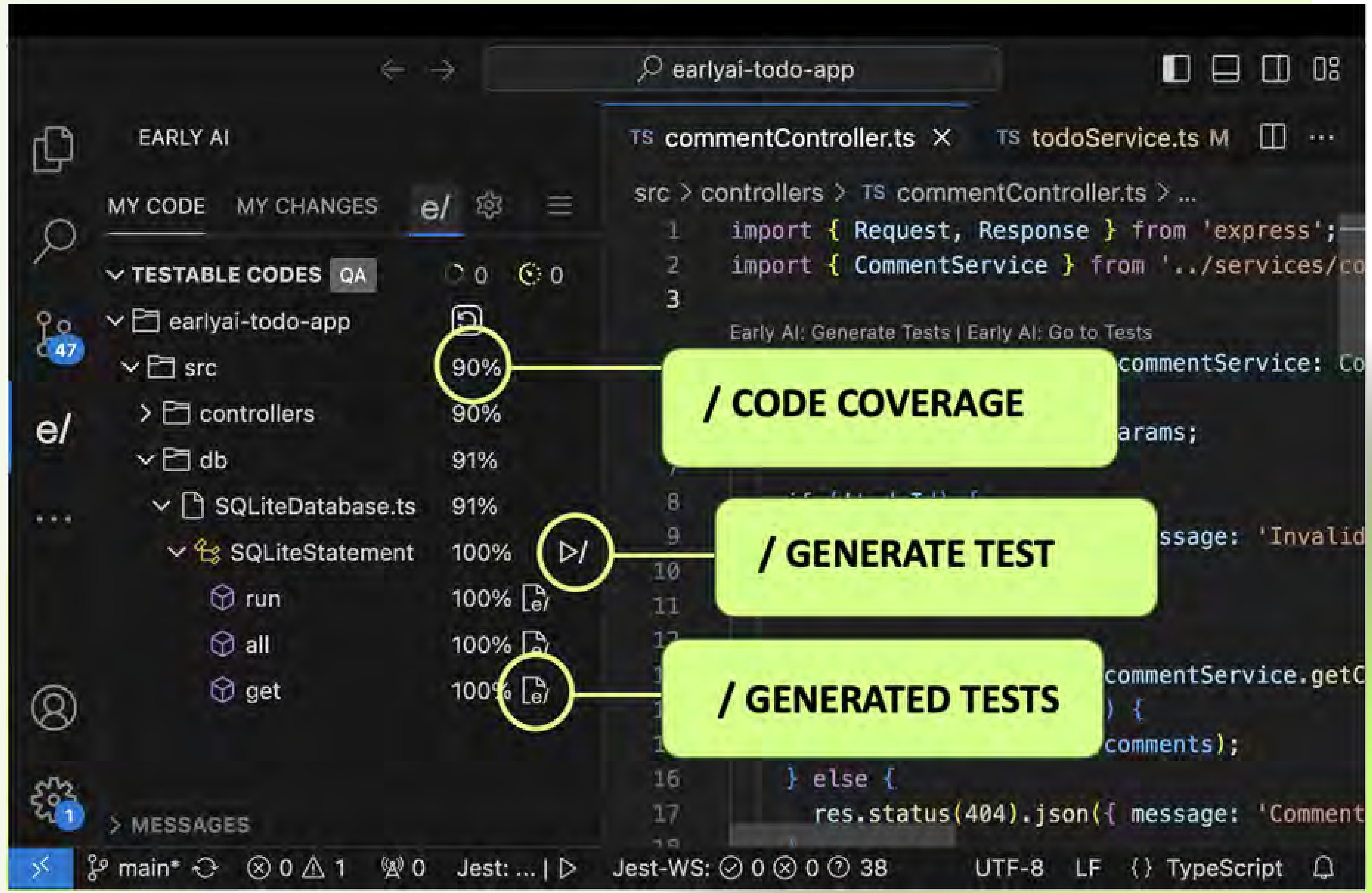
are written using Jest, which is commonly used for testing in TypeScript applications.

```
import { UserService } from './user.servi
import { UserRepository } from './user.re
import { ServiceSettingsService } from '@
import { AsyncLocalStorageService } from
import { UsersLimitReachedError } from '.
import { CheckCapSettingsDto } from './dt

describe('UserService', () => {
  let userService: UserService;
  let userRepository: UserRepository;
  let serviceSettingsService: ServiceSett
```

user.service.ts Current File

Ask followup (⌘Y)

gpt-4o-mini | chat | ⌘ codebase



earlyai-todo-app	
src	90%
controllers	90%
db	91%
SQLiteDatabase.ts	91%
SQLiteStatement	100%
run	100%
all	100%
get	100%

```
import { Request, Response } from 'express';
import { CommentService } from '../services/co

Early AI: Generate Tests | Early AI: Go to Tests
commentService: Co
arams;
ssage: 'Invalid
commentService.getC
) {
comments);
} else {
  res.status(404).json({ message: 'Comment
```

