Lessons Learned from AI-Powered Visual Reasoning Feedback

Detecting UI issues, accessibility problems, and design inconsistencies with LLMs

Risko Ruus

June 5, 2026

About Me

Risko Ruus
Principal QA Engineer
@Rush Street Interactive

EXPERIENCE

20 years in QA — from early-stage startups to products at scale

HOBBIES

- Backcountry hiking
- Learning Japanese with my son
- Tinkering with AI

Agenda

Traditional UI Assertions

Agenda

Traditional UI Assertions

Visual AI Reasoning: Core Concepts & Basic Usage

Agenda

Traditional UI Assertions

Visual AI Reasoning: Core Concepts & Basic Usage

Advanced Analysis Techniques

Agenda

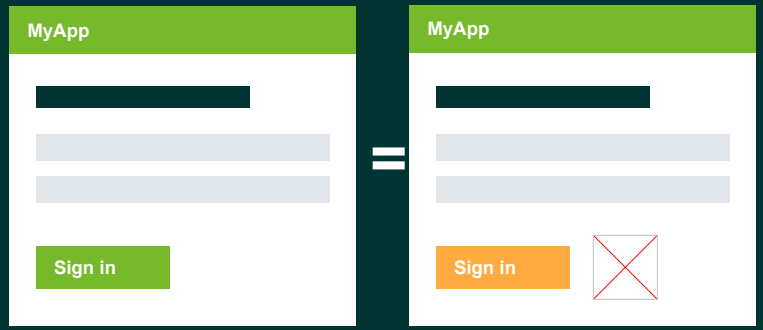
Traditional UI Assertions

Visual AI Reasoning: Core Concepts & Basic Usage

Advanced Analysis Techniques

Integrating Visual Reasoning Checks into Workflows

Traditional UI Assertions – Full Match

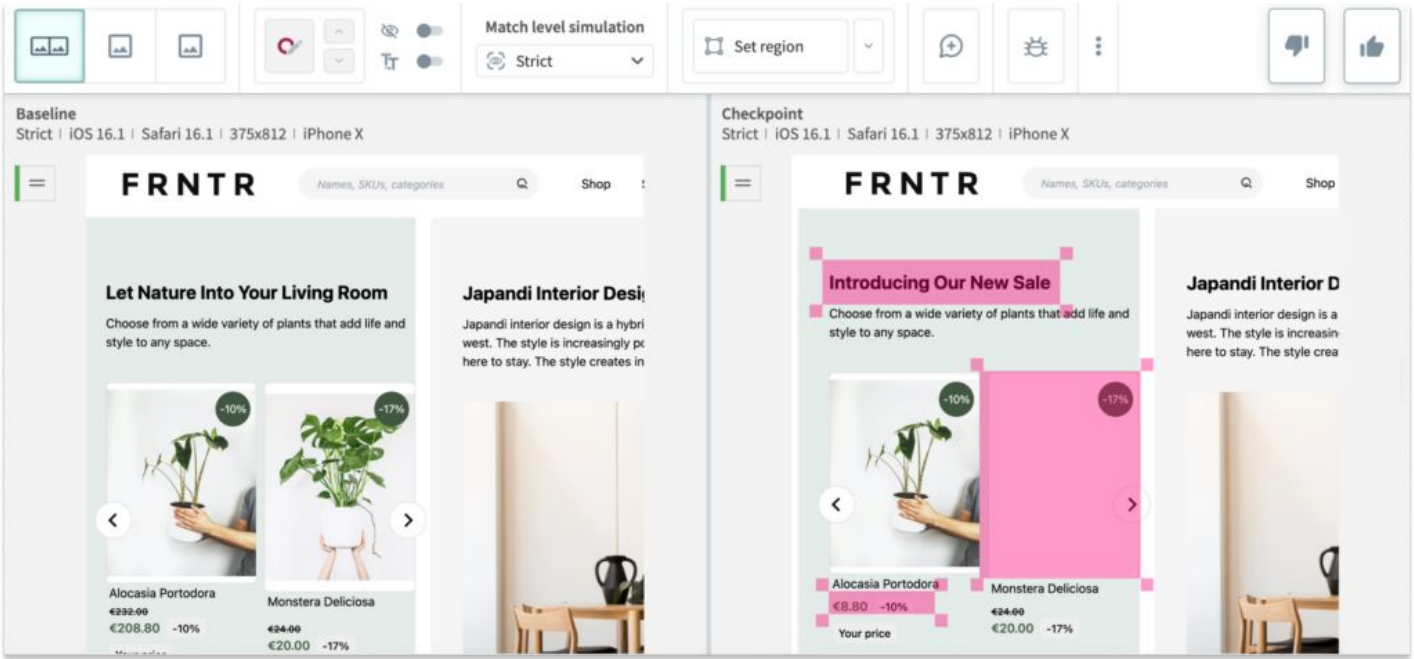


Problems:

- ⚠️ Too flaky — fails on minor changes
- ⚠️ Font rendering, anti-aliasing cause false failures
- ⚠️ Requires constant screenshot updates

