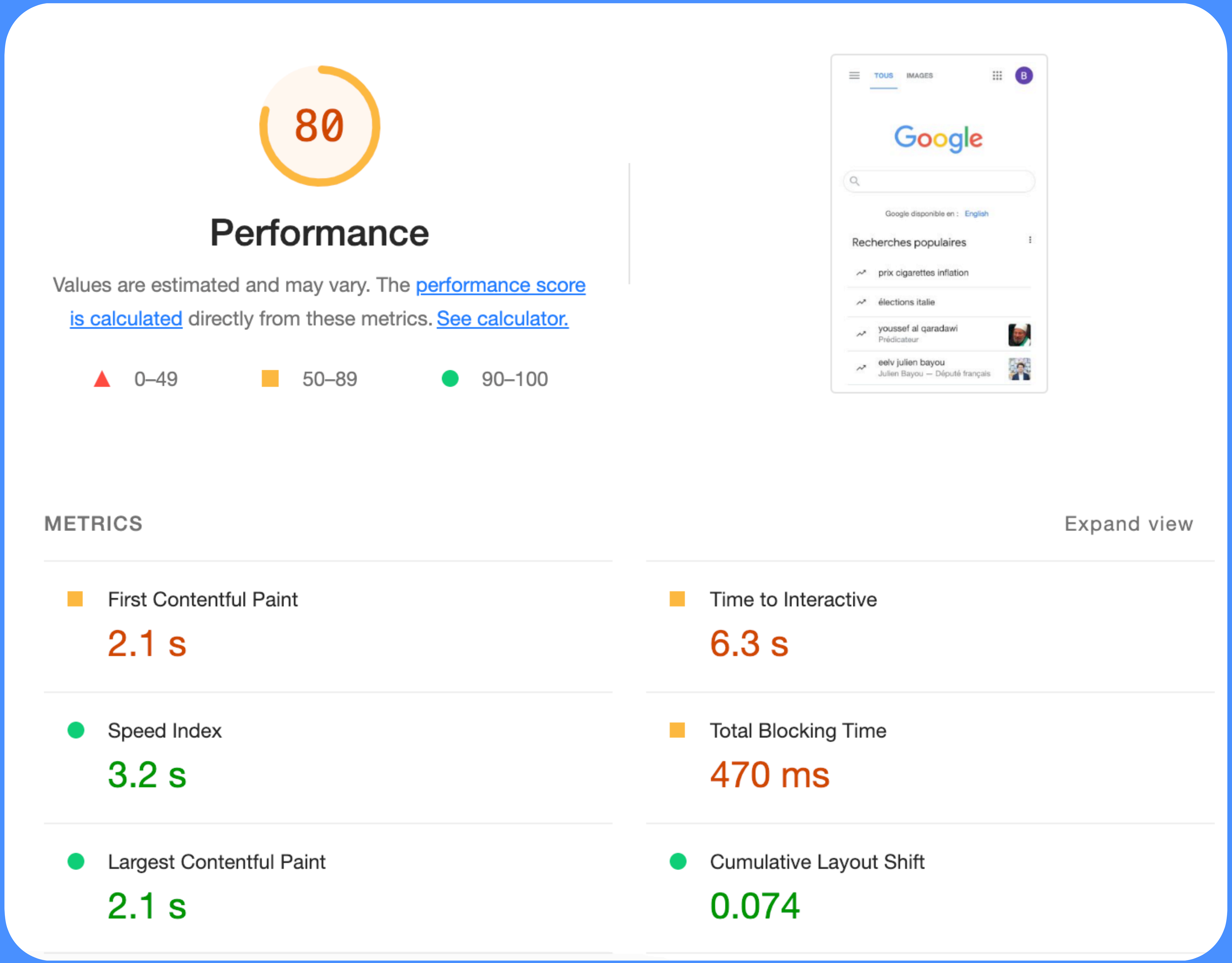
# Let’s start with a short story

# A client with measurable performance requirements

On web: 90+ Lighthouse score on desktop 🔥

# Example Lighthouse Report



80

## Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS

Expand view

■ First Contentful Paint
2.1 s

■ Time to Interactive
6.3 s

● Speed Index
3.2 s

■ Total Blocking Time
470 ms

● Largest Contentful Paint
2.1 s

● Cumulative Layout Shift
0.074

# A client with measurable performance requirements

On web: 90+ Lighthouse score on desktop 🔥

On mobile: “It should not lag” 😅

# A client with measurable performance requirements

On web: 90+ Lighthouse score on desktop 🔥

On mobile: “It should not lag” 😅

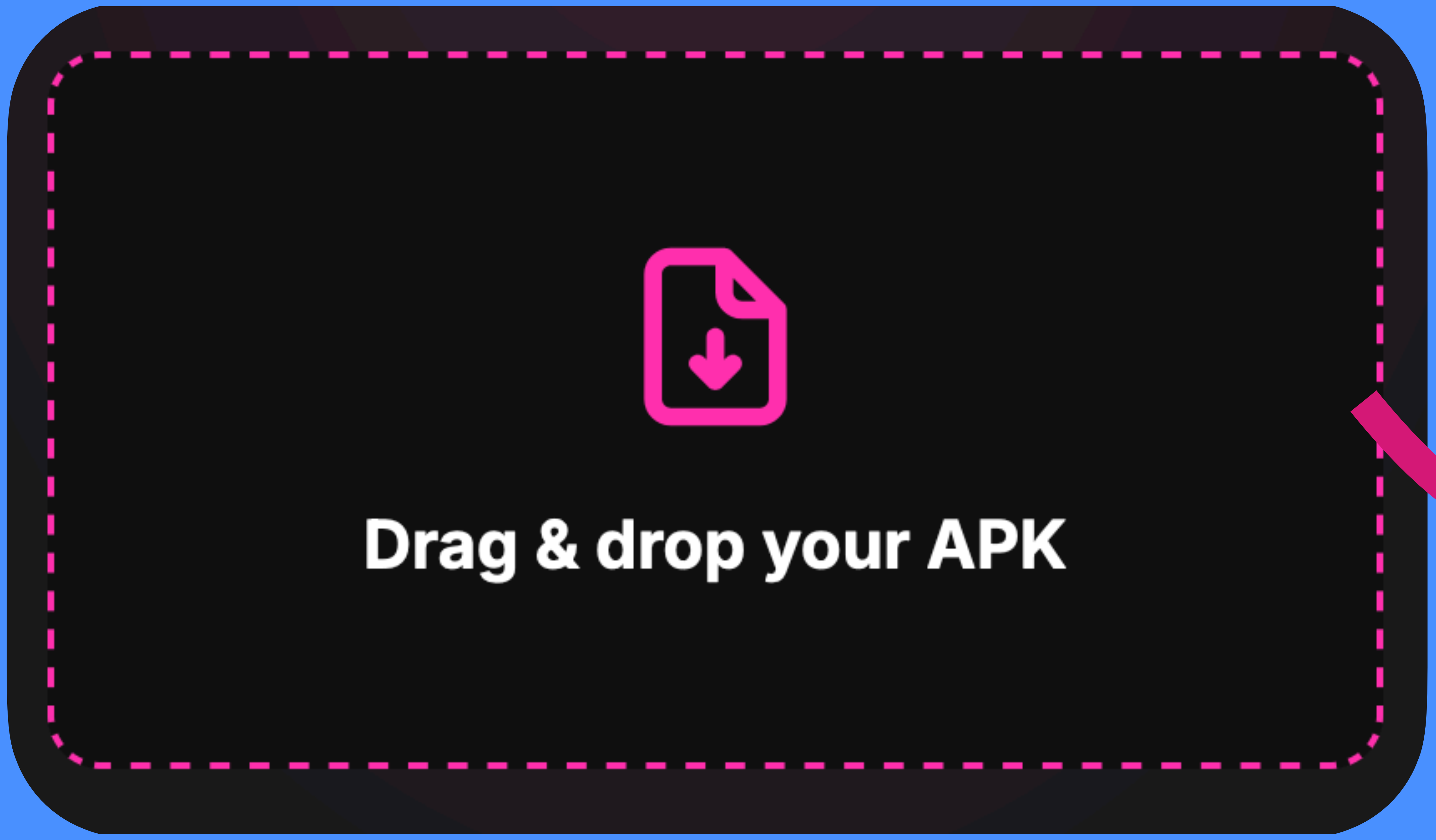
# How do I know if my app has good performance? 🤔

# Alexandre Moureaux

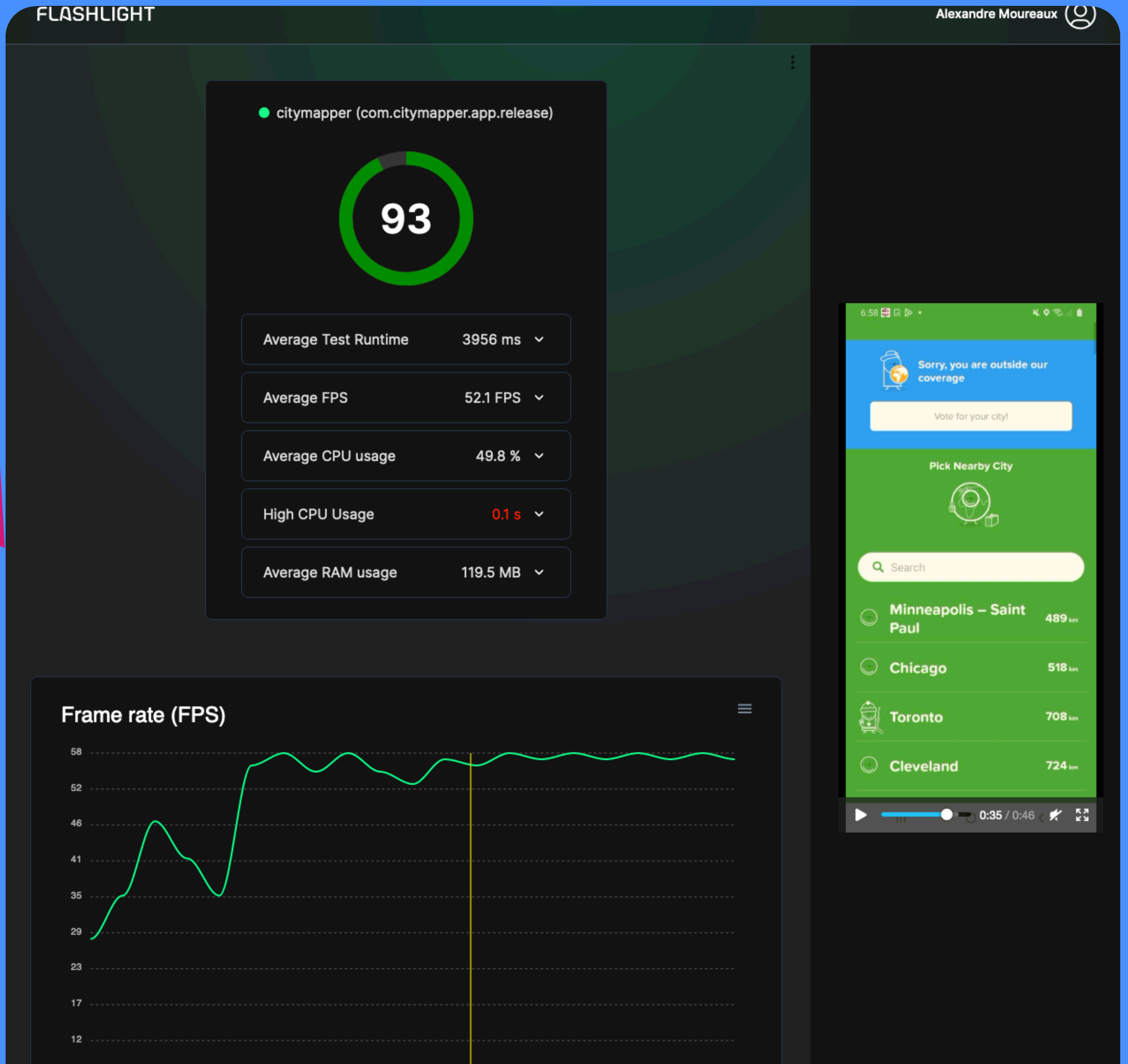
## App performance expert at theodo. Apps

in @almouro

# FLASHLIGHT 🔦, a mobile lighthouse



Drag & drop your APK



FLASHLIGHT

Alexandre Moureaux

● citymapper (com.citymapper.app.release)

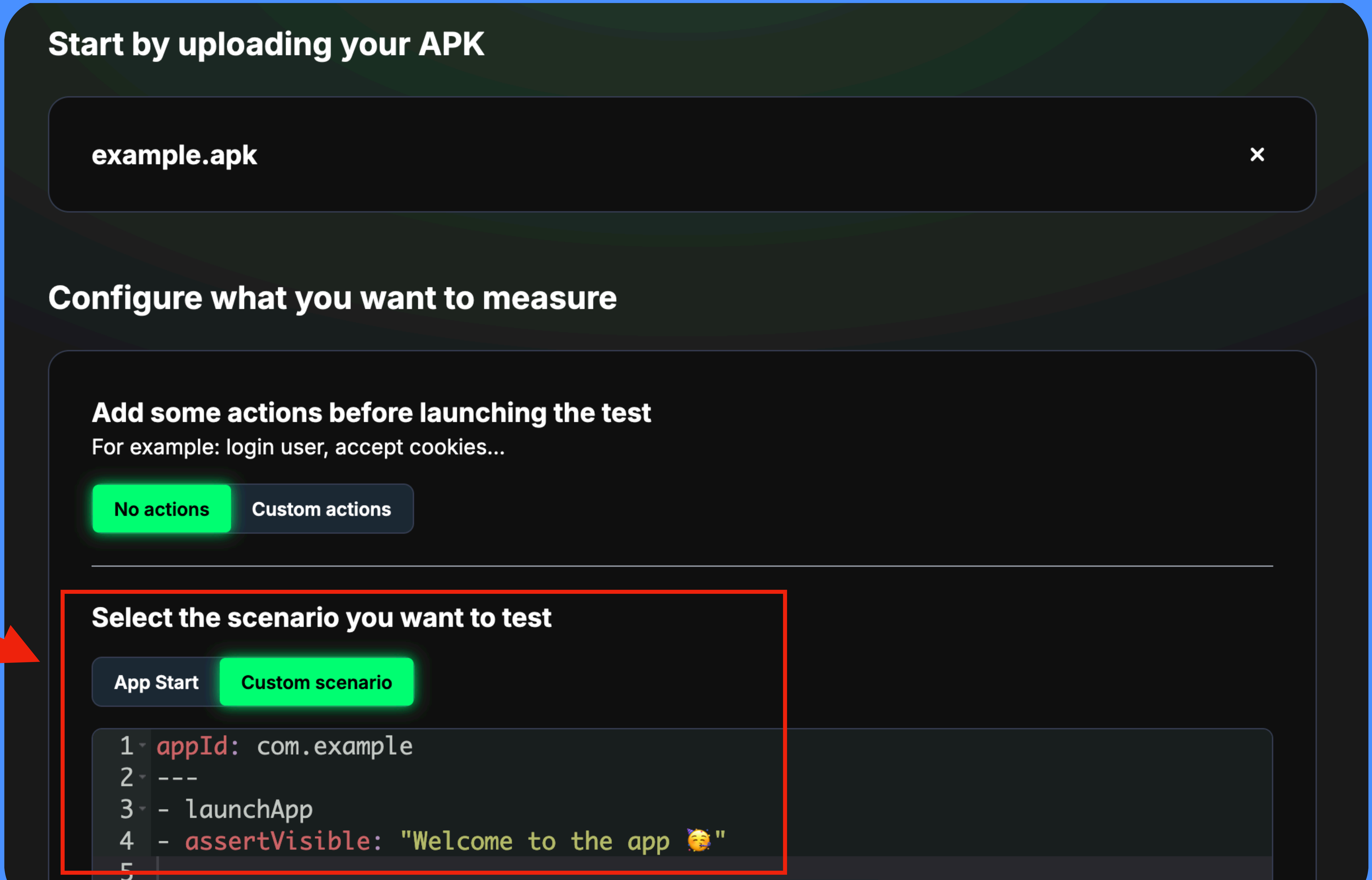
93

| | |
|---|---|
| Average Test Runtime | 3956 ms |
| Average FPS | 52.1 FPS |
| Average CPU usage | 49.8 % |
| High CPU Usage | 0.1 s |
| Average RAM usage | 119.5 MB |

Frame rate (FPS)

# Get more than a startup score with E2E tests

End to end tests



## Start by uploading your APK

example.apk ×

## Configure what you want to measure

### Add some actions before launching the test

For example: login user, accept cookies...