/ CODE COVERAGE

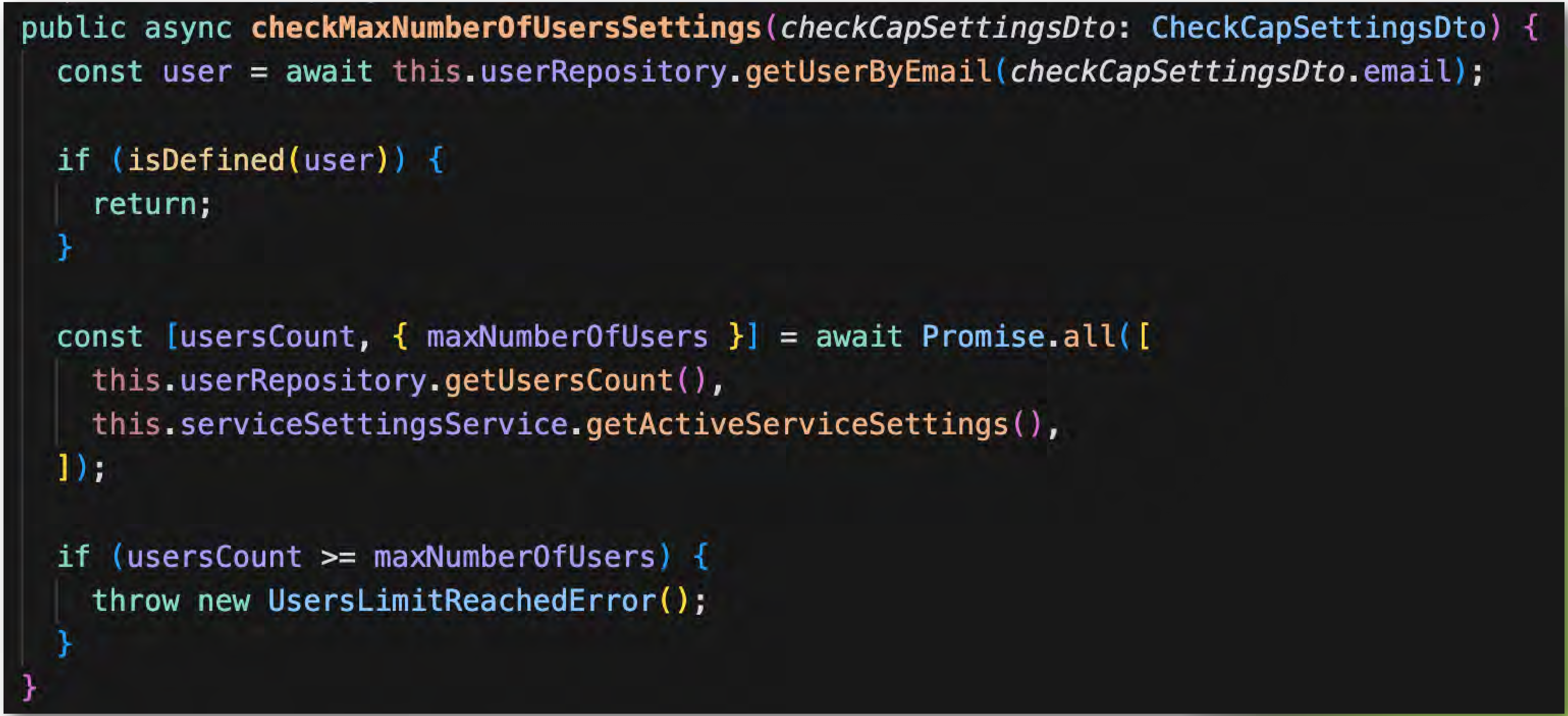
/ GENERATE TEST

/ GENERATED TESTS

CURSOR

early

Solutions we’ll explore for test code generation

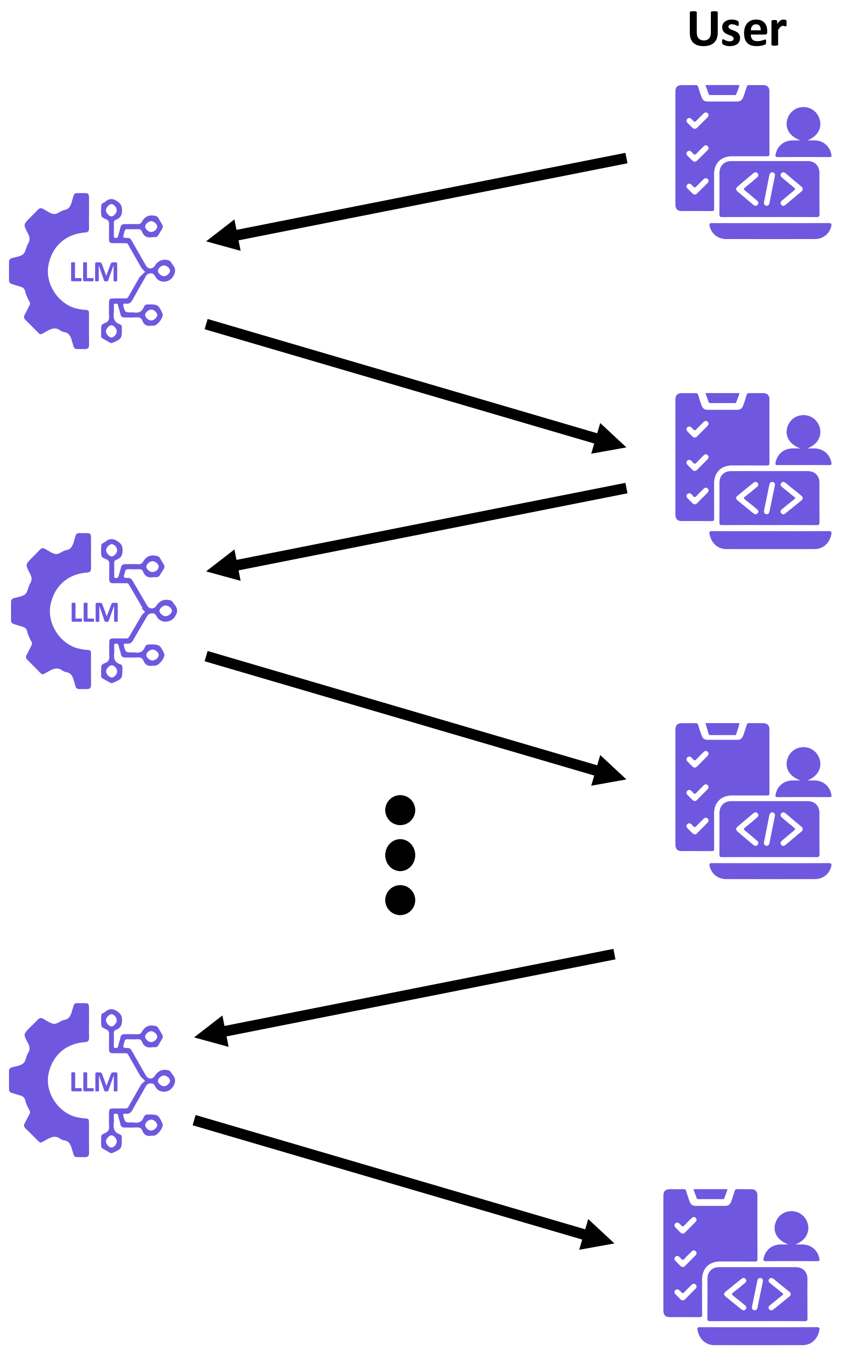


```
public async checkMaxNumberOfUsersSettings(checkCapSettingsDto: CheckCapSettingsDto) {
  const user = await this.userRepository.getUserByEmail(checkCapSettingsDto.email);

  if (isDefined(user)) {
    return;
  }

  const [usersCount, { maxNumberOfUsers }] = await Promise.all([
    this.userRepository.getUsersCount(),
    this.serviceSettingsService.getActiveServiceSettings(),
  ]);

  if (usersCount >= maxNumberOfUsers) {
    throw new UsersLimitReachedError();
  }
}
```