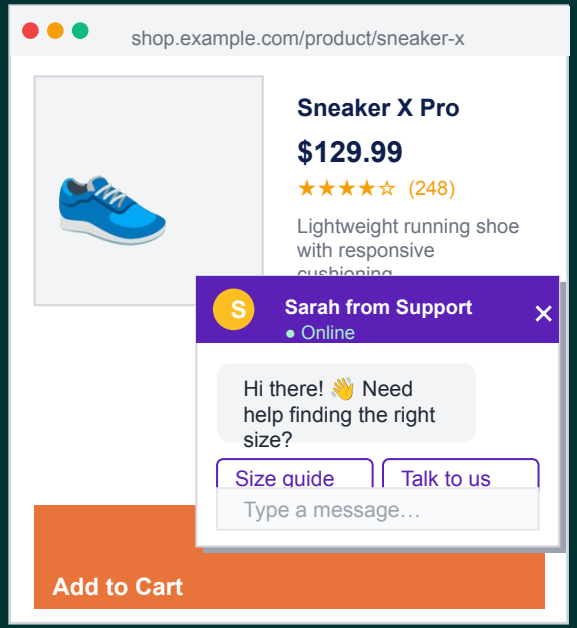
High Maintenance **Brittle**

Example Diff Output

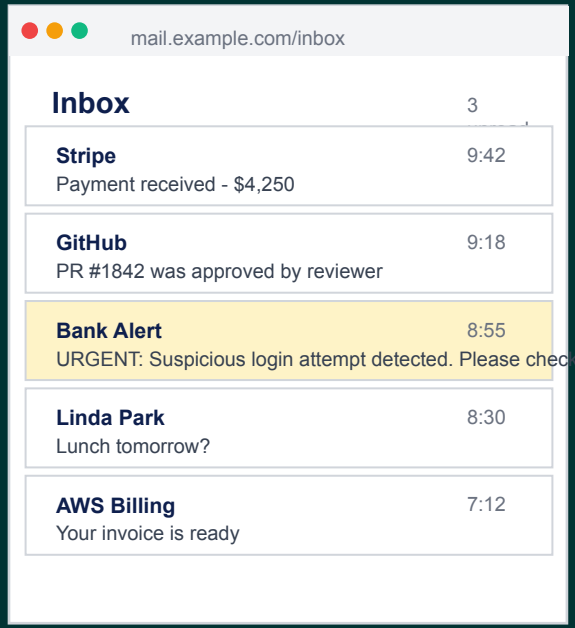


Real-World Examples: When isDisplayed/isVisible is Not Enough

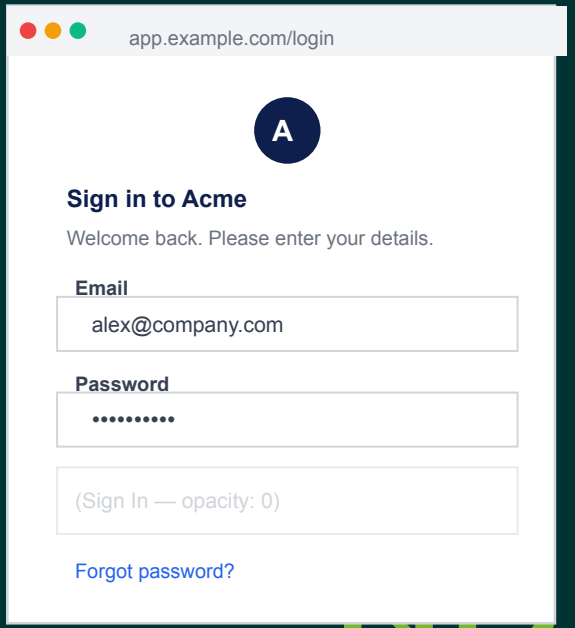
Each page looks fine to a non-vision UI test — but a real user has a poor experience



Covered by overlay



Clipped by overflow

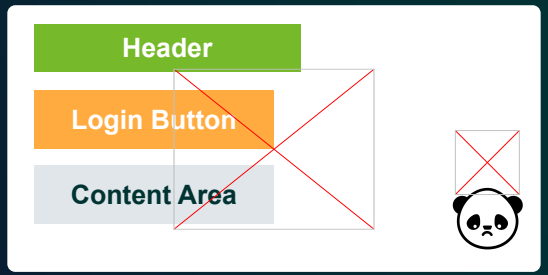


Opacity: 0

Traditional UI Assertions – Specific Element Check

Aka - The Giant Panda Problem

Goal - Verify that a login button is displayed



→ PASS

→ MISSED!

Problems:

⚠️ Explicit checks miss new erroneous content
⚠️ Only validates what you coded it to check

Blind Spots

False Confidence

Visual AI Reasoning: Core Concepts & Basic Usage