[✓ No actions] Custom actions

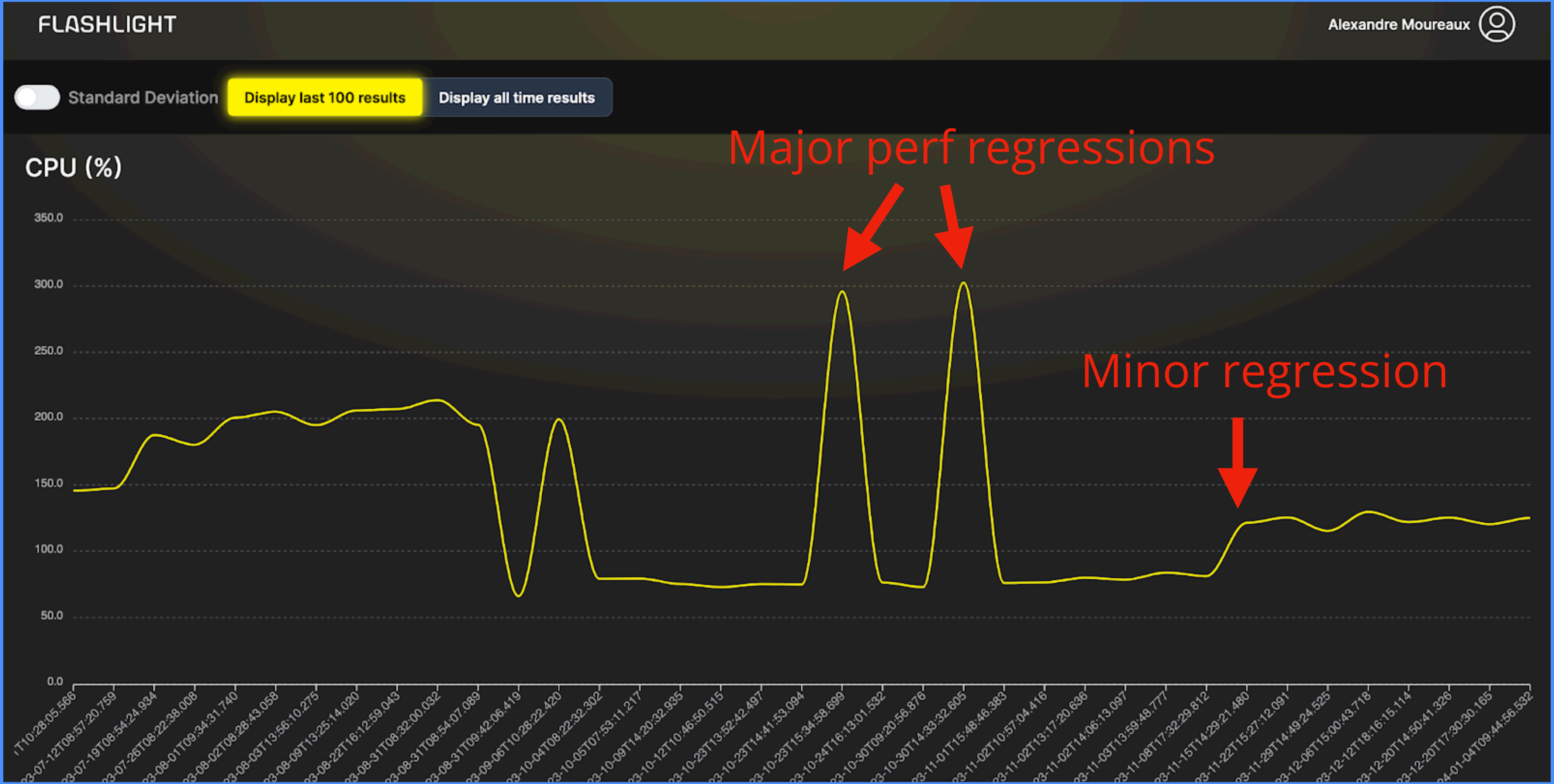
### Select the scenario you want to test

App Start [✓ Custom scenario]

```yaml
appId: com.example
---
- launchApp
- assertVisible: "Welcome to the app 🥳"

```

# Performance Regression dashboard



Major perf regressions

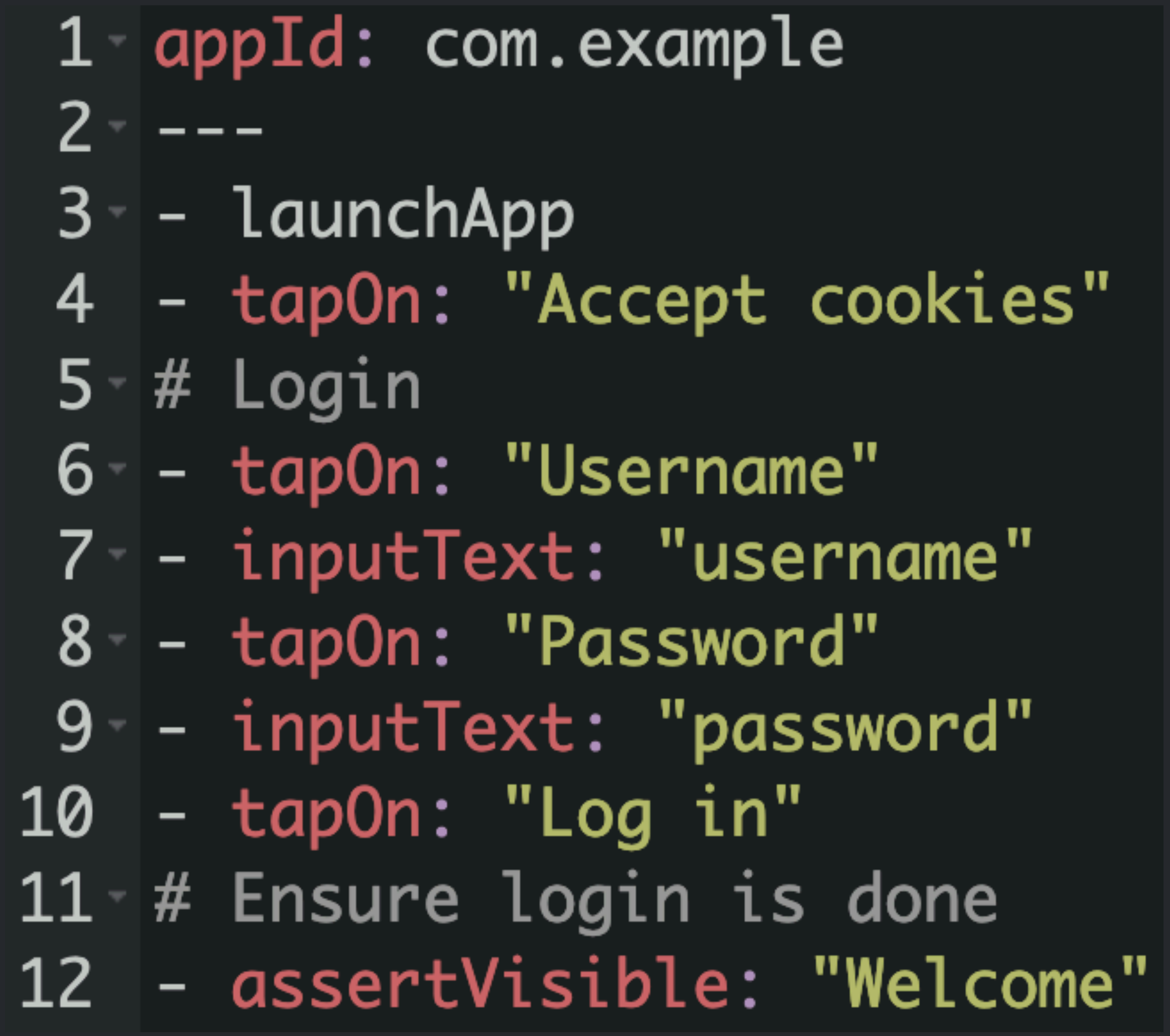
Minor regression

We’ve made E2E mobile performance testing
as easy as possible 🥳

So… what’s the problem? 🤔

Writing a test can be
as simple as this

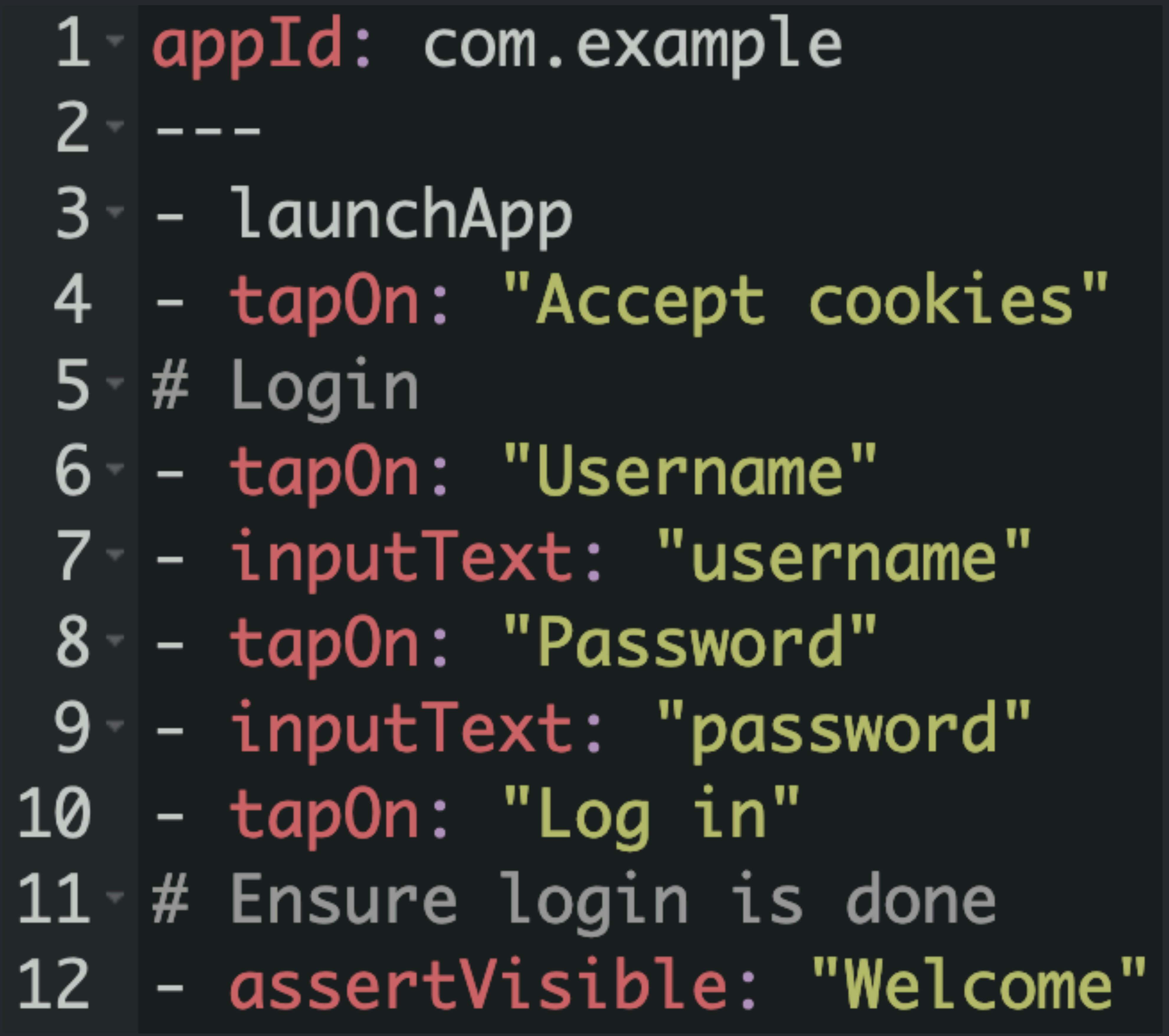
⬇️



```yaml
appId: com.example
---
- launchApp
- tapOn: "Accept cookies"
# Login
- tapOn: "Username"
- inputText: "username"
- tapOn: "Password"
- inputText: "password"
- tapOn: "Log in"
# Ensure login is done
- assertVisible: "Welcome"
```