From **AI assistant** to **AI agents**

AI **Assistant**

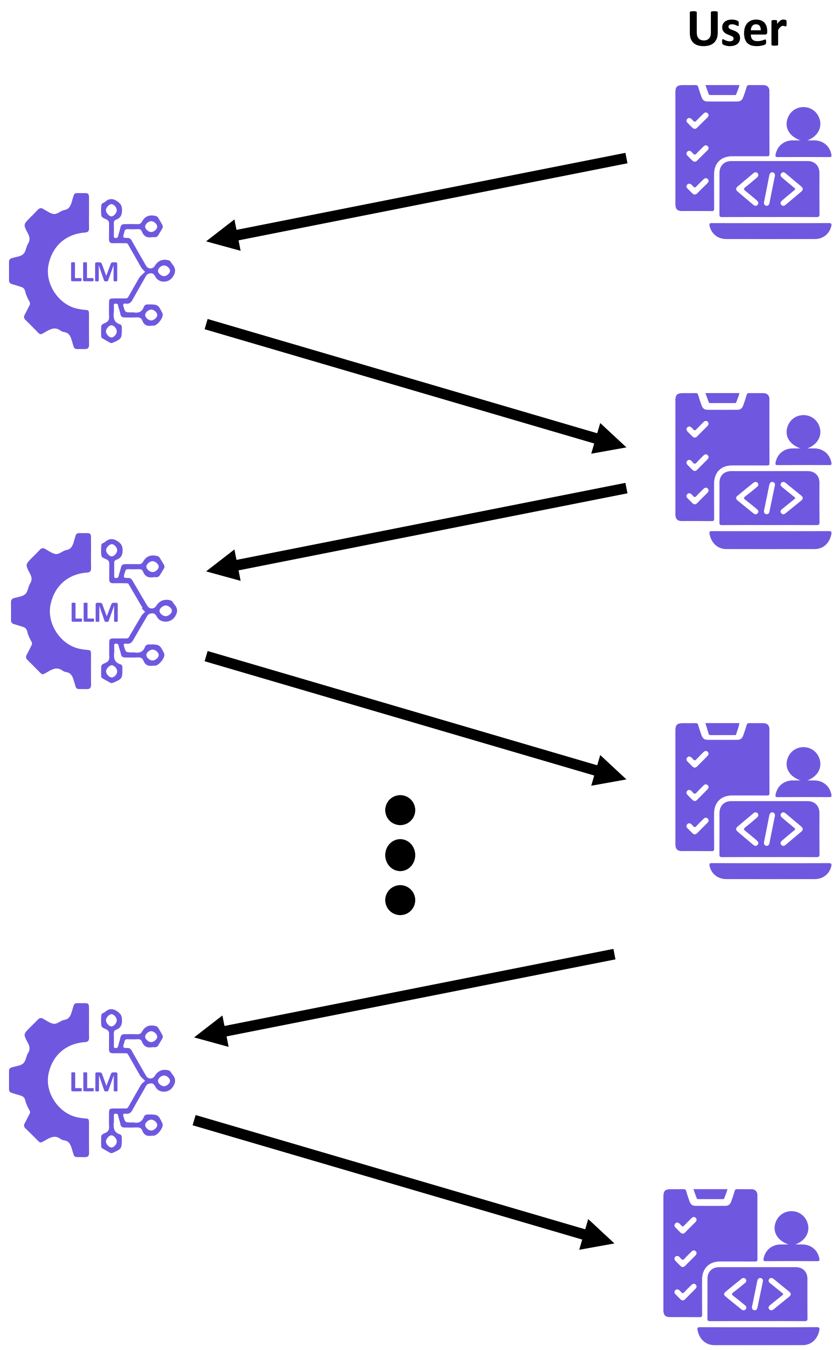
General purpose, ‘ok’ at everything



From **AI assistant** to **AI agents**

AI **Assistant**

General purpose, ‘ok’ at everything



Human’s daily **tasks**

1 min tasks

5 min tasks

30 min tasks

1 hour tasks

⋮

8 hours tasks

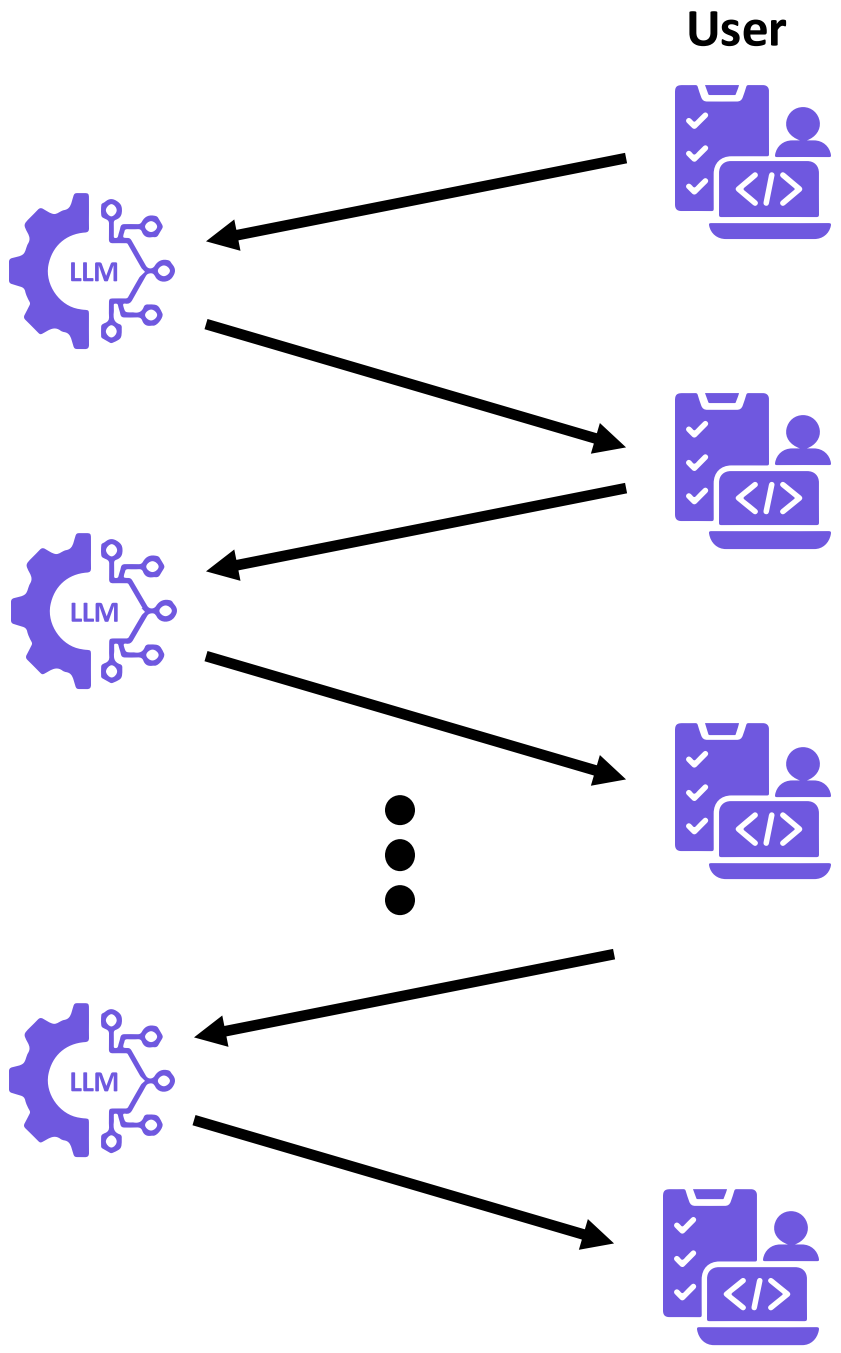
⋮

Complete a job

From **AI assistant** to **AI agents**

AI **Assistant**

General purpose, ‘ok’ at everything



Human’s daily **tasks**

1 min tasks

5 min tasks

30 min tasks

1 hour tasks

⋮

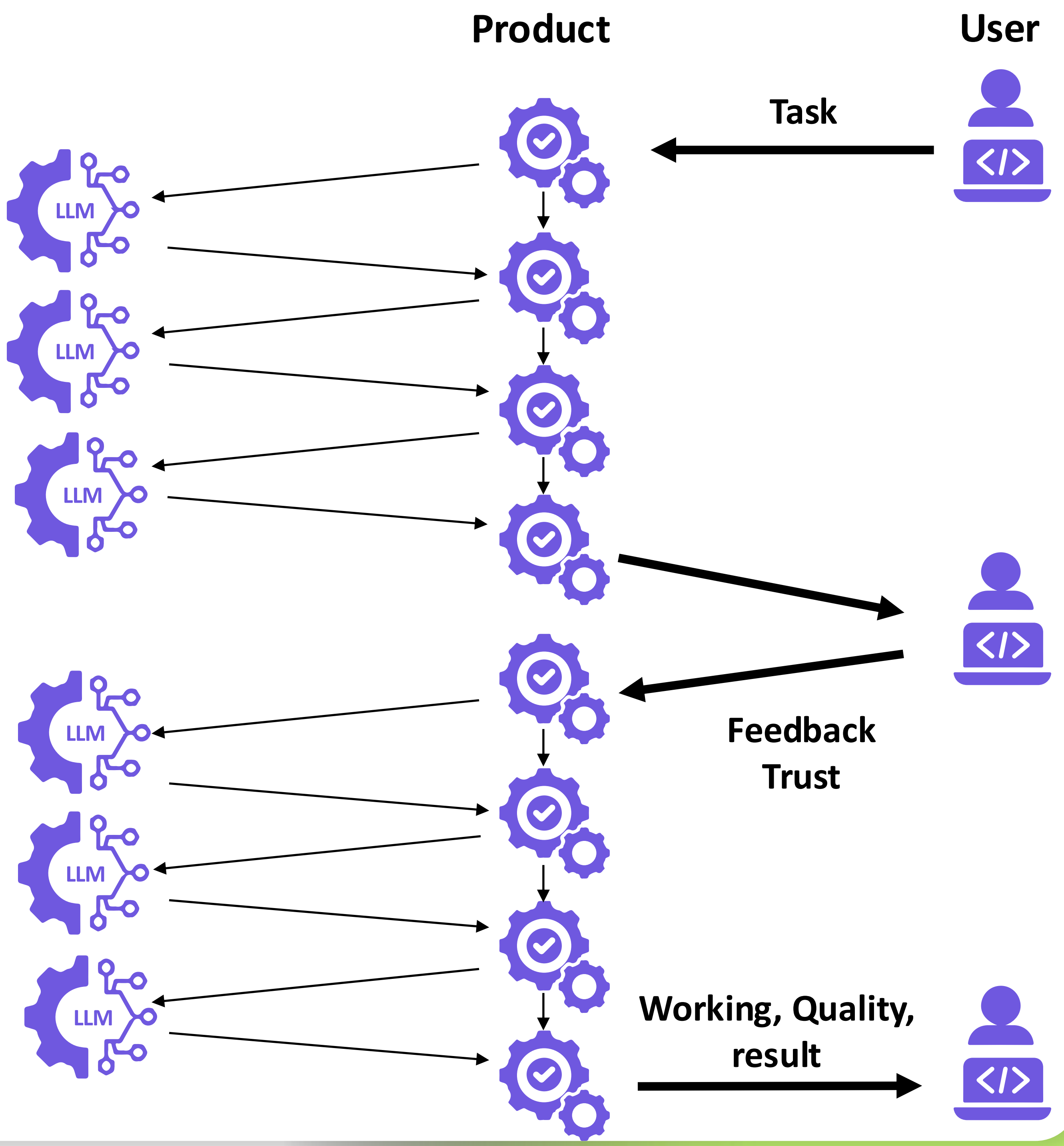
8 hours tasks

⋮

Complete a job

AI **Agent**

Expert at **Specific** task/s



From **test code assistant** to **test code agent**