Three steps from screen image to readable feedback on bugs and design

1. Screenshot + prompt in

Just a screenshot + text

Show the AI a picture with a task of finding issues. No design spec or "before" image needed, but it can be helpful.

2. Visual + semantic check

“Looks” and reasons

The LLM breaks the image and prompt text into tokens. Then it spends any internal thinking tokens and finally generates a response.

3. Clear feedback out

A list of issues

Returns issues with severity and a short explanation. References shapes or regions by name, not pixel coordinates.

Key insight: *General-purpose LLMs catch many visual and design bugs out of the box*

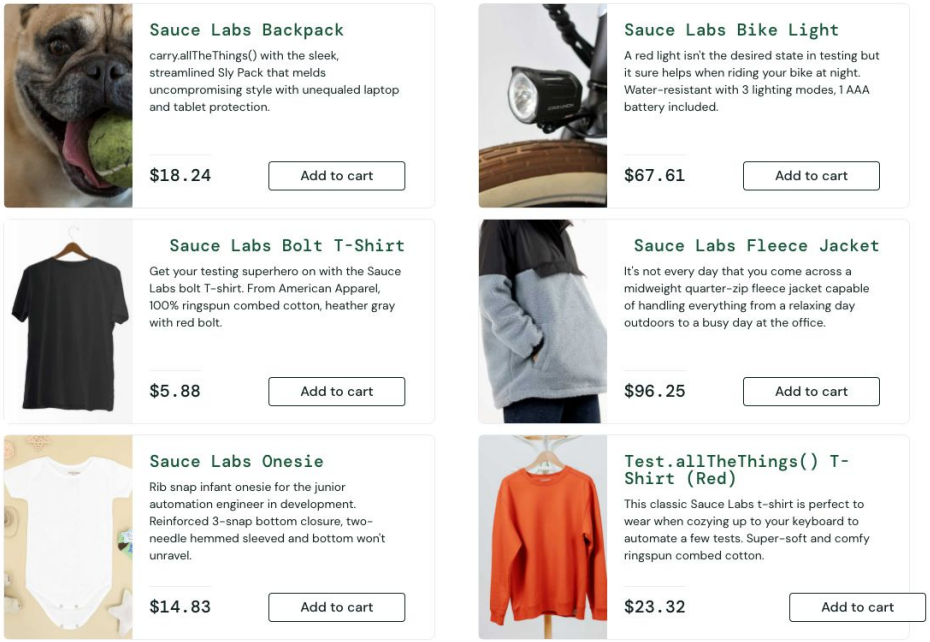
Example Vision LLM Input Image

This public website was developed by Sauce Labs for demo/testing purposes:

https://www.saucedemo.com/inventory.html

It includes deliberate bugs that we expect the AI to identify.

NB! Always check which data you are allowed to send to LLM providers e.g. enterprise / regulated / private data usage



Example Vision LLM Prompt

Analyze this image and identify all UI, content, and functional issues.

Severity levels:

- **Critical** – Blocks the main flow/actions, damages trust
- **Major** – Clear defect, workaround exists
- **Minor** – Cosmetic, unlikely to affect users

Output valid JSON using this schema →

SYSTEM PROMPT

+

```
{
  "issues": [
    {
      "id": 1,
      "severity": "critical|major|minor",
      "category": "Content, UI/Layout...",
      "element": "Specific element",
      "description": "What is wrong",
      "expected": "Correct state",
      "impact": "User/biz impact"
    }
  ],
  "summary": {
    "critical": 0,
    "major": 0,
    "minor": 0,
    "total": 0,
    "overall_assessment": "Summary..."
  }
}
```

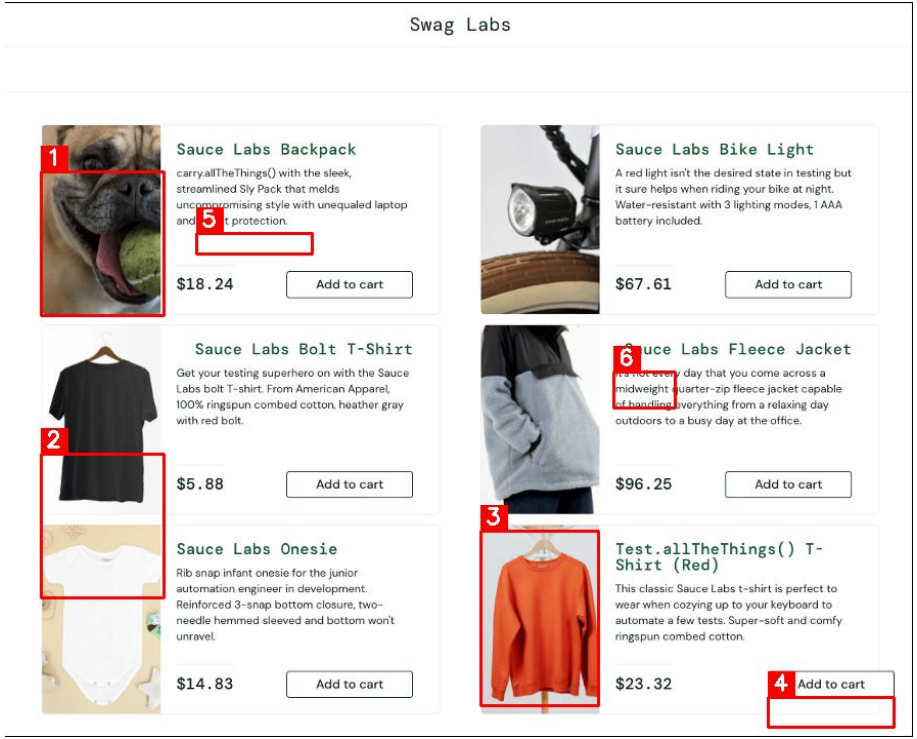
JSON SCHEMA

Example Vision LLM Prompt Output

JSON OUTPUT


```
{
  "id": 1,
  "severity": "critical",
  "category": "Content/Copy",
  "element": "Product name: 'Test.allTheThings() T-Shirt (Red)'",
  "description": "Product name contains a literal test method
    call 'Test.allTheThings()' — unmistakable test/developer
    placeholder data visible to end users.",
  "expected": "A real, customer-facing product name
    (e.g., 'Classic Red T-Shirt')",
  "impact": "Confirms test data is in production;
    severely damages credibility and trust.
    Indicates staging or QA environment exposed."
}
```