Writing a test can be
as simple as this

⬇️



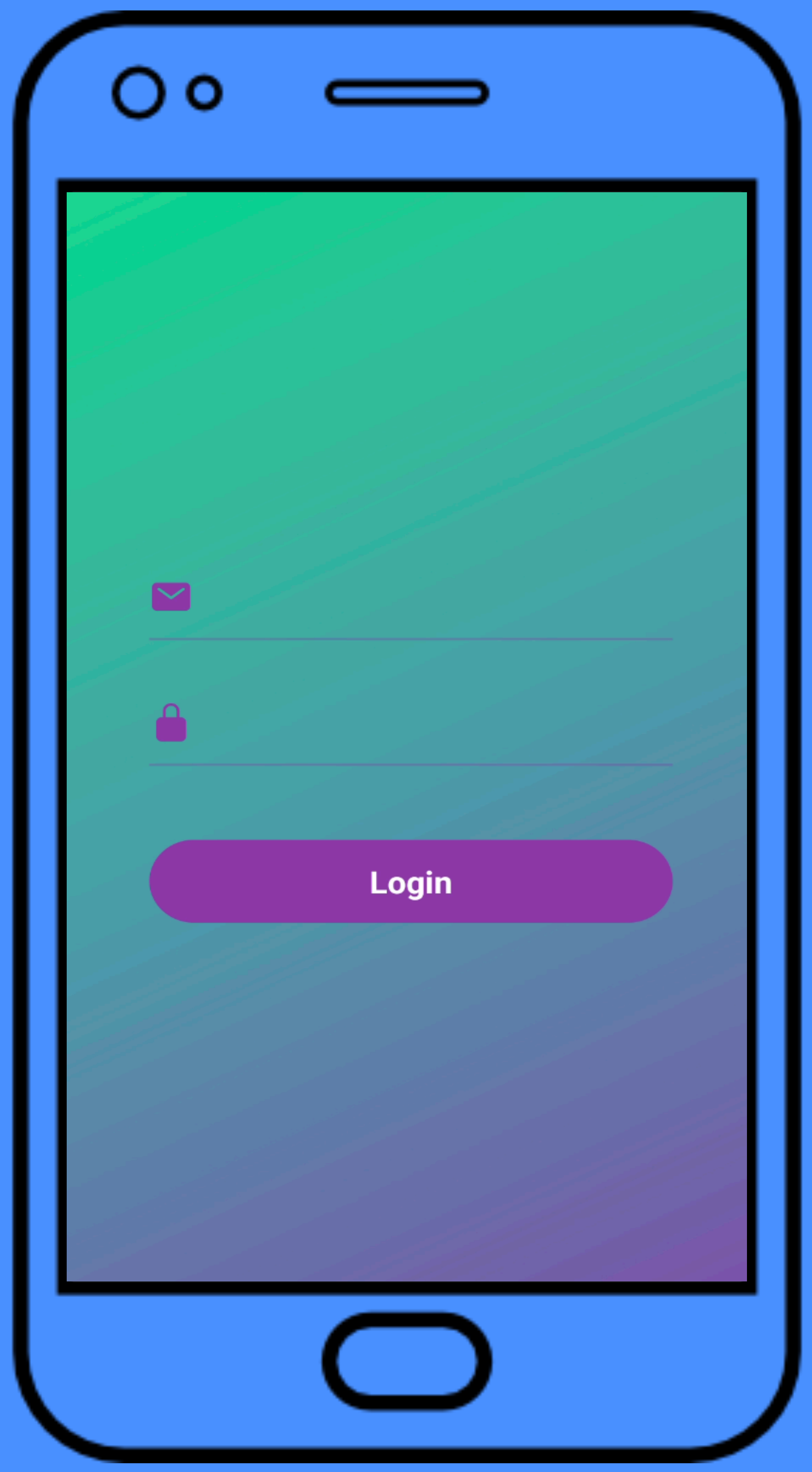
```
appId: com.example
---
- launchApp
- tapOn: "Accept cookies"
# Login
- tapOn: "Username"
- inputText: "username"
- tapOn: "Password"
- inputText: "password"
- tapOn: "Log in"
# Ensure login is done
- assertVisible: "Welcome"
```

## And yet...

💔 they can break easily

🛠️ can be tricky to maintain

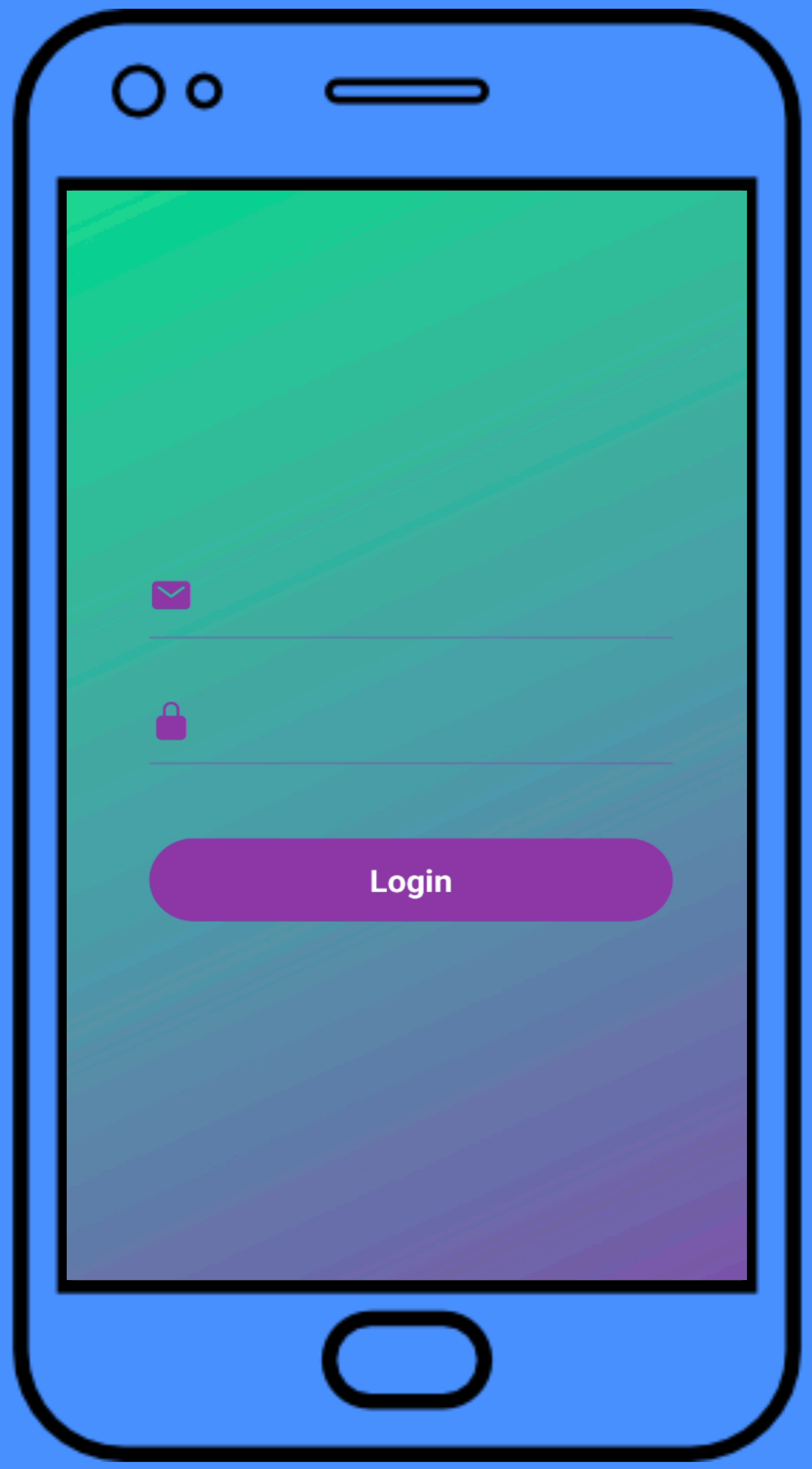
⏰ time is still needed to write them



Explore the app 🤔
- manually
- or via an end-to-end test

Collect Performance Data ✅
(%CPU, FPS, RAM...)

Detect issues ✅
(low FPS, high JS thread usage)



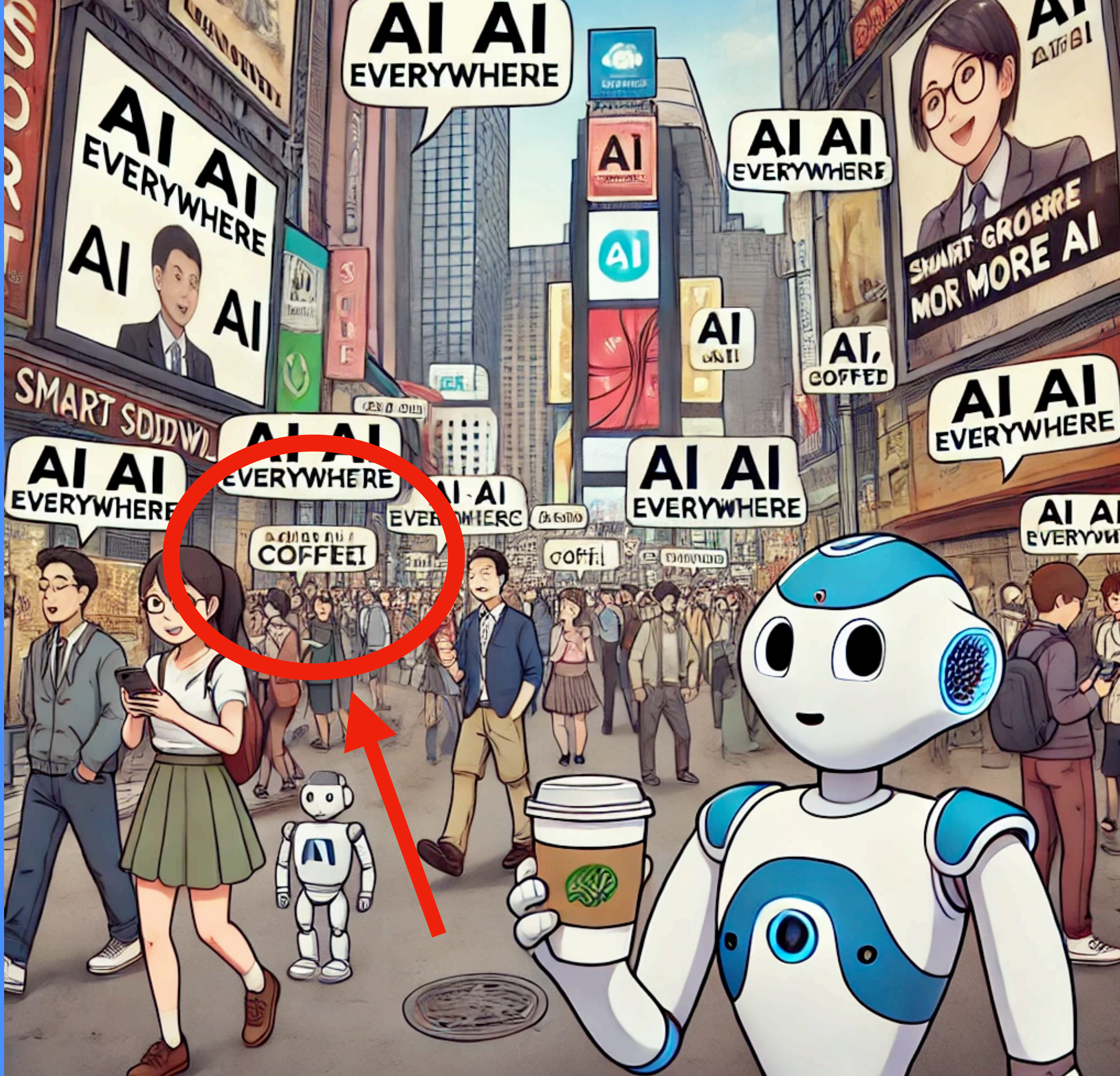
Explore the app 🤔
- manually
- or via an end-to-end test

Collect Performance Data ✅
(%CPU, FPS, RAM...)

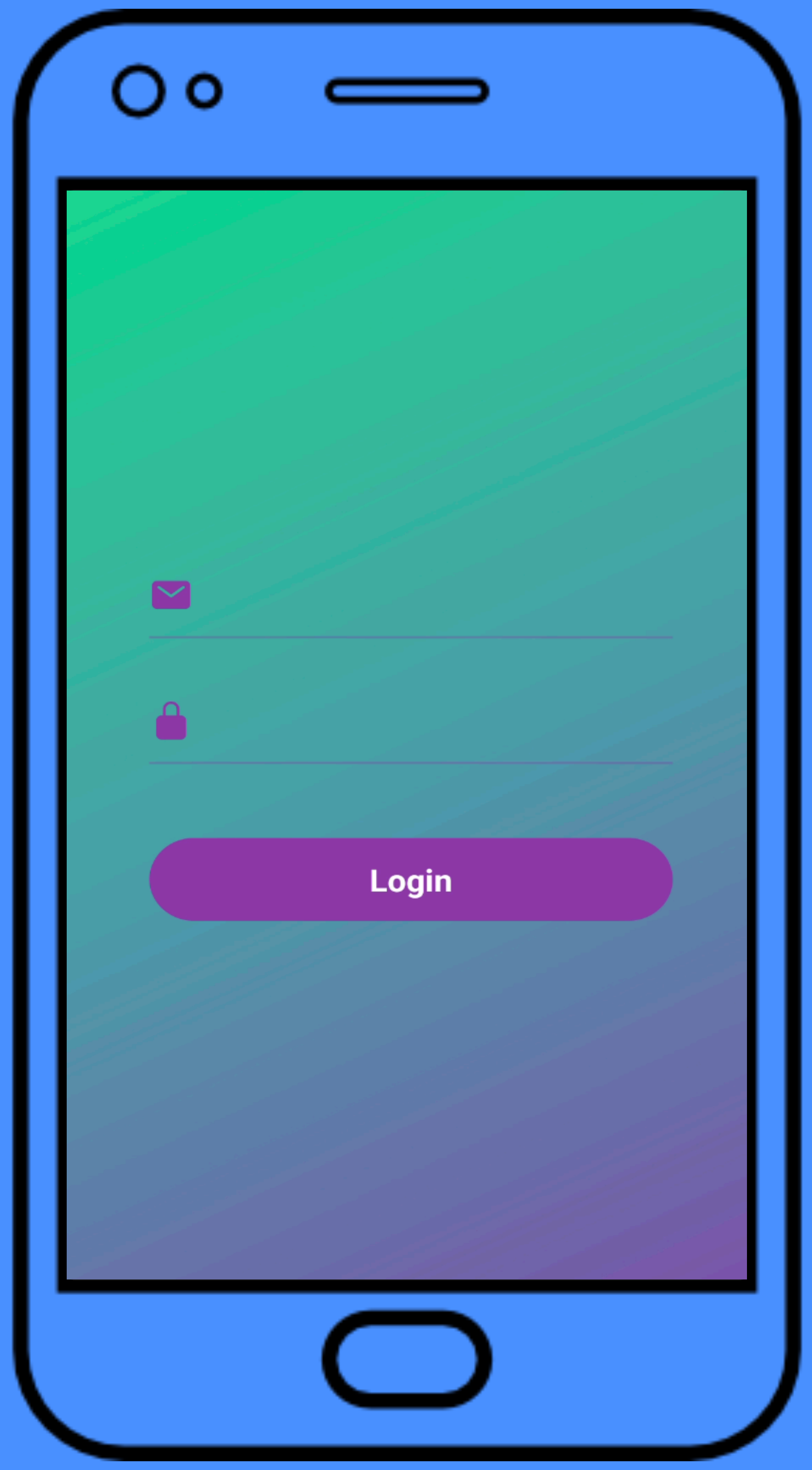
Detect issues ✅
(low FPS, high JS thread usage)