Area	AI test code **Assistant**	AI test code **Agent**
Output	Code suggestion	Working code
Quality	Low to medium	High quality, comprehensive tests (happy path, edge cases, mocks)
Developers' effort	Consumes time	Saves time
Impact	**Productivity** 1 test suggestion at a time	**Transformational** Dozens of quality working unit tests Under a minutes

Use cases for unit test generation

Use Cases for using new DevTools:

1. Continuous testing during development – Test Early and often

2. Test your PR thoroughly at high converge before submitting

3. Testing private methods – Maintain encapsulation while finding hidden bugs

4. TDD done differently – Simplified and enhanced

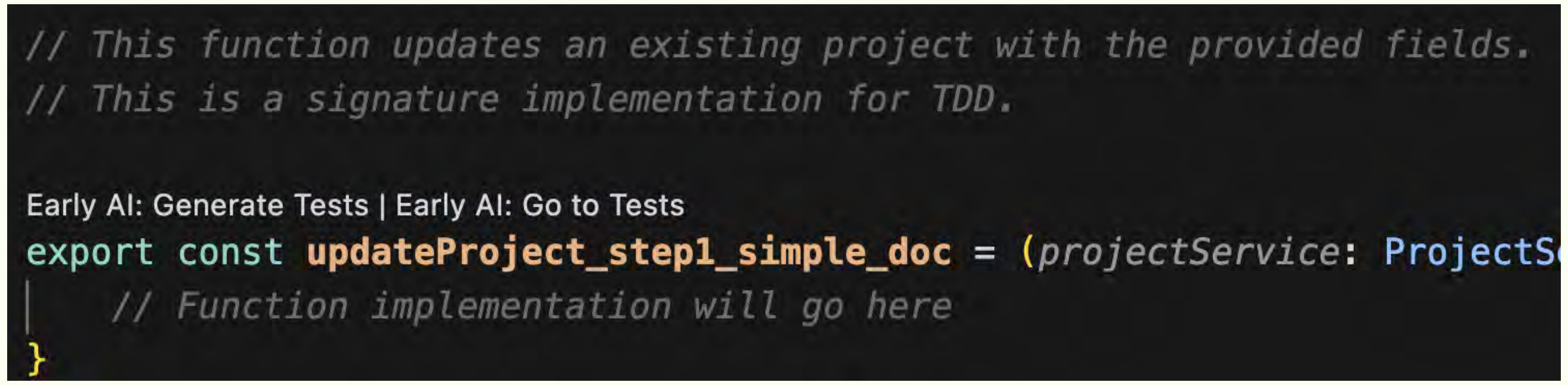
Mutation testing:

- Measuring the tests quality themselves

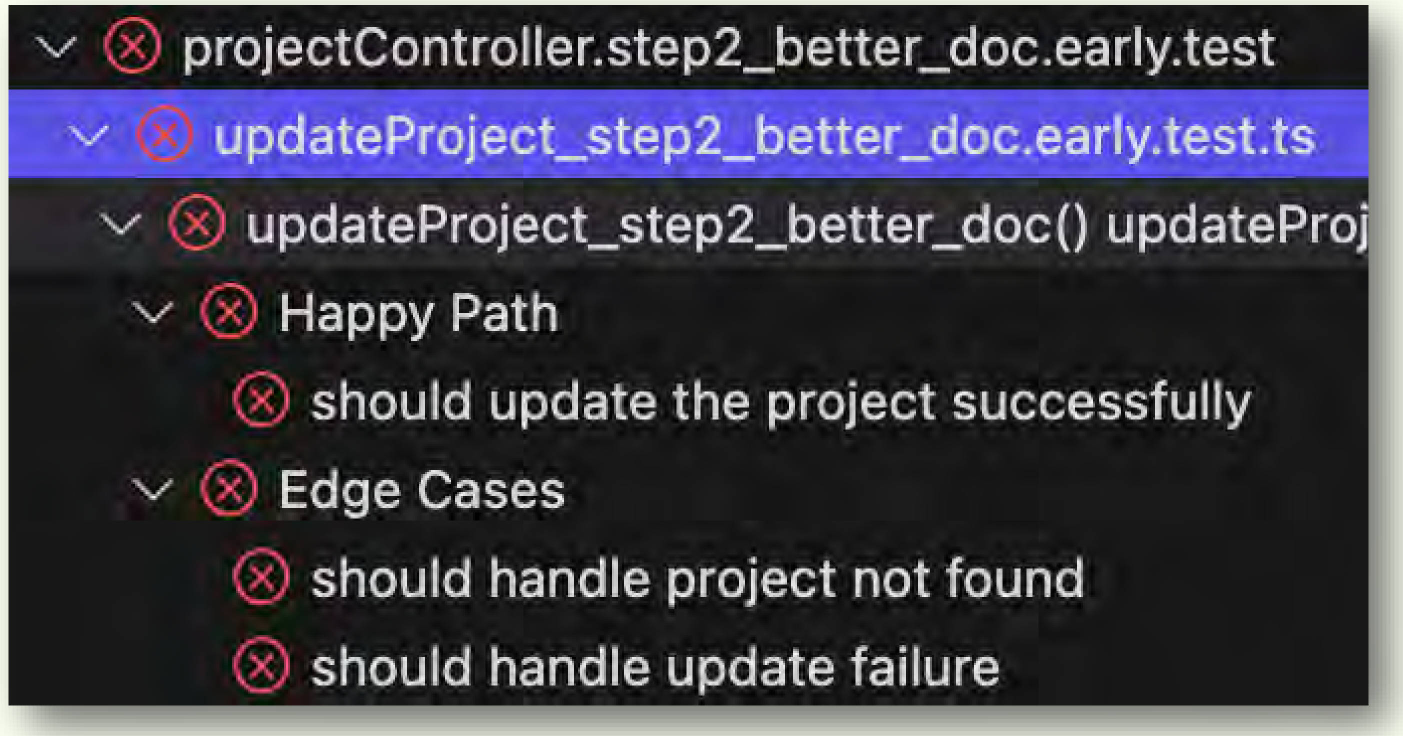
Use Case 3: TDD made easy

Method signature

Step 1: Simple method description



```
// This function updates an existing project with the provided fields.
// This is a signature implementation for TDD.

Early AI: Generate Tests | Early AI: Go to Tests
export const updateProject_step1_simple_doc = (projectService: ProjectS
    // Function implementation will go here
}
```

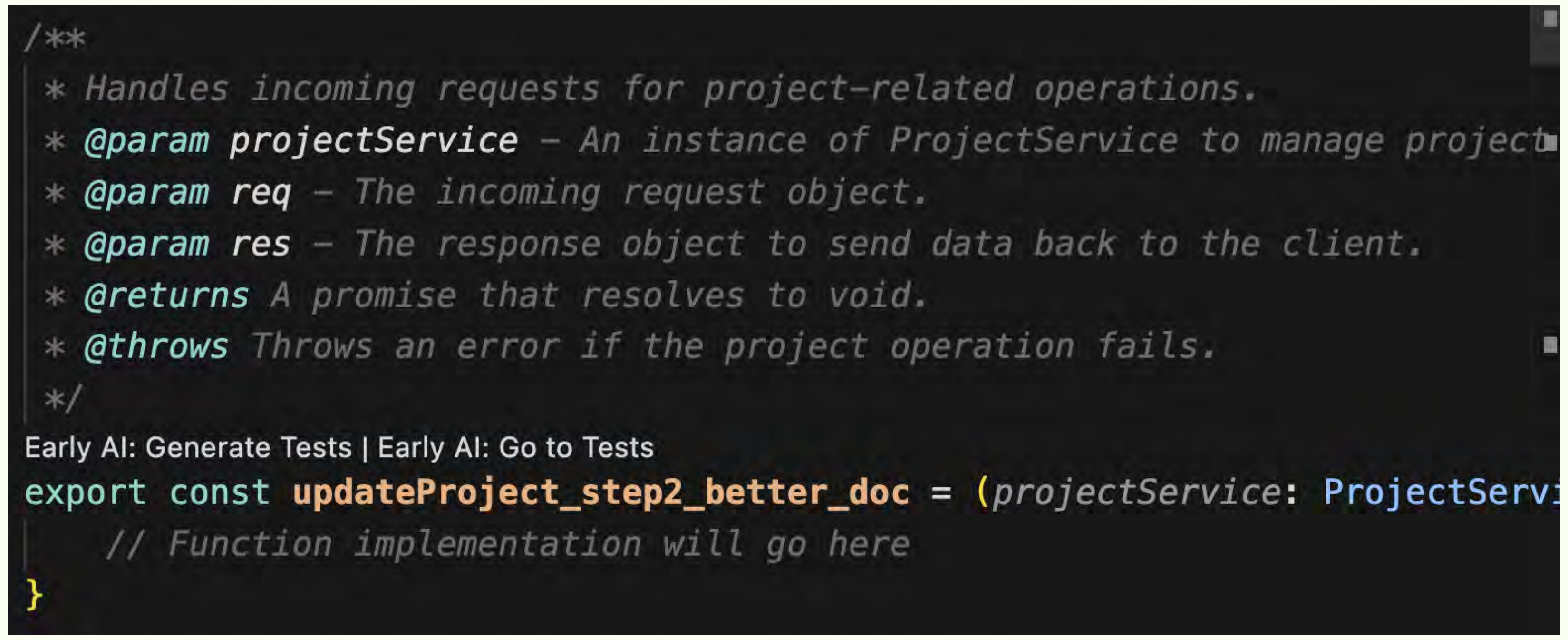


```
⌄ ⊗ projectController.step2_better_doc.early.test
  ⌄ ⊗ updateProject_step2_better_doc.early.test.ts
    ⌄ ⊗ updateProject_step2_better_doc() updateProj
      ⌄ ⊗ Happy Path
          ⊗ should update the project successfully
      ⌄ ⊗ Edge Cases
          ⊗ should handle project not found
          ⊗ should handle update failure
```