Agentic Vision with Annotations Highlighting Detected Issues



ID	Severity	Issue Summary
1	Critical	Wrong image on Backpack (shows dog photo)
2	Major	Bolt T-Shirt missing red bolt graphic
3	Major	T-Shirt shows sweatshirt image
4	Minor	Inconsistent Add to Cart button width
5	Minor	Developer placeholder text in description
6	Major	Suspicious pricing suggests staging data

From Pixel Diffs to LLM Visual Intelligence Comparison

	Traditional Pixel Diff	AI-Powered Visual Reasoning
Detects	Pixel-level differences	**Semantic UI issues**
Reports	Changed vs unchanged pixels	Layout, hierarchy, meaning, accessibility
False Positives	High	Medium (contextual reasoning)
Feedback	"Pixels differ at (x,y)"	"Button is hidden by overlay"
Exclusions	Manual exclusion areas per image	Ignore prompts
Speed	Milliseconds (local compute)	Seconds per image
Cost	Near zero (local CPU)	API token fees per invocation
Determinism	Deterministic	**Non-deterministic!**

AI visual checks should complement, not replace, deterministic checks!

Advanced Analysis Techniques – Model Selection Overview

OpenAI	Anthropic	Google	Open Weights
• GPT-5.5 • GPT-5.4 Pro • GPT-5.4 • GPT-5.4 Mini • GPT-5.4 Nano	• Opus 4.7 • Sonnet 4.6 • Haiku 4.5	• Gemini 3.5 Flash • Gemini 3.1 Pro • Gemini 3.1 Flash-Lite • Gemini 3 Flash	• Qwen 3.5 397B A17B • Qwen 3.5 27B • Kimi K2.6 …

Thinking effort is configurable on most models

A **"reasoning budget"** setting (minimal / low / medium / high, or a token cap) controls how many internal reasoning tokens the model spends before answering.

Higher effort = better accuracy on hard tasks at more latency & cost; lower effort = faster, cheaper responses.

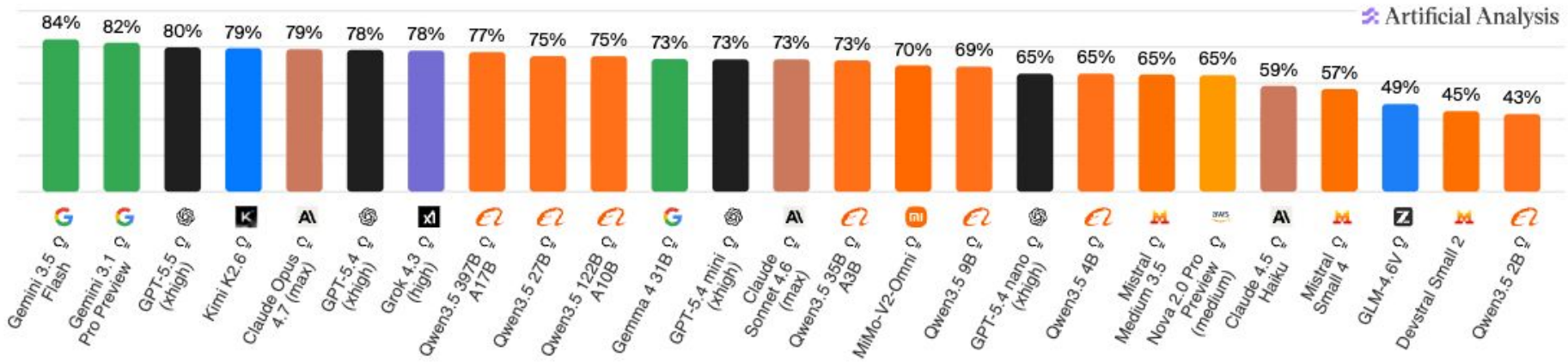
**as of: May 20, 2026*

Visual Reasoning Intelligence (MMMU Pro evaluation)

25 of 190 models

0 of 417 model & provider com...