# 2024 in a nutshell

# AI everywhere Meme - ChatGPT edition



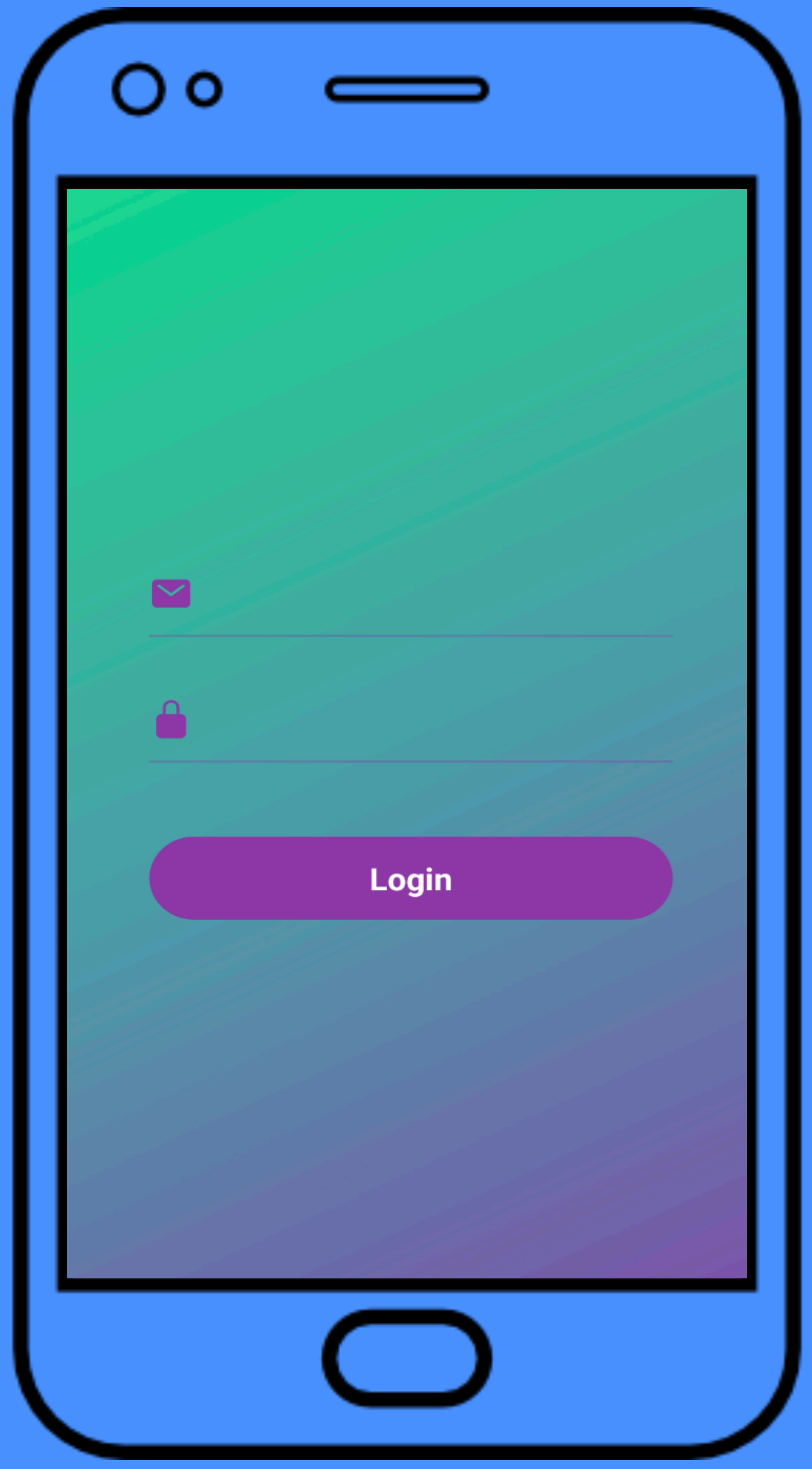
# Use AI to automate the annoying stuff



Explore the app 🤔
- manually
- or via an end-to-end test

Collect Performance Data ✅
(%CPU, FPS, RAM...)

Detect issues ✅
(low FPS, high JS thread usage)



Explore the app ✅

with AI 🤖

Collect Performance Data ✅
(%CPU, FPS, RAM...)

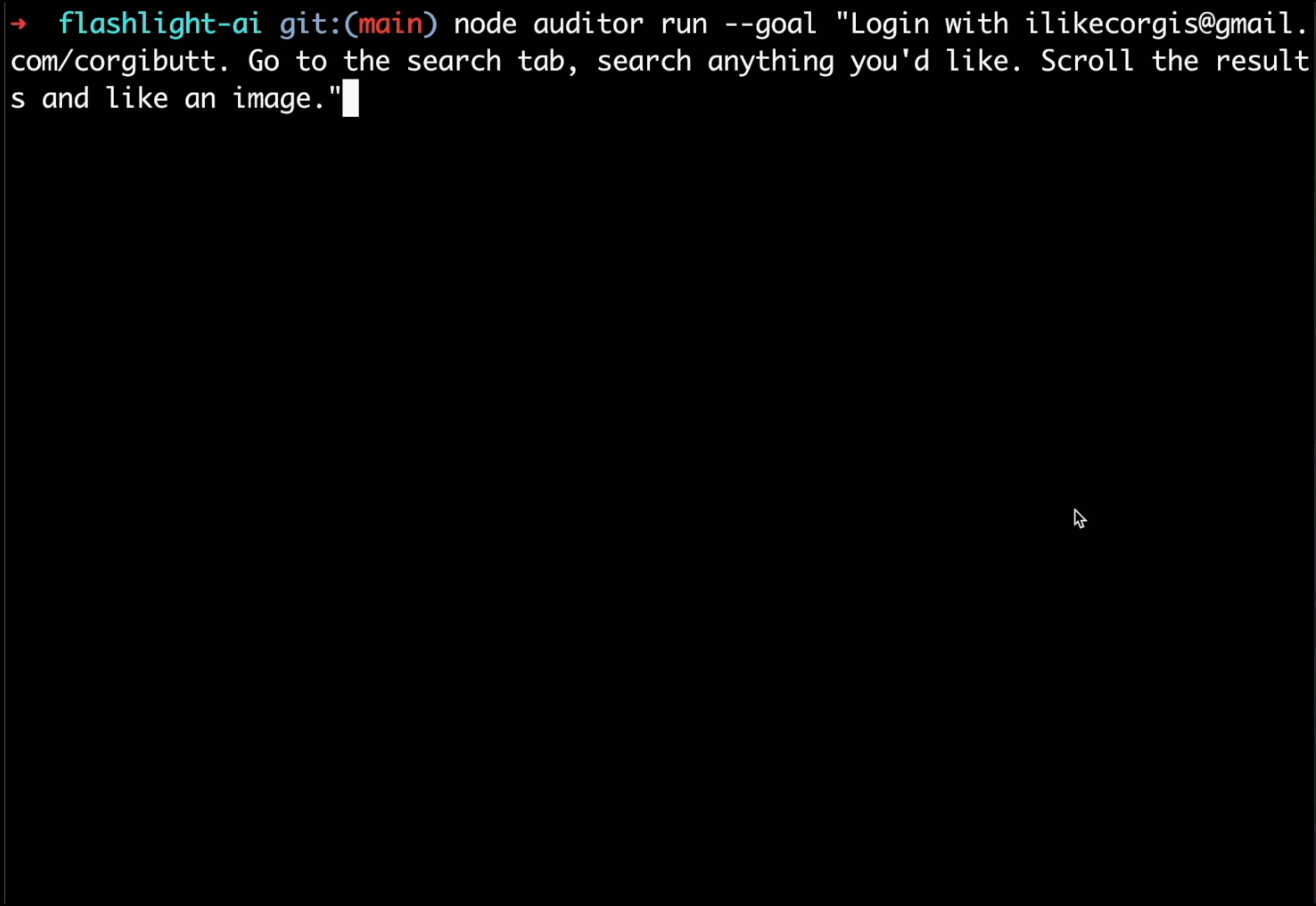
Detect issues ✅
(low FPS, high JS thread usage)

flashlight🔦

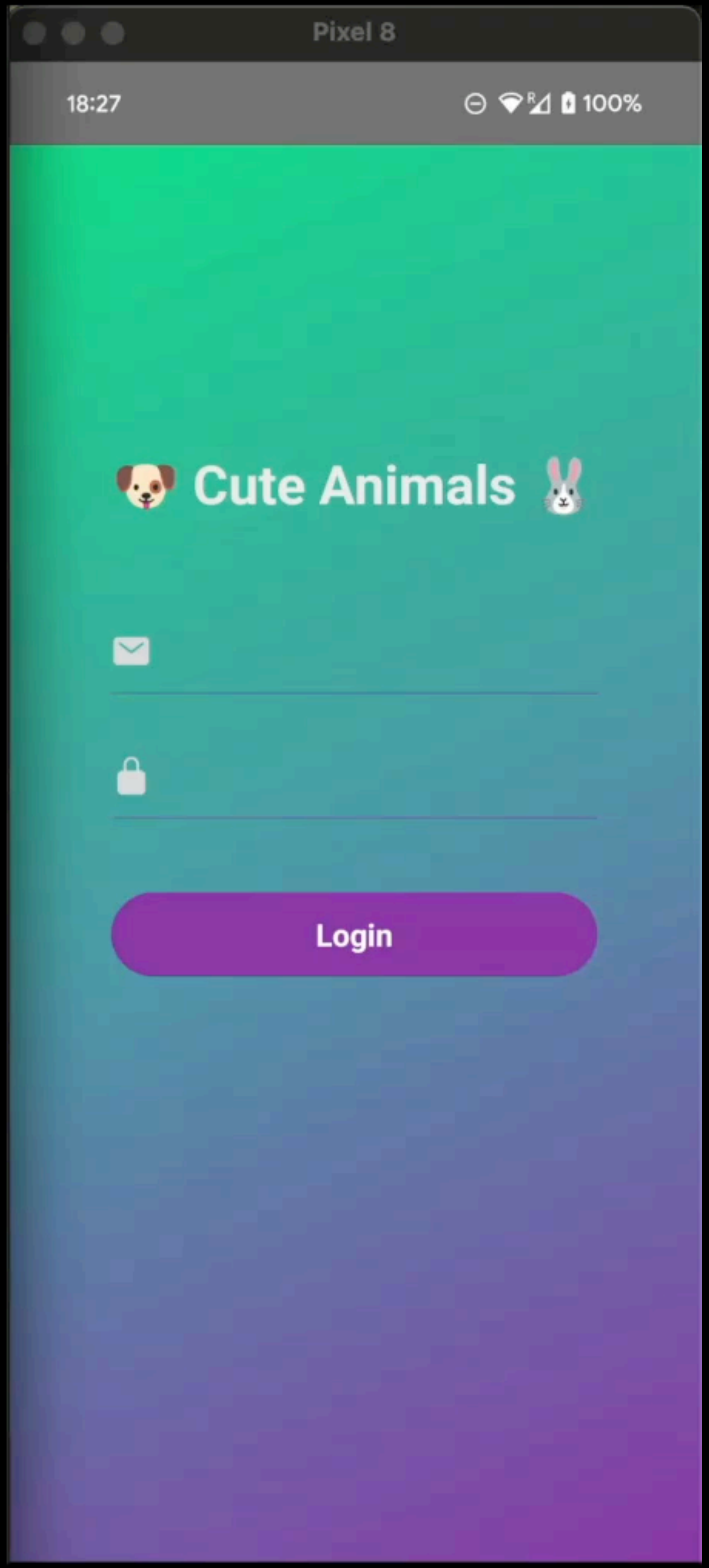
# flAshlIght 🤖

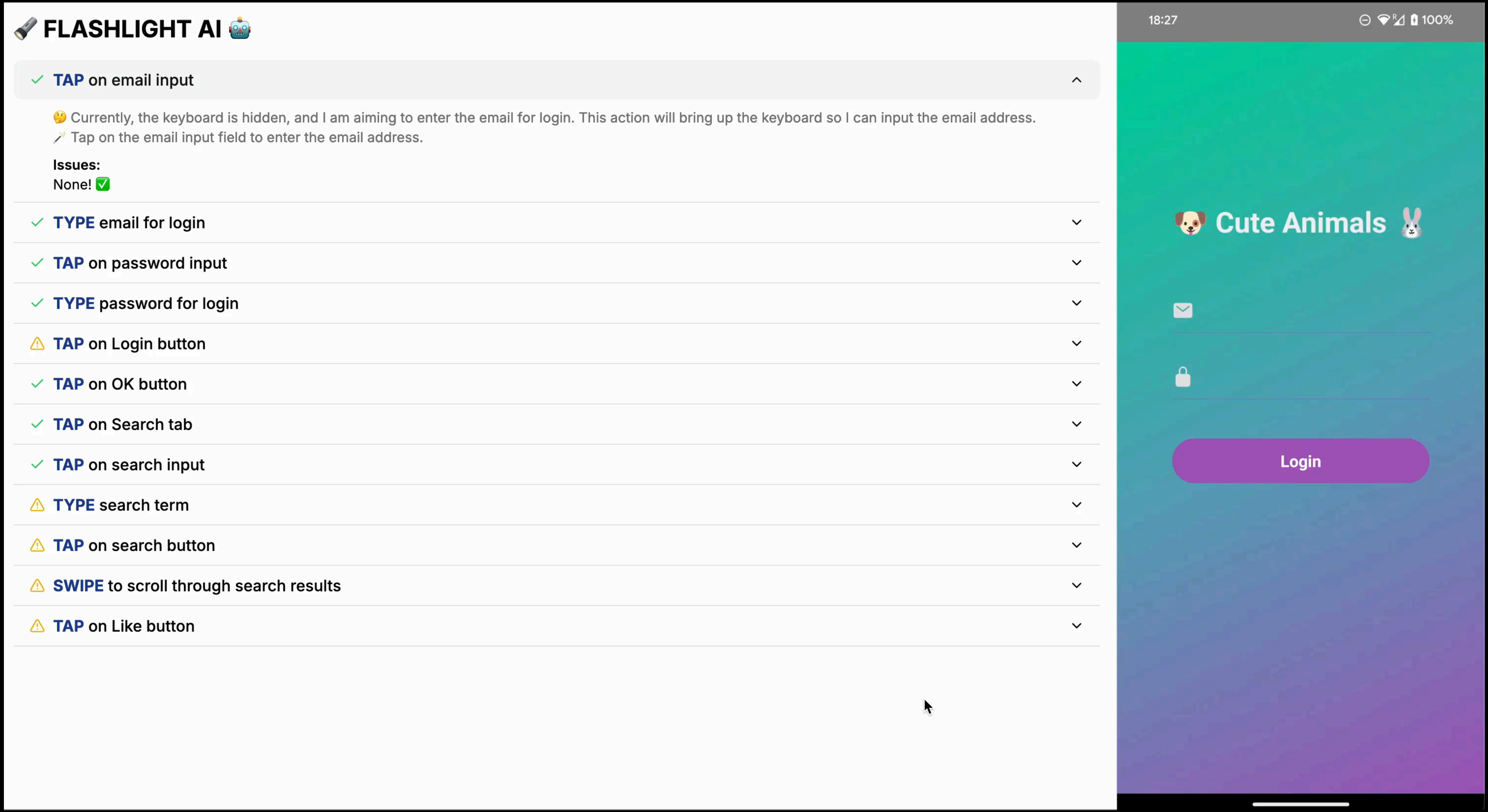
AI 🤖

# Live DEMO 🥳



```
➜  flashlight-ai git:(main) node auditor run --goal "Login with ilikecorgis@gmail.
com/corgibutt. Go to the search tab, search anything you'd like. Scroll the result
s and like an image."
```





🔦 FLASHLIGHT AI 🤖

✓ **TAP** on email input

🤔 Currently, the keyboard is hidden, and I am aiming to enter the email for login. This action will bring up the keyboard so I can input the email address.
🪄 Tap on the email input field to enter the email address.