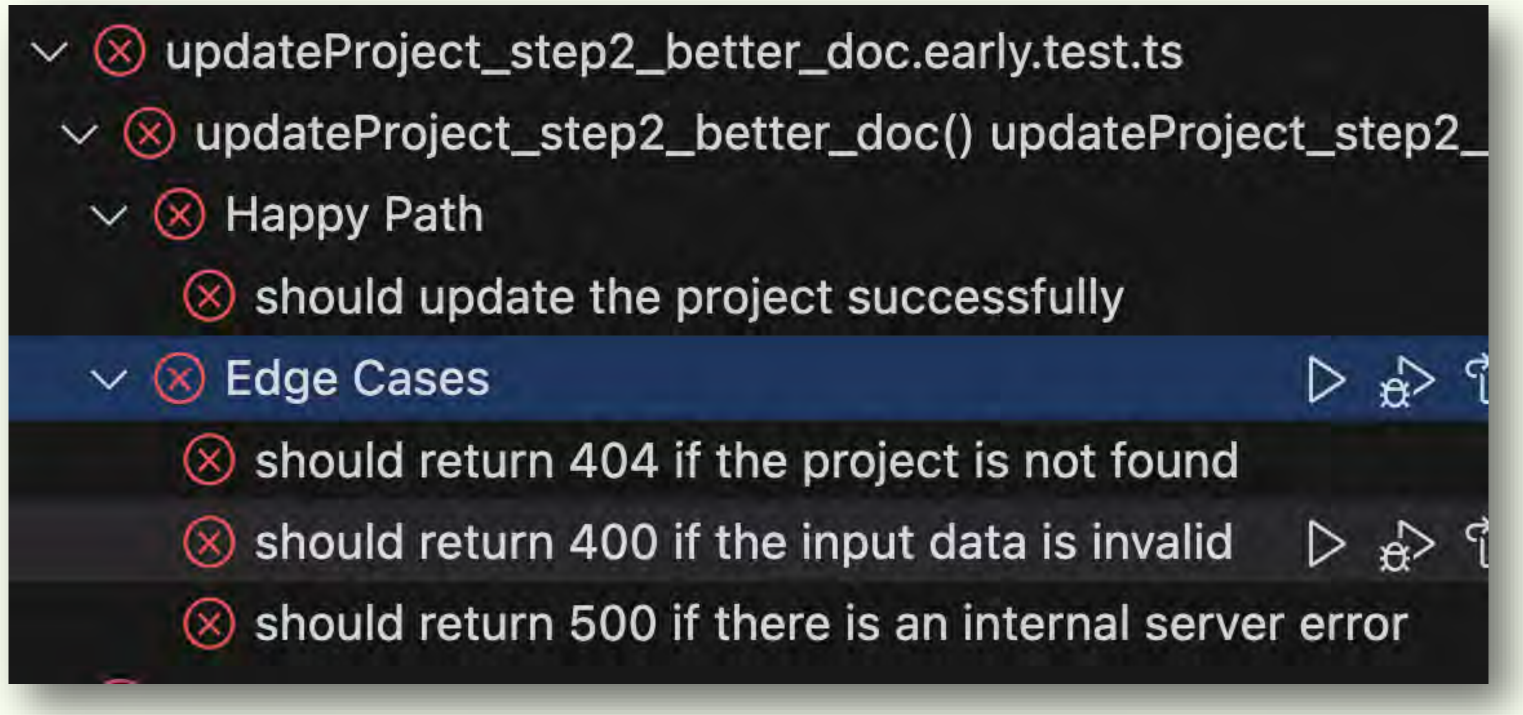
Step 1: Basic Red unit tests

Method signature

Step 2: Ellaborated documentation



```
/**
 * Handles incoming requests for project-related operations.
 * @param projectService - An instance of ProjectService to manage project
 * @param req - The incoming request object.
 * @param res - The response object to send data back to the client.
 * @returns A promise that resolves to void.
 * @throws Throws an error if the project operation fails.
 */
Early AI: Generate Tests | Early AI: Go to Tests
export const updateProject_step2_better_doc = (projectService: ProjectServ
    // Function implementation will go here
}
```



```
⌄ ⊗ updateProject_step2_better_doc.early.test.ts
  ⌄ ⊗ updateProject_step2_better_doc() updateProject_step2_
    ⌄ ⊗ Happy Path
        ⊗ should update the project successfully
    ⌄ ⊗ Edge Cases
        ⊗ should return 404 if the project is not found
        ⊗ should return 400 if the input data is invalid
        ⊗ should return 500 if there is an internal server error
```