Visual reasoning intelligence: MMMU Pro evaluation



Reasoning models are indicated by a lightbulb icon.

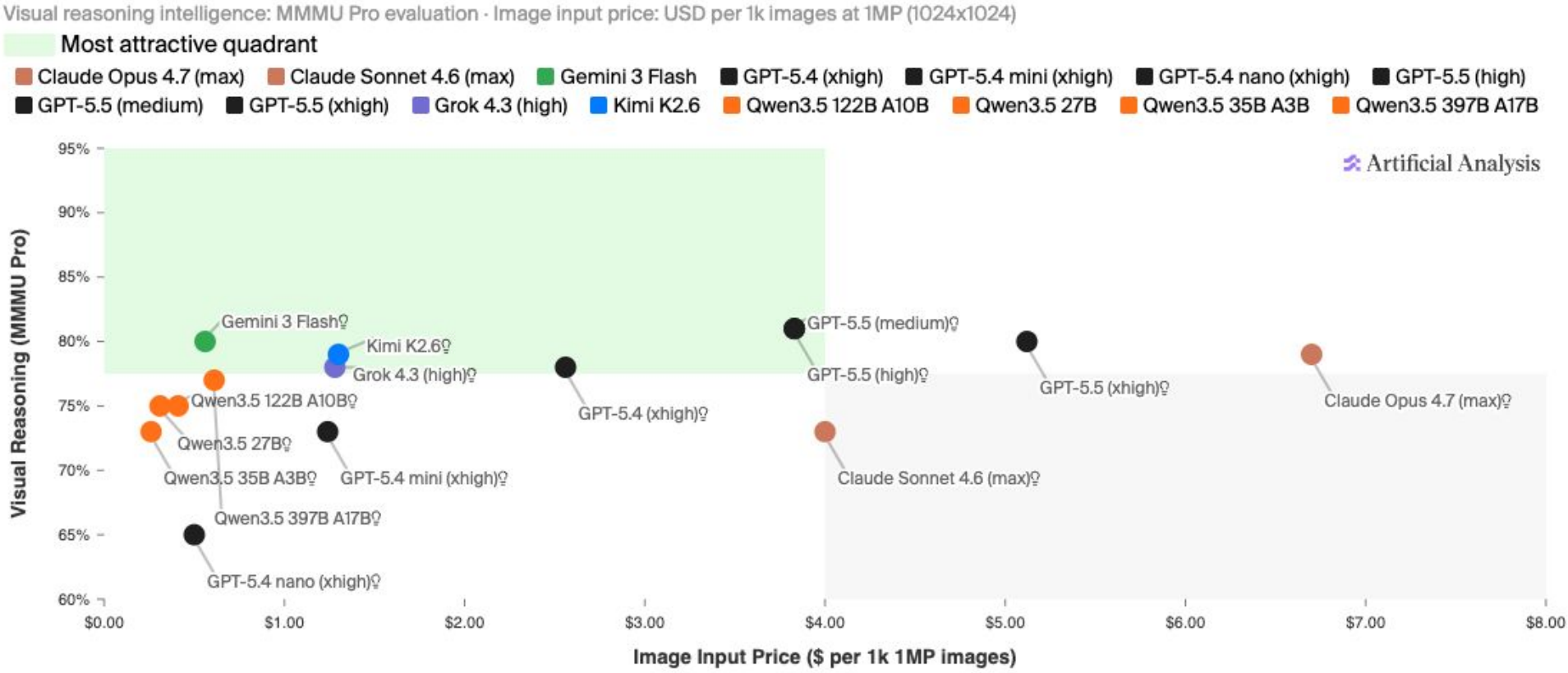
MMMU Pro

Multimodal reasoning quality evaluation based on 1.7k questions which require interpreting and reasoning over images.