**Issues:**
None! ✅

✓ **TYPE** email for login

✓ **TAP** on password input

✓ **TYPE** password for login

⚠ **TAP** on Login button

✓ **TAP** on OK button

✓ **TAP** on Search tab

✓ **TAP** on search input

⚠ **TYPE** search term

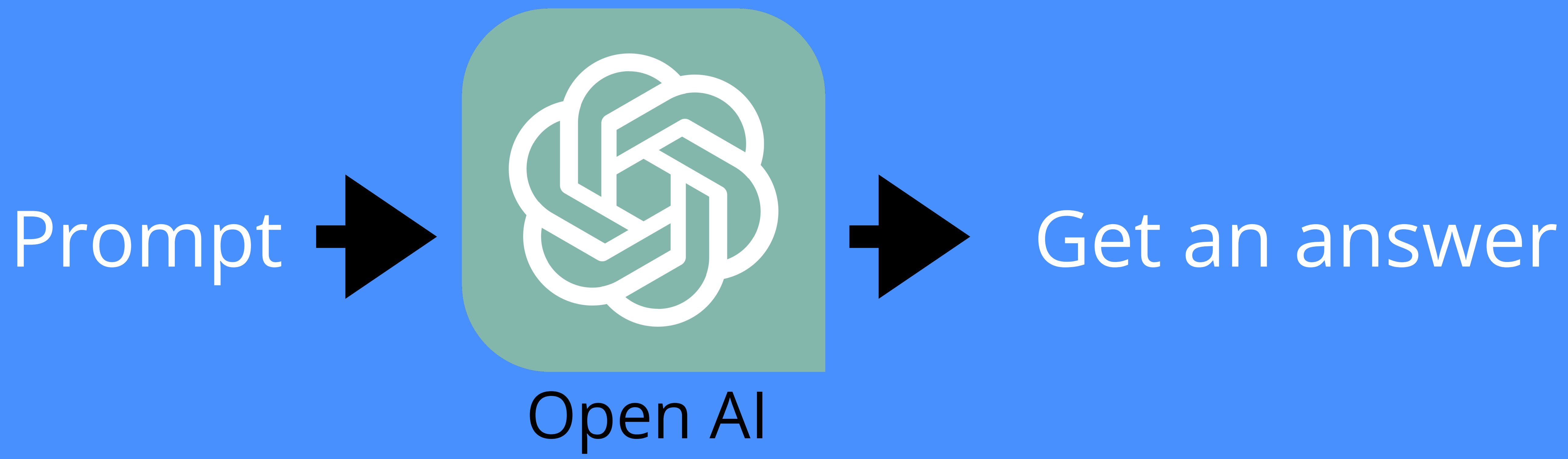
⚠ **TAP** on search button

⚠ **SWIPE** to scroll through search results

⚠ **TAP** on Like button

# How does it work?

# How does it work?

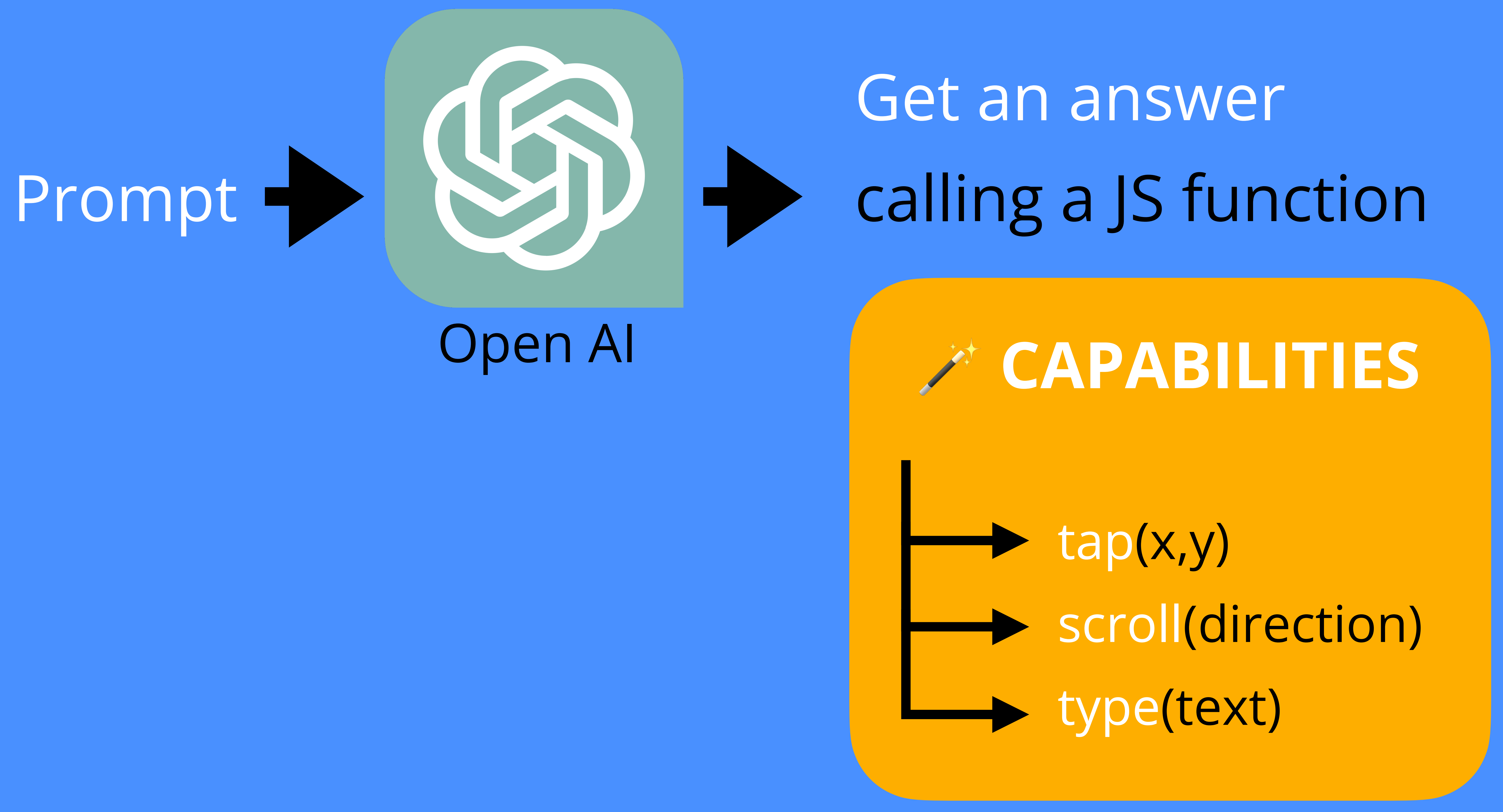


Prompt

Open AI

Get an answer

# How does it work?



Prompt

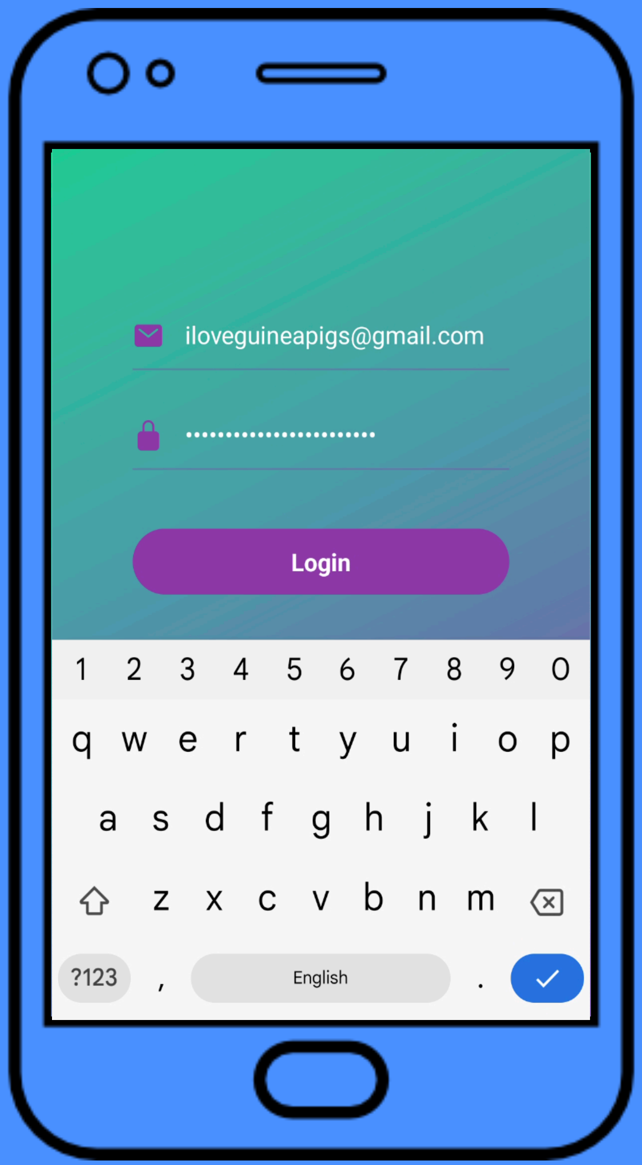
Open AI

Get an answer calling a JS function

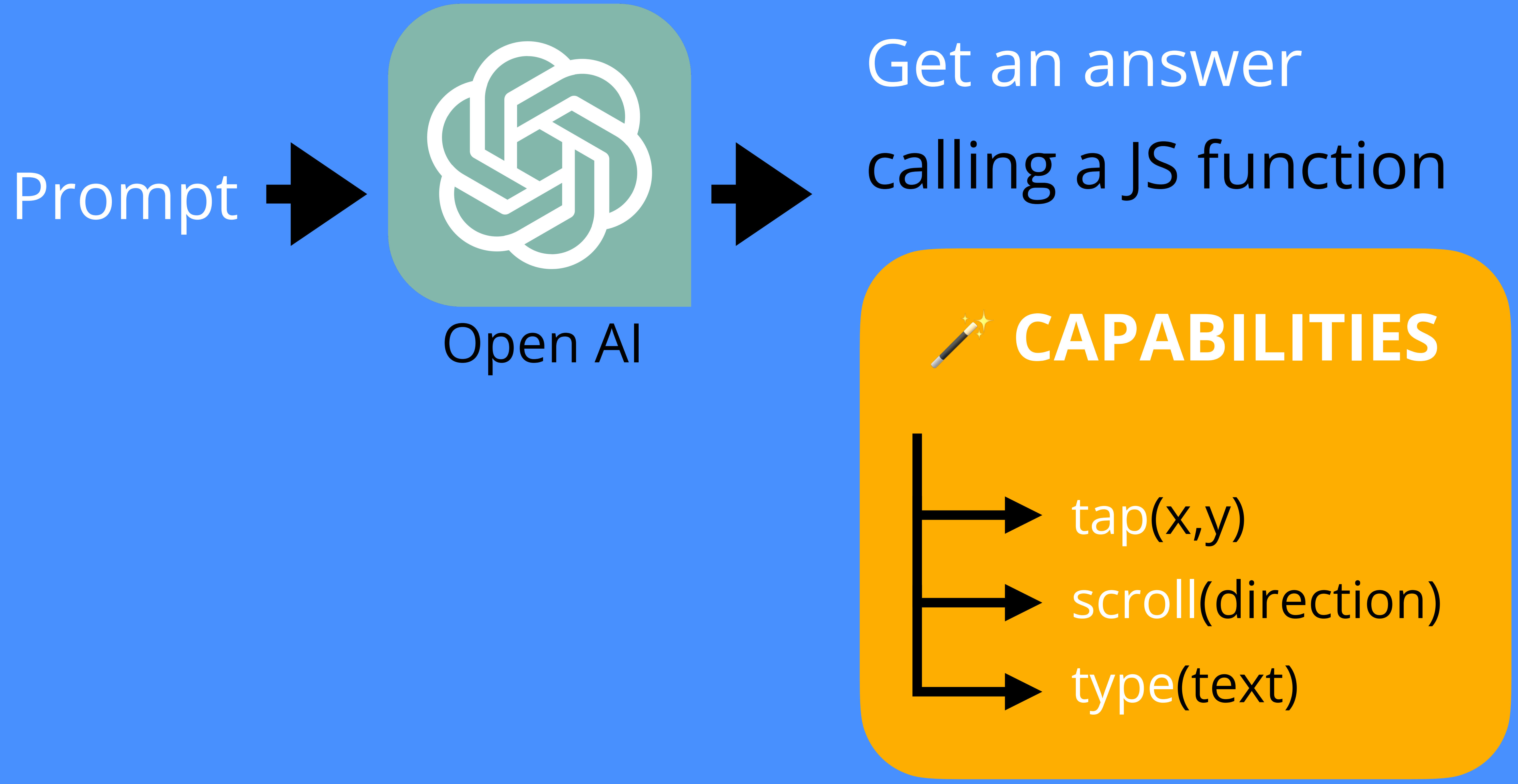
🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# How does it work?