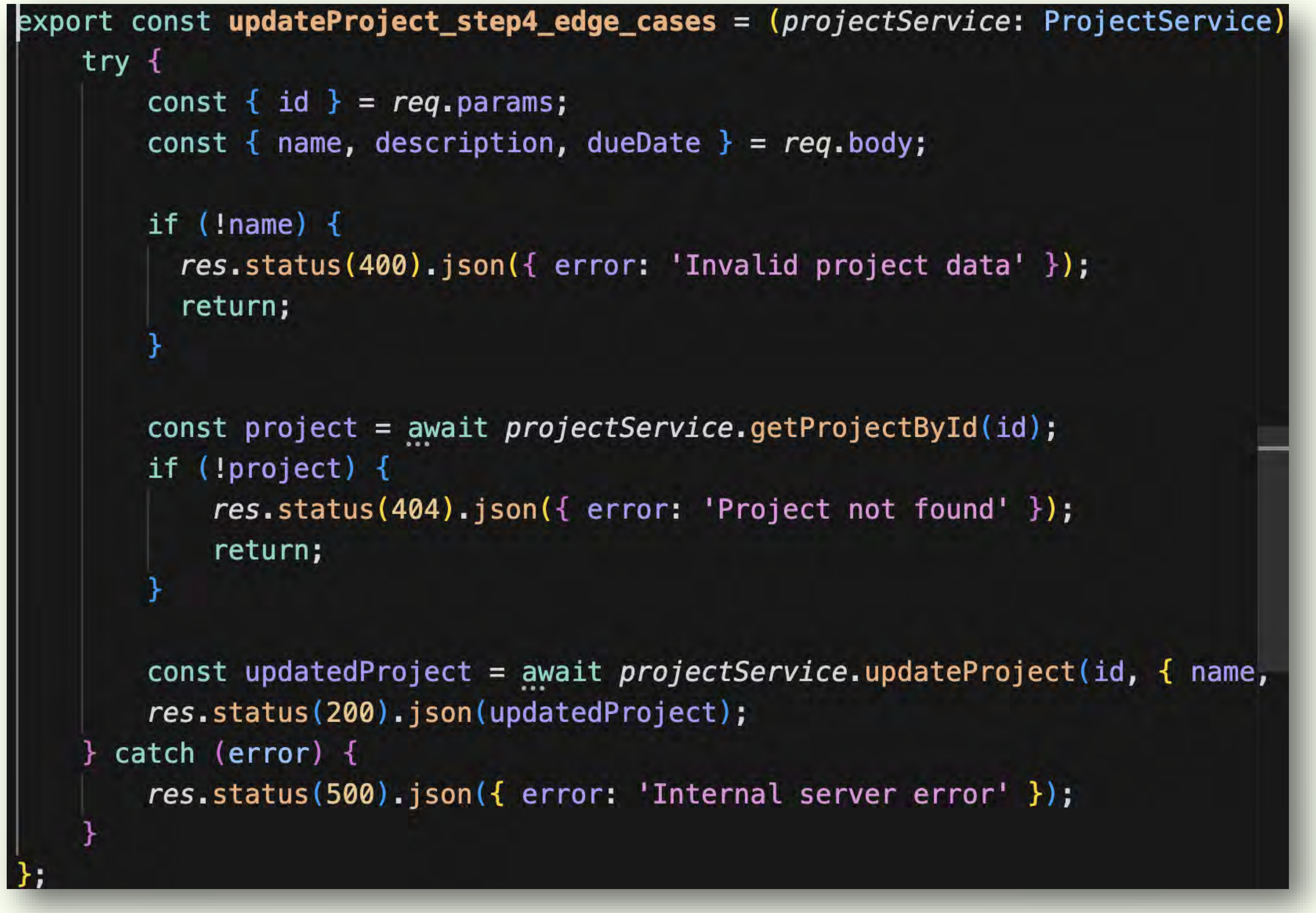
Step 2: Specific Red unit tests

Use Case 3: TDD made easy

Implementation

Step 3: Happy path

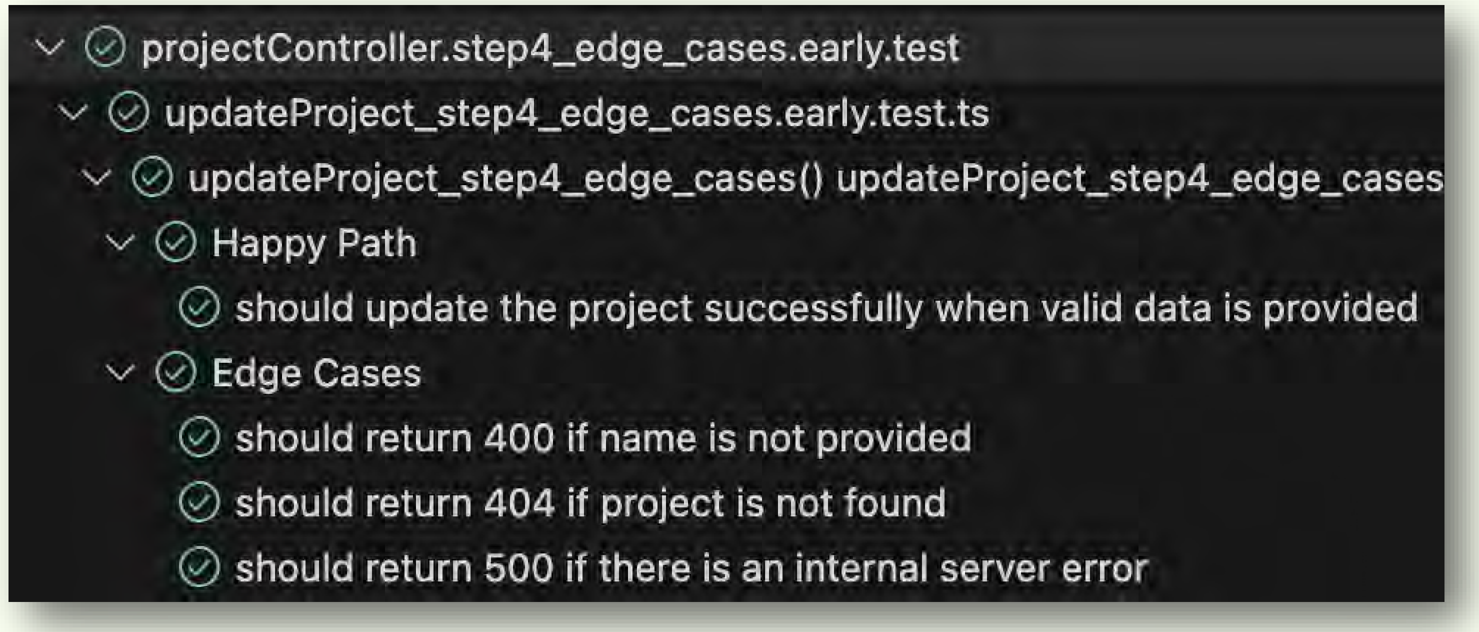
Step 4: Edge cases



```
export const updateProject_step4_edge_cases = (projectService: ProjectService)
    try {
        const { id } = req.params;
        const { name, description, dueDate } = req.body;

        if (!name) {
          res.status(400).json({ error: 'Invalid project data' });
          return;
        }

        const project = await projectService.getProjectById(id);
        if (!project) {
            res.status(404).json({ error: 'Project not found' });
            return;
        }

        const updatedProject = await projectService.updateProject(id, { name,
        res.status(200).json(updatedProject);
    } catch (error) {
        res.status(500).json({ error: 'Internal server error' });
    }
};
```

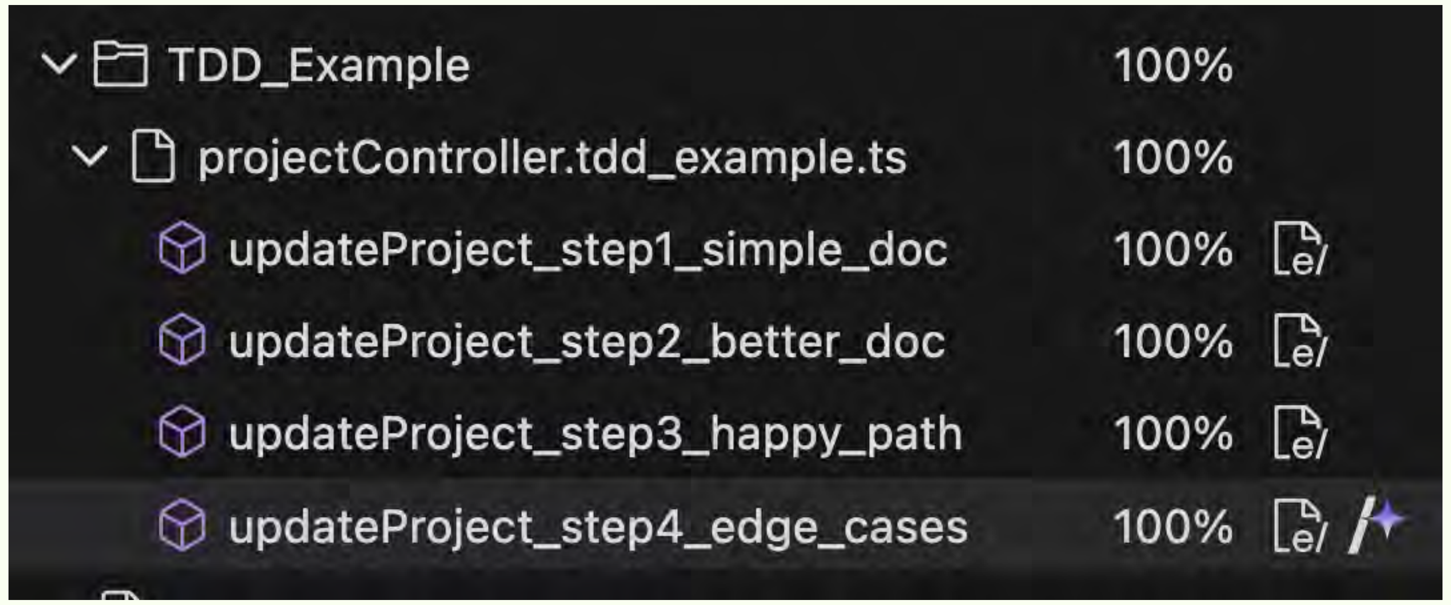
Code quality outcome

Step 4: full coverage



```
⌄ ✓ projectController.step4_edge_cases.early.test
  ⌄ ✓ updateProject_step4_edge_cases.early.test.ts
    ⌄ ✓ updateProject_step4_edge_cases() updateProject_step4_edge_cases
      ⌄ ✓ Happy Path
          ✓ should update the project successfully when valid data is provided
      ⌄ ✓ Edge Cases
          ✓ should return 400 if name is not provided
          ✓ should return 404 if project is not found
          ✓ should return 500 if there is an internal server error
```

Green unit tests



```
⌄ TDD_Example                                   100%
  ⌄ projectController.tdd_example.ts            100%
      updateProject_step1_simple_doc            100%
      updateProject_step2_better_doc            100%
      updateProject_step3_happy_path            100%
      updateProject_step4_edge_cases            100%
```