**as of: May 20, 2026*

https://artificialanalysis.ai/models/multimodal/vision

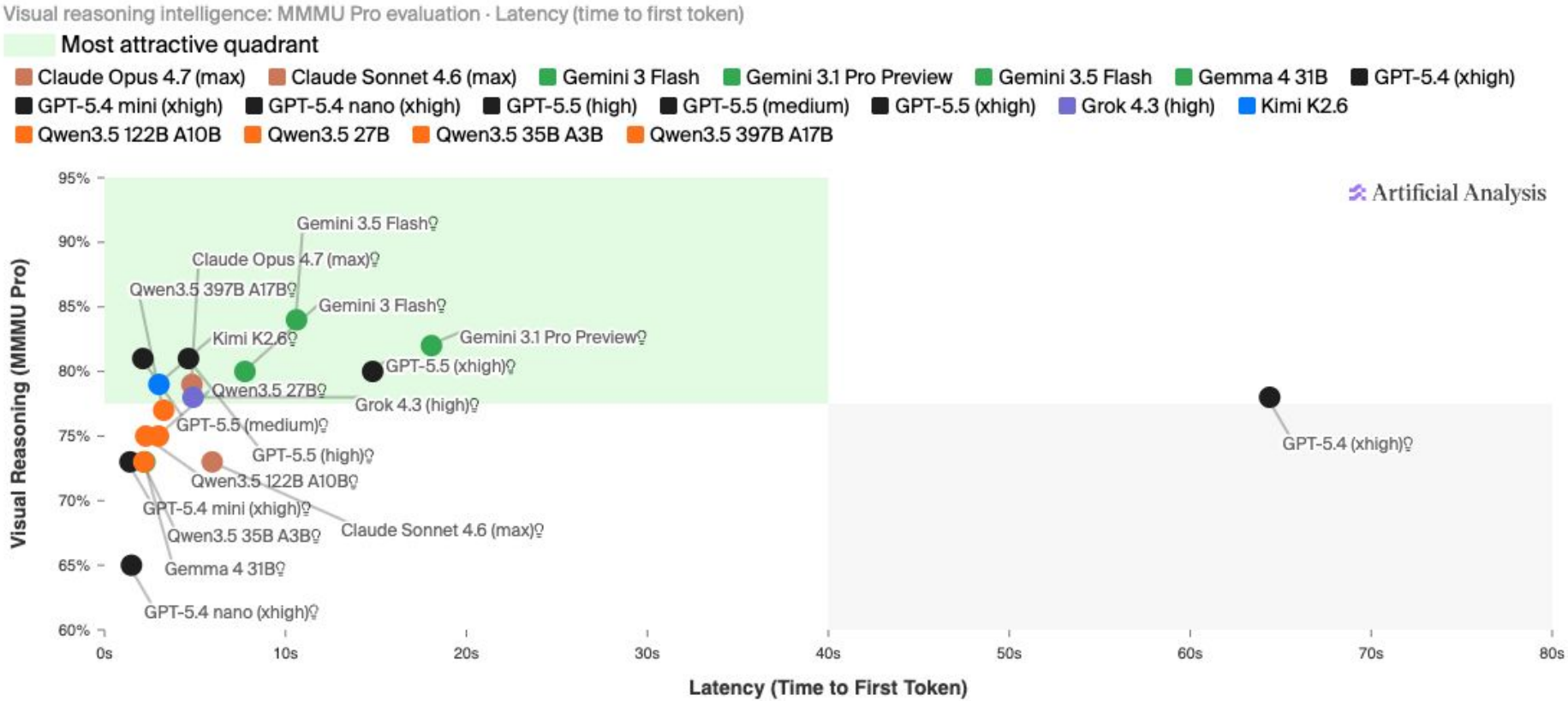
Visual Models Comparison – Visual Reasoning vs. Price*