iloveguineapigs@gmail.com



Prompt →

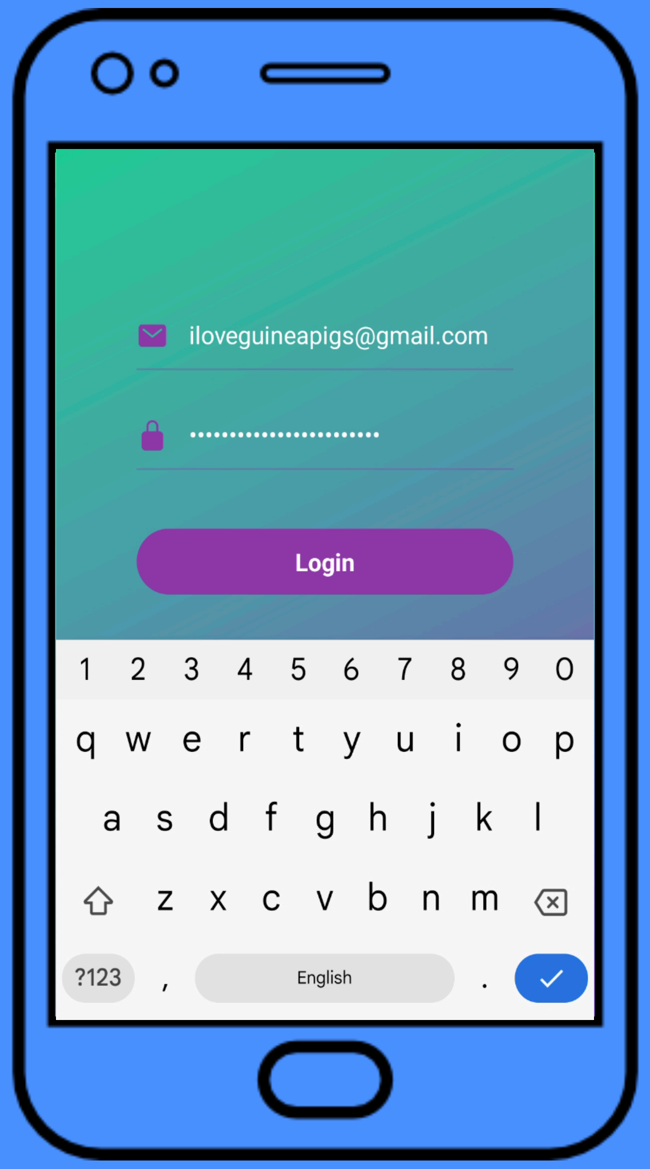
Open AI →

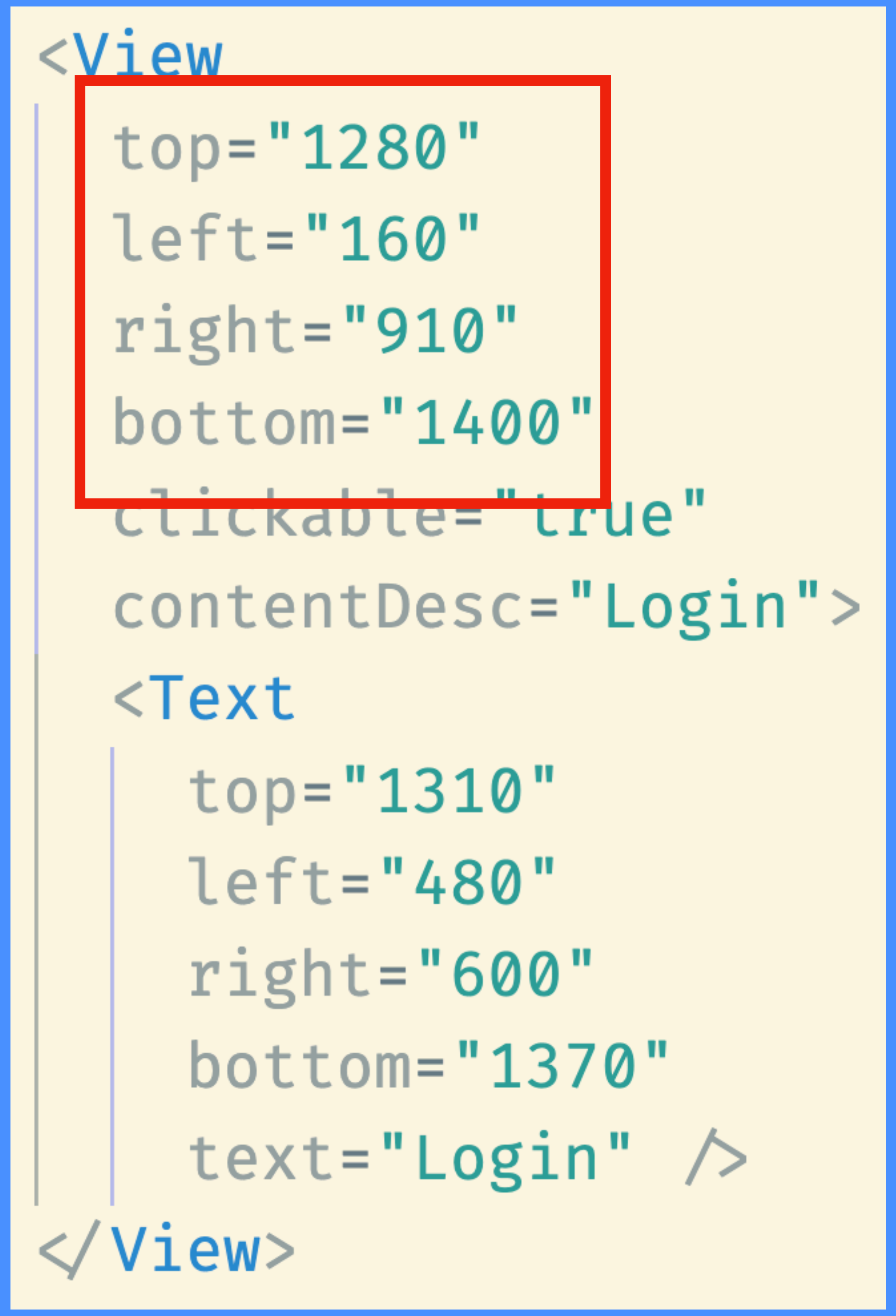
Get an answer calling a JS function

🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# How does it work?





```
<View
  top="1280"
  left="160"
  right="910"
  bottom="1400"
  clickable="true"
  contentDesc="Login">
  <Text
    top="1310"
    left="480"
    right="600"
    bottom="1370"
    text="Login" />
</View>
```

Prompt

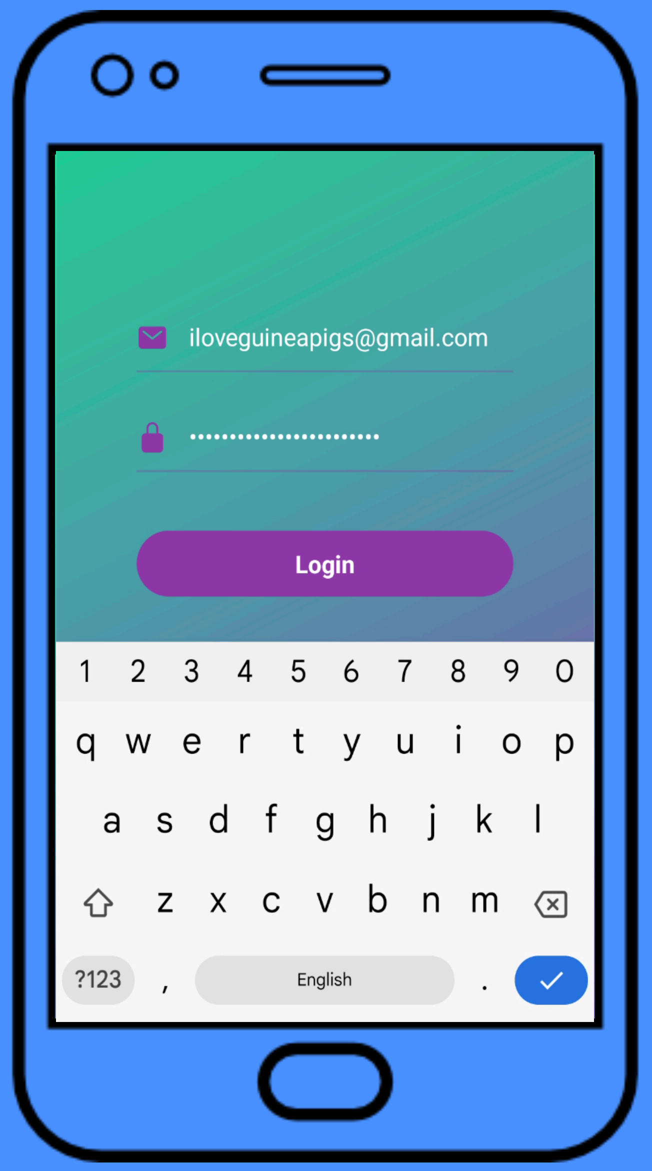
Open AI

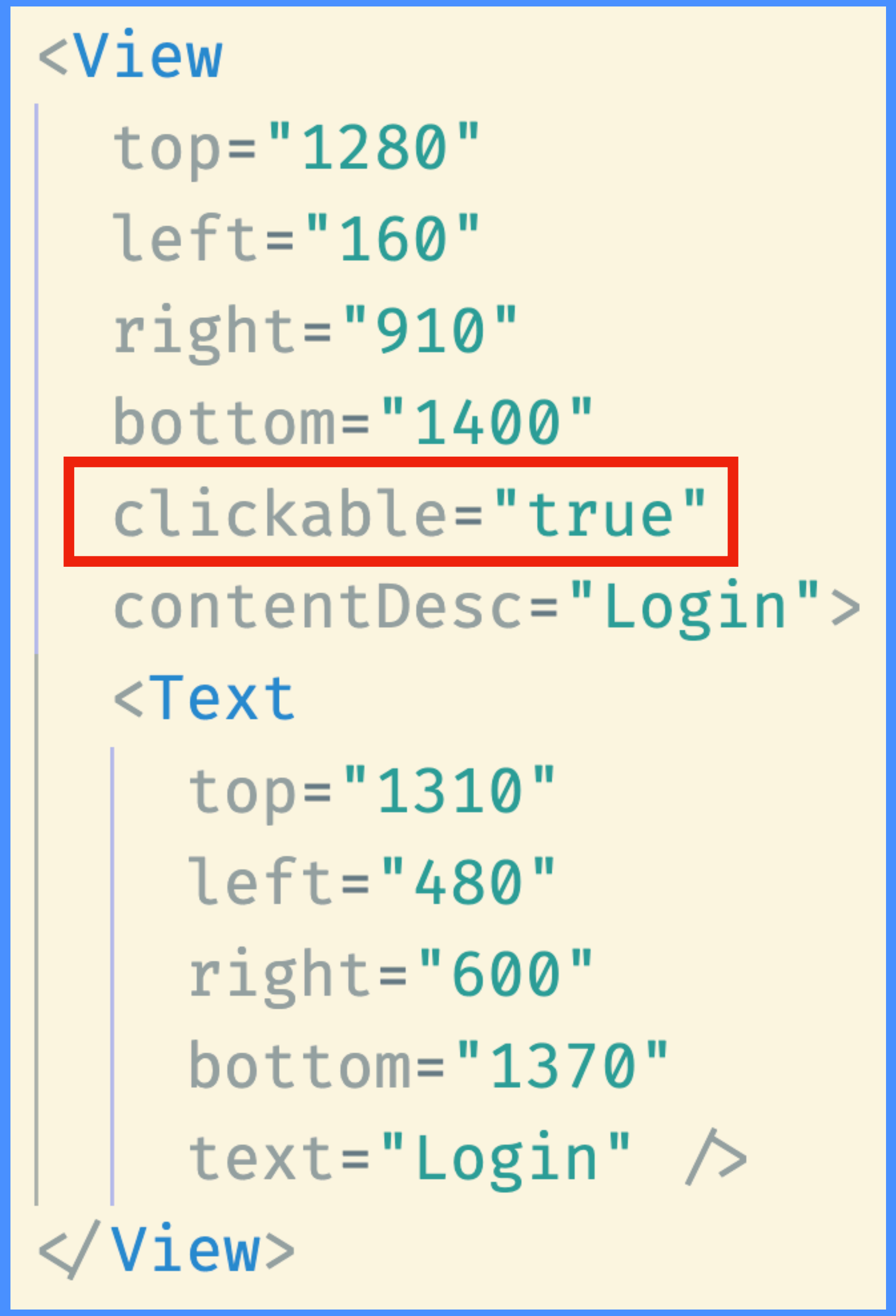
Get an answer calling a JS function

🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# How does it work?





```
<View
  top="1280"
  left="160"
  right="910"
  bottom="1400"
  clickable="true"
  contentDesc="Login">
  <Text
    top="1310"
    left="480"
    right="600"
    bottom="1370"
    text="Login" />
</View>
```