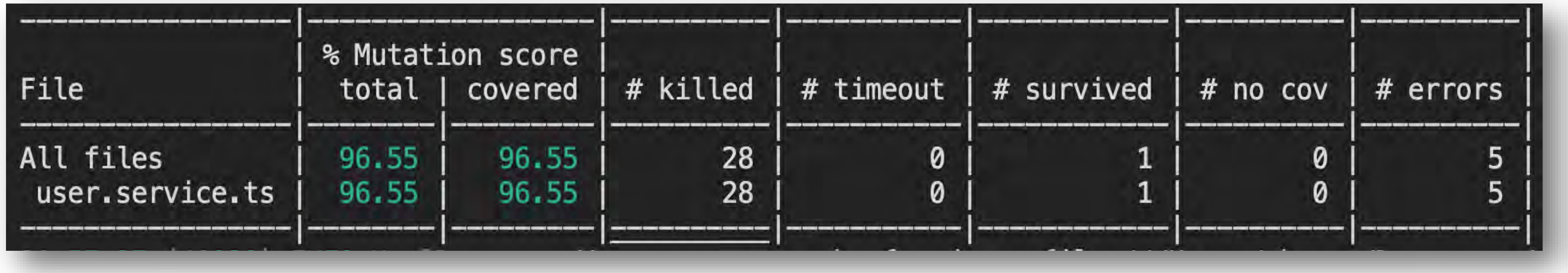
100% code coverage

Who tests the tests – Mutation testing

High coverage != High Quality

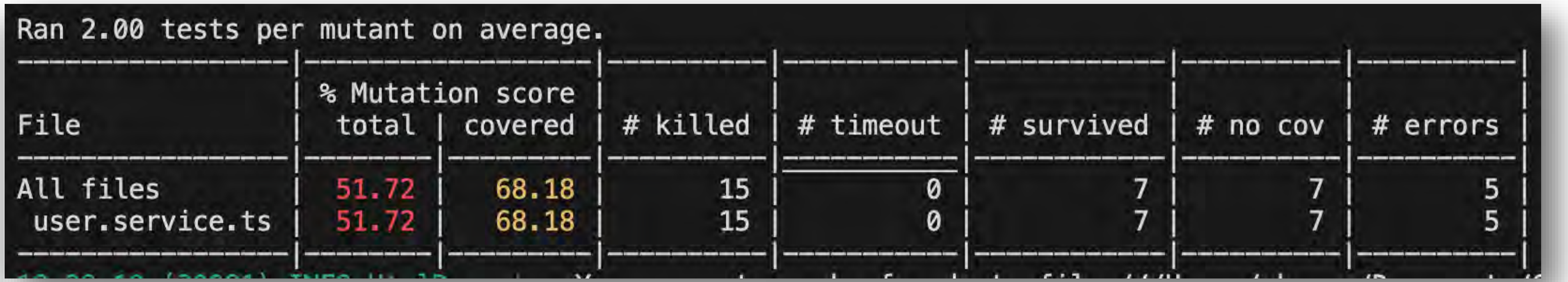
- Mutation testing introduces changes to your code
- Then runs your unit tests against the changed code
- It is expected that your unit tests will now fail
- If they don't fail, it might indicate your tests do not sufficiently cover the code

$$\text{Mutation score} = \frac{\text{Killed mutants + Timed-out mutants}}{\text{All mutants – Error mutants}} \text{ X 100}$$



```
-------------------|------------------|-----------|------------|------------|------------|----------|
                   | % Mutation score |           |            |            |            |          |
File               |  total | covered | # killed  | # timeout  | # survived | # no cov   | # errors |
-------------------|--------|---------|-----------|------------|------------|------------|----------|
All files          |  96.55 |   96.55 |        28 |          0 |          1 |          0 |        5 |
 user.service.ts   |  96.55 |   96.55 |        28 |          0 |          1 |          0 |        5 |
-------------------|--------|---------|-----------|------------|------------|------------|----------|
```

☺ Coverage 100%, Mutaion Score **96.55%**

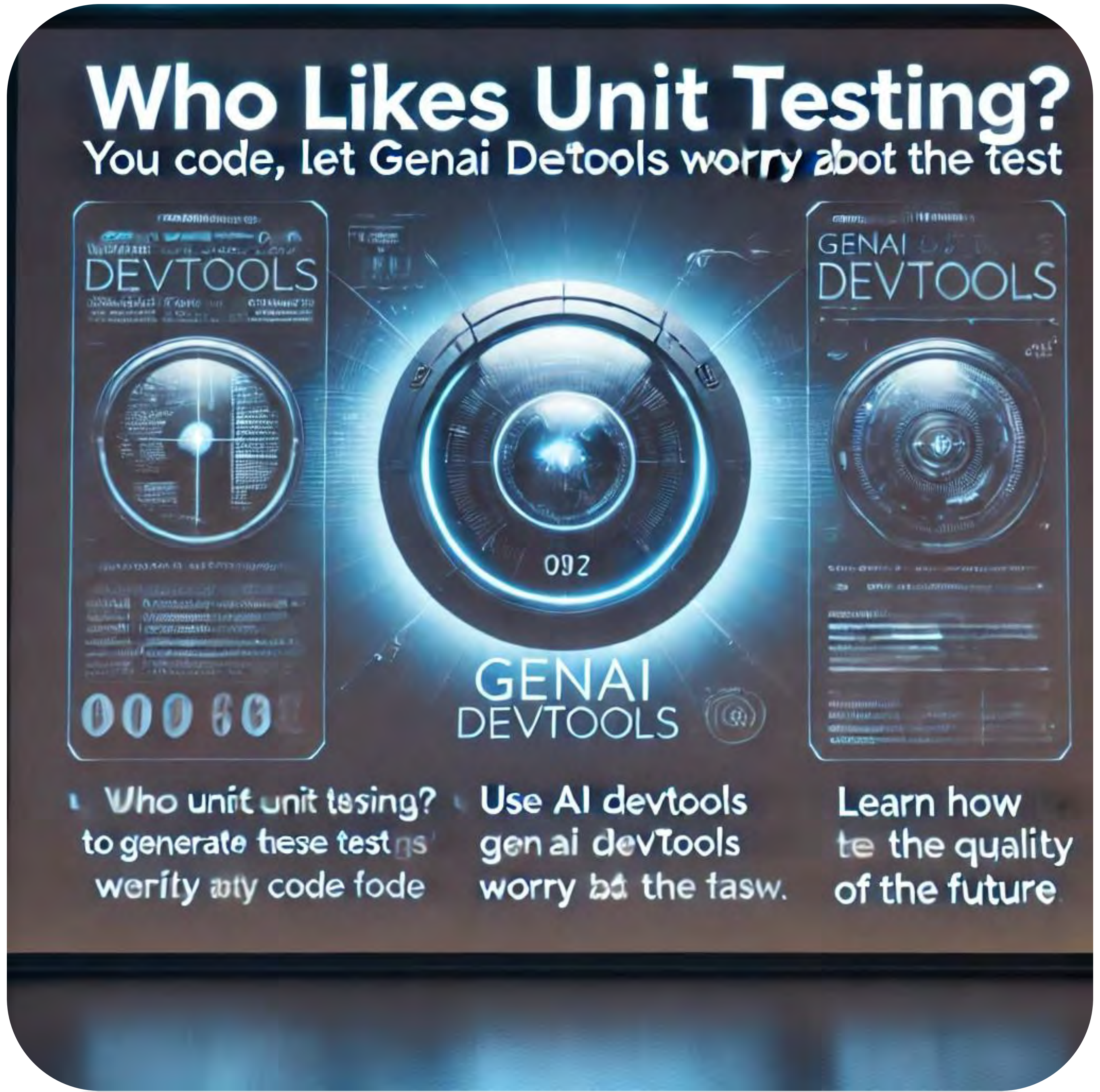


```
Ran 2.00 tests per mutant on average.
-----------------|------------------|----------|-----------|------------|----------|----------|
                 | % Mutation score |          |           |            |          |          |
File             |  total | covered | # killed | # timeout | # survived | # no cov | # errors |
-----------------|--------|---------|----------|-----------|------------|----------|----------|
All files        |  51.72 |   68.18 |       15 |         0 |          7 |        7 |        5 |
 user.service.ts |  51.72 |   68.18 |       15 |         0 |          7 |        7 |        5 |
-----------------|--------|---------|----------|-----------|------------|----------|----------|
```

☹ Coverage 84%, Mutaion Score **51.72%**

Summary

- Testing has never been so Easy
- Use AI DevTools to generate test code for you
- Learn how to embed these tools in your development flow
- Deliver high quality code faster
- Become the 100x engineer of the future


Who Likes Unit Testing?
You code, let Genai Detools worry abot the test
DEVTOOLS
GENAI DEVTOOLS
092
000 60
GENAI DEVTOOLS
Who unit unit tesing? to generate these test werity aty code fode
Use AI devtools gen ai devTools worry bd the fasw.
Learn how te the quality of the future

Thank You

Sharon Barr
sharon@startearly.ai