Prompt

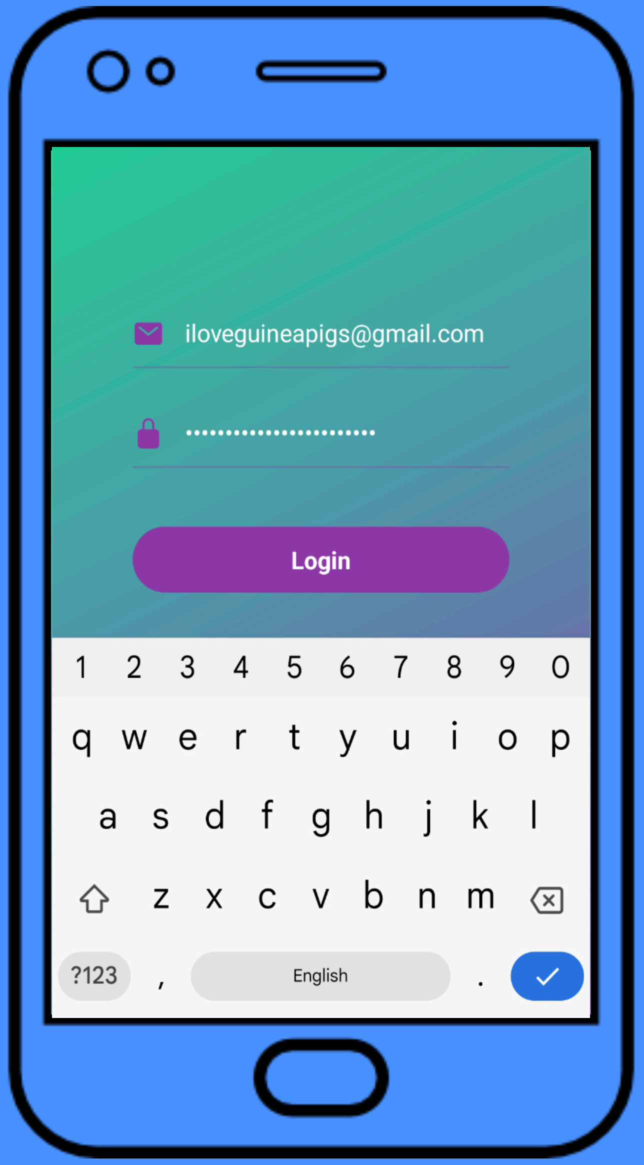
Open AI

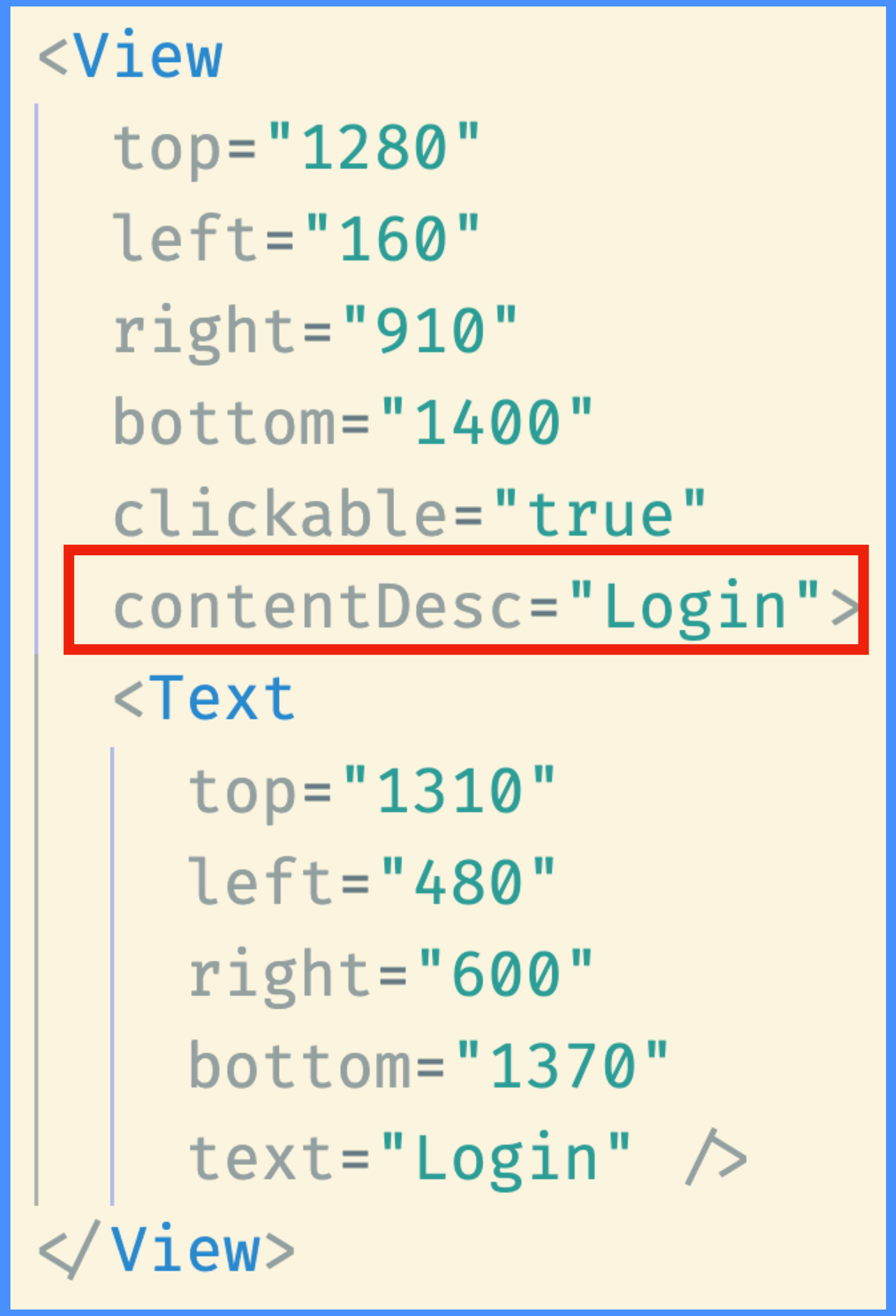
Get an answer calling a JS function

🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# How does it work?





```
<View
  top="1280"
  left="160"
  right="910"
  bottom="1400"
  clickable="true"
  contentDesc="Login">
  <Text
    top="1310"
    left="480"
    right="600"
    bottom="1370"
    text="Login" />
</View>
```

Prompt

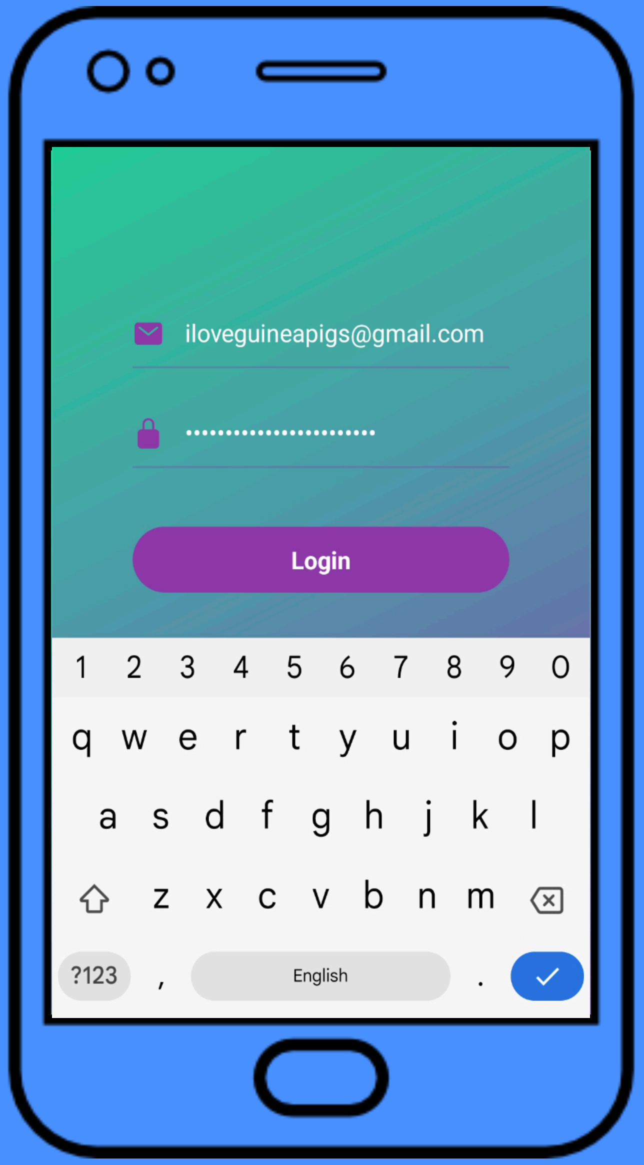
Open AI

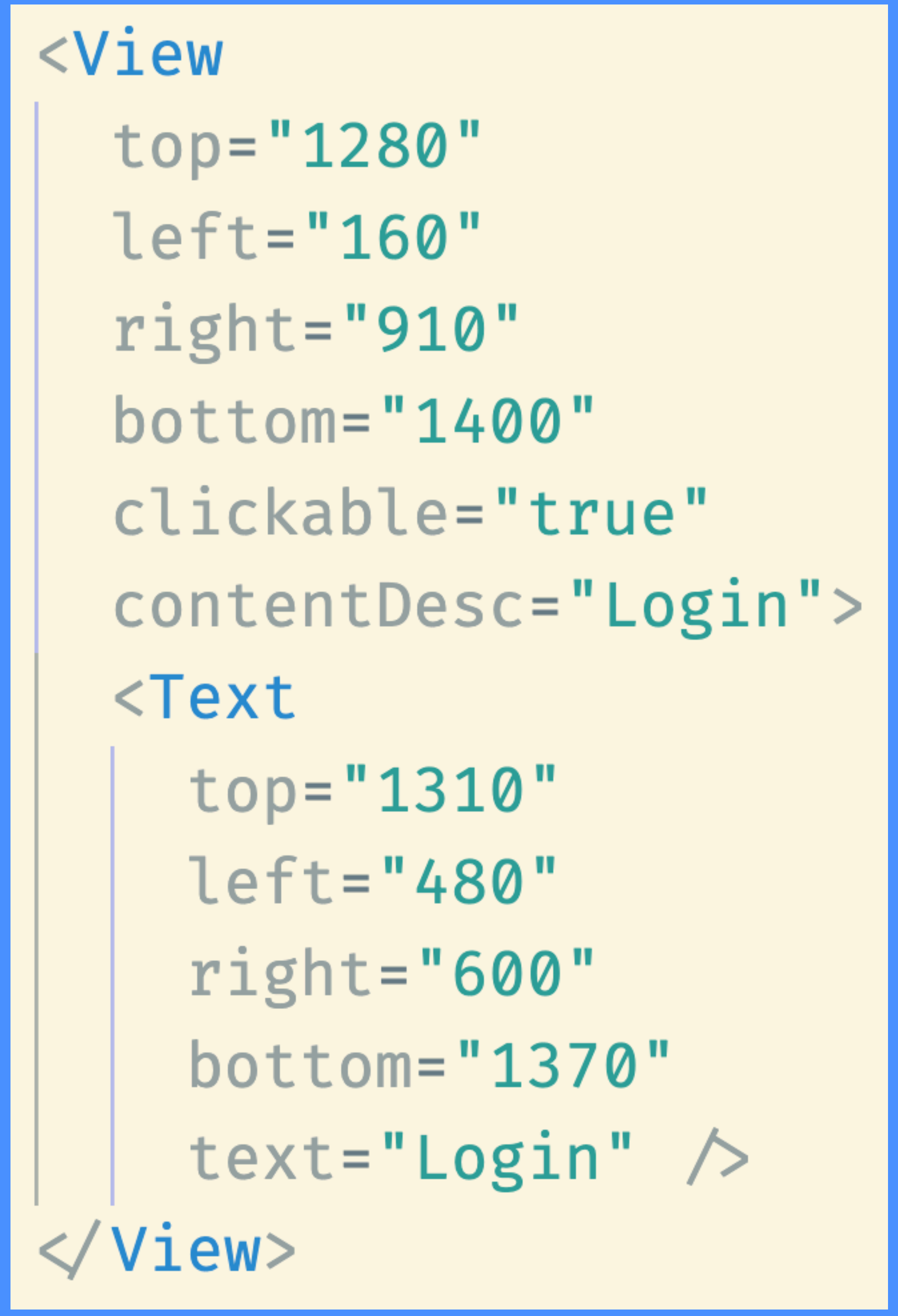
Get an answer calling a JS function

🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# How does it work?





```
<View
  top="1280"
  left="160"
  right="910"
  bottom="1400"
  clickable="true"
  contentDesc="Login">
  <Text
    top="1310"
    left="480"
    right="600"
    bottom="1370"
    text="Login" />
</View>
```

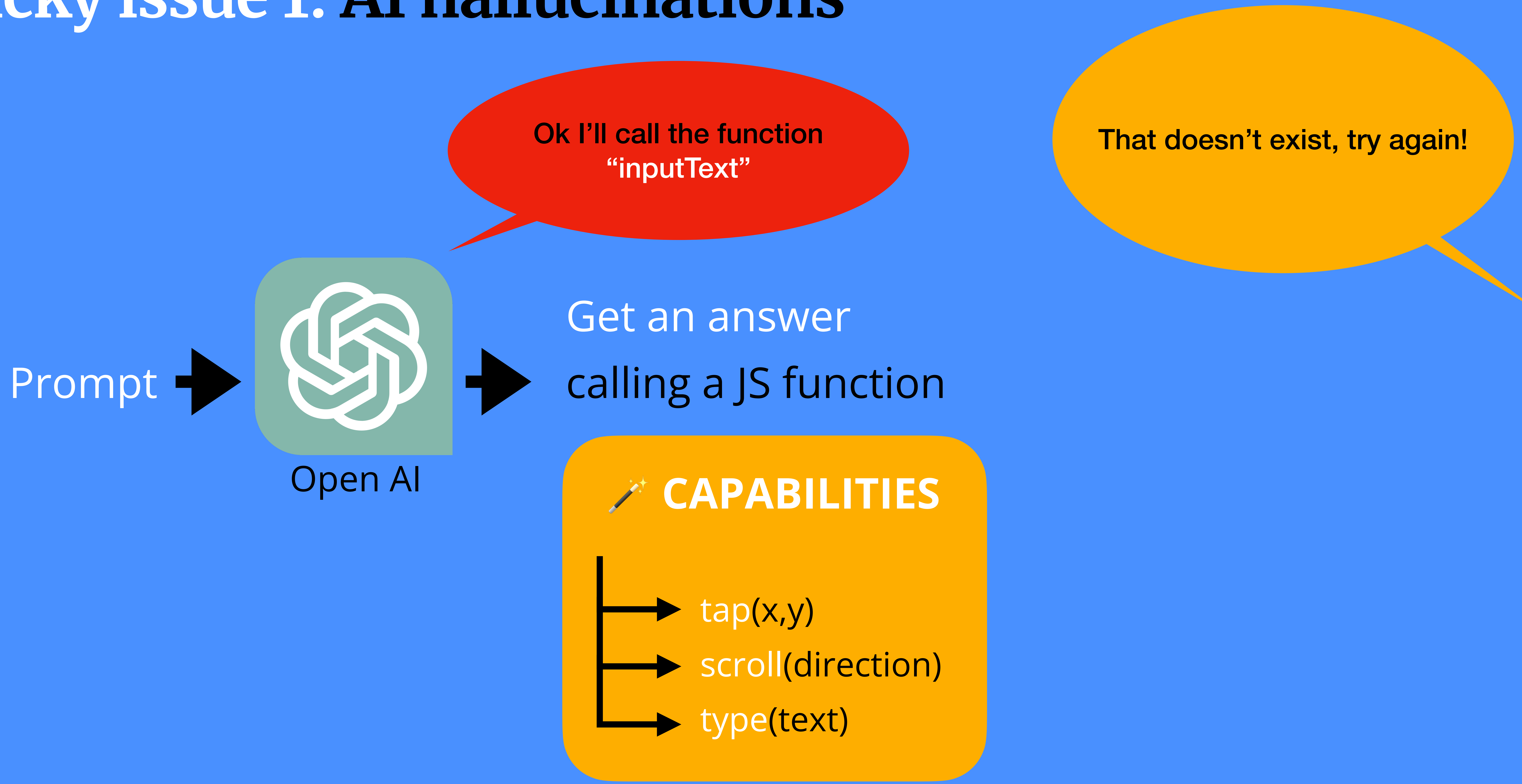
Prompt

Open AI

Get an answer calling a JS function

🪄 **tap(1340, 535)**

# Tricky issue 1: **AI hallucinations**



Ok I'll call the function "inputText"

That doesn't exist, try again!

Prompt

Open AI

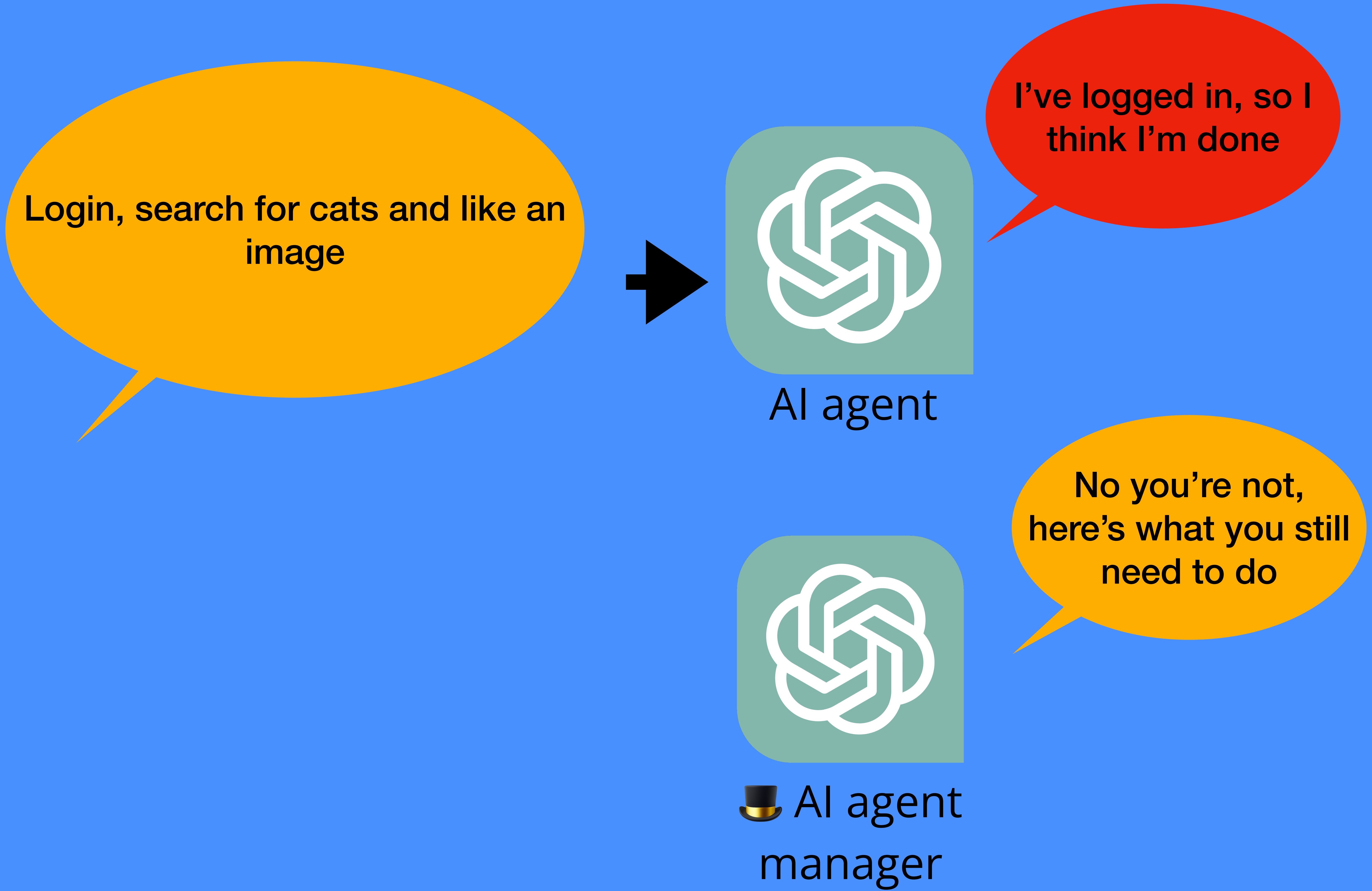
Get an answer

calling a JS function

🪄 **CAPABILITIES**

- tap(x,y)
- scroll(direction)
- type(text)

# Tricky issue 2: AI stopping prematurely



Login, search for cats and like an image

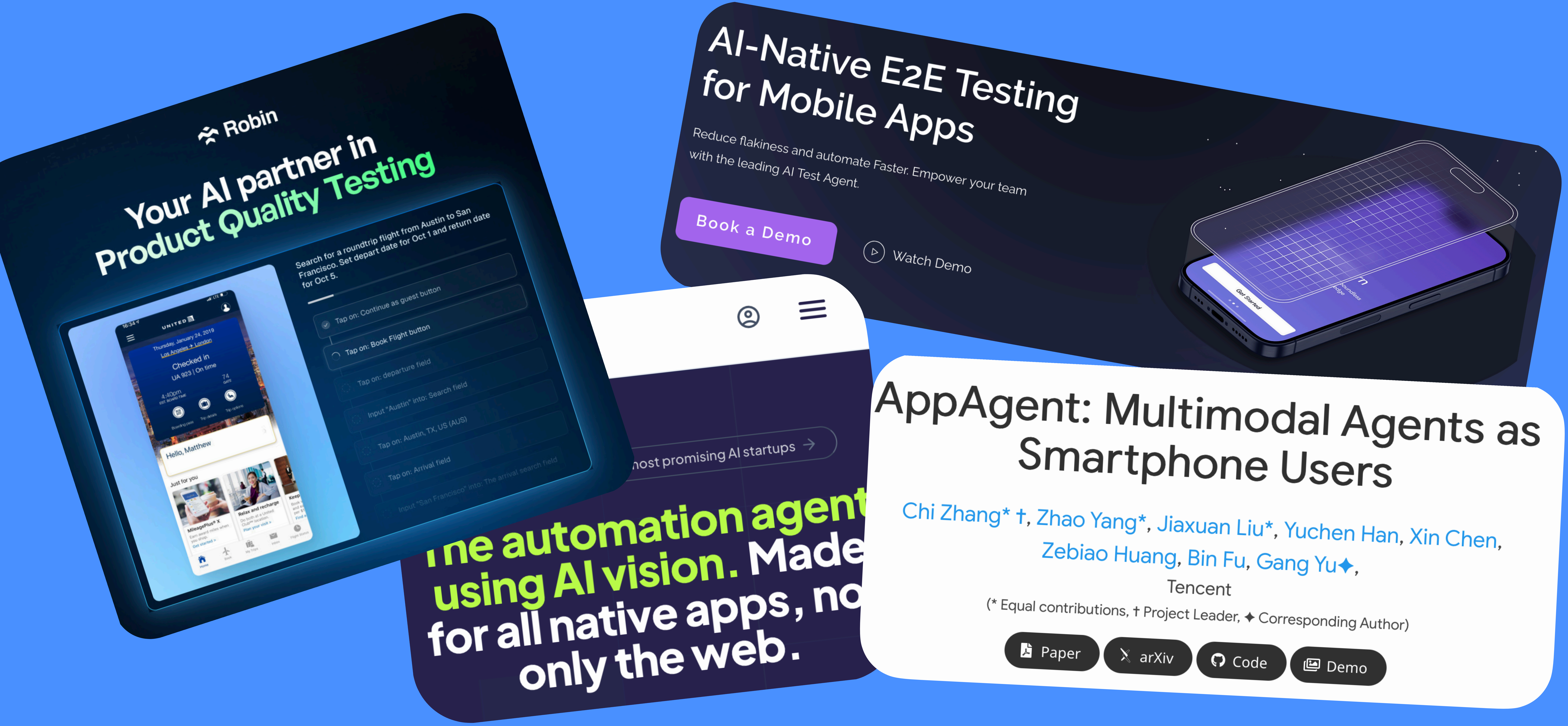
I’ve logged in, so I think I’m done

AI agent

No you’re not, here’s what you still need to do

🎩 AI agent manager

# Many people have realized AI in E2E tests is big



Robin

Your AI partner in Product Quality Testing

Search for a roundtrip flight from Austin to San Francisco. Set depart date for Oct 1 and return date for Oct 5.

- Tap on: Continue as guest button
- Tap on: Book Flight button
- Tap on: departure field
- Input "Austin" into: Search field
- Tap on: Austin, TX, US (AUS)
- Tap on: Arrival field
- Input "San Francisco" into: The arrival search field

## AI-Native E2E Testing for Mobile Apps

Reduce flakiness and automate Faster. Empower your team with the leading AI Test Agent.

he automation agent using AI vision. Made for all native apps, no only the web.

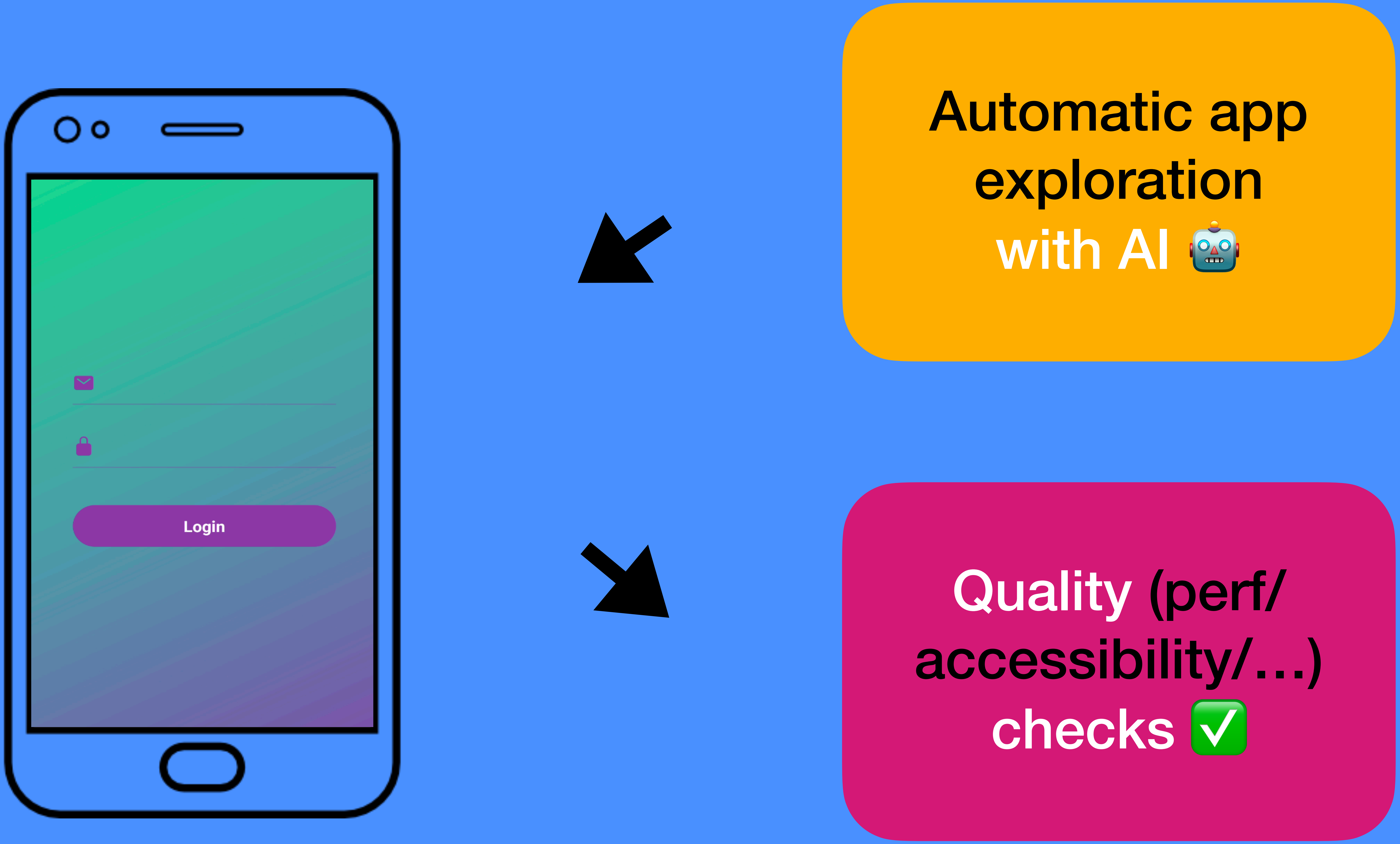
## AppAgent: Multimodal Agents as Smartphone Users

Chi Zhang* †, Zhao Yang*, Jiaxuan Liu*, Yuchen Han, Xin Chen, Zebiao Huang, Bin Fu, Gang Yu✦,

Tencent

(* Equal contributions, † Project Leader, ✦ Corresponding Author)

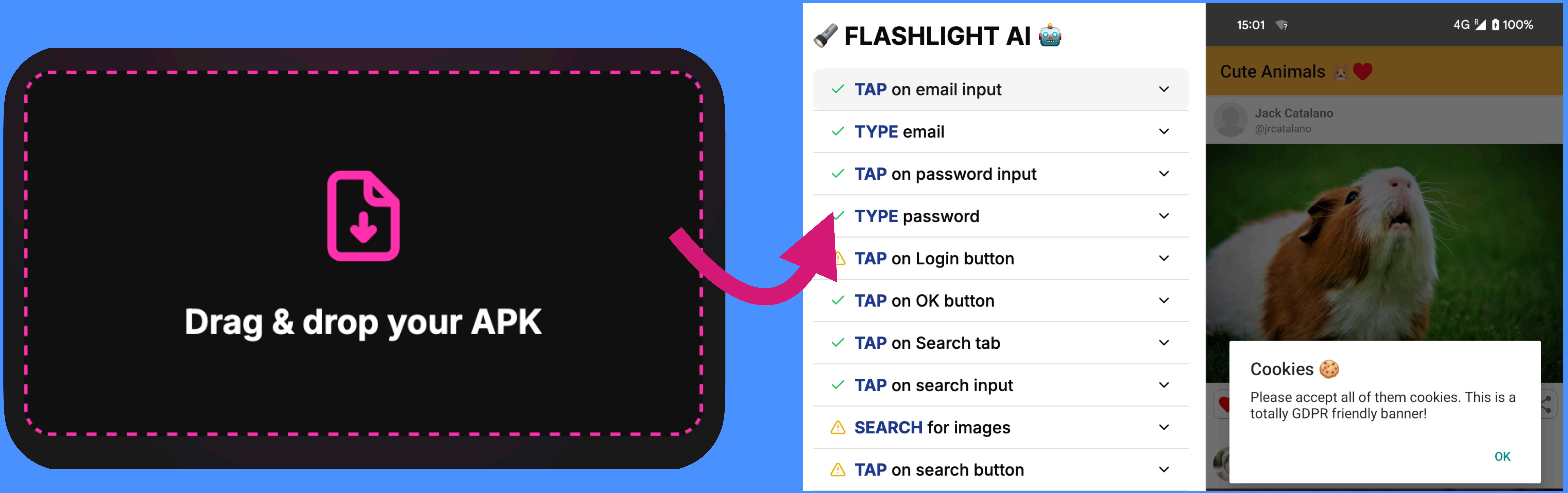
# The whole idea: AI enhanced quality audit



Automatic app exploration with AI 🤖

Quality (perf/accessibility/…) checks ✅

# The End Goal 🤞: 1-click performance audit



Drag & drop your APK

🔦 FLASHLIGHT AI 🤖

- ✓ TAP on email input
- ✓ TYPE email
- ✓ TAP on password input
- ✓ TYPE password
- ⚠ TAP on Login button
- ✓ TAP on OK button
- ✓ TAP on Search tab
- ✓ TAP on search input
- ⚠ SEARCH for images
- ⚠ TAP on search button

Cookies 🍪

Please accept all of them cookies. This is a totally GDPR friendly banner!

OK

# Let’s recap

🔥 E2E should include perf/accessibility/security testing/…

🤖 AI in E2E tests is **BIG**

🤞AI can automate annoying tasks

🔗 Link to the repo
🙏 Leave feedback
📚 Slides/Resources