*as of: May 20, 2026

https://artificialanalysis.ai/models/multimodal/vision

Visual Models Comparison – Visual Reasoning vs. Latency*



*as of: May 20, 2026

https://artificialanalysis.ai/models/multimodal/vision

Dealing with Duplicate Findings

The problem

- AI tools flag the same bug with different wording each run
- Reviewers waste time closing duplicates instead of fixing bugs
- Simple text matching misses rephrased findings

We need to compare by meaning, not just words.
An **embedding** model can be used for that

Example: same bug, different words

Run 1

“Login button on mobile is cut off; users can’t tap submit.”

Run 2

“Sign-in CTA overflows viewport on iPhone — primary action not clickable.”

The second run finding gets linked to the initial on — no duplicate ticket created.

✓ 90% match — treated as a duplicate

Agent Review: Cross-Checking VLM Findings

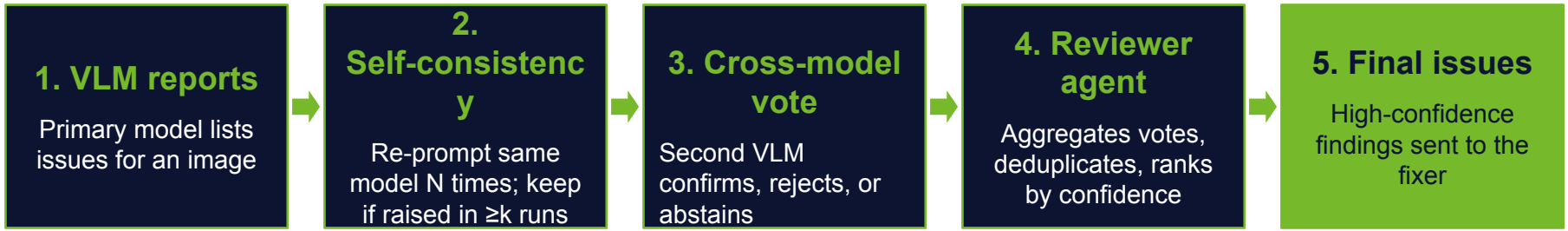
The problem

- An LLM may hallucinate or flag subjective issues
- Hard to separate signal from noise in one pass
- Acting on every report wastes engineering time

Agent-based review

- Reviewer agent rechecks each reported issue
- Self-consistency: N calls to the same model
- Cross-model: a second VLM votes
- Keep issues confirmed above a threshold

How the review loop works



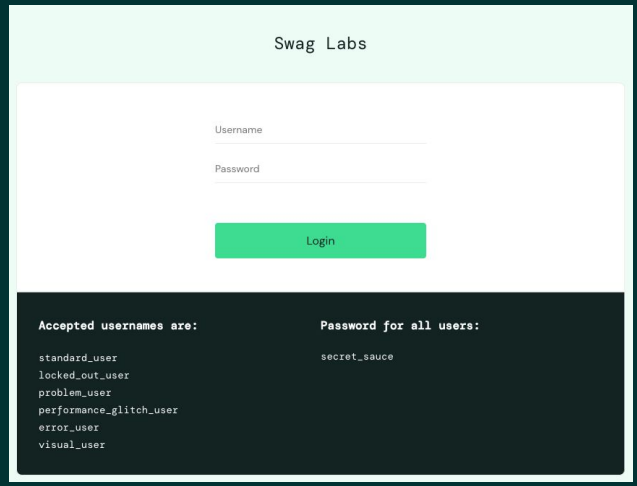
Integrating Visual Reasoning Checks

Drop-in for any framework

Selenium • Playwright • Cypress • Appium

test_visual_check.e2e.ts

```
const result = await ai.check(screenshot,
[
  "The login button is visible",
  "No error messages are displayed",
  "A banner ad is present at the top"
]);
assertVisualResult(result);
```



OUTPUT

```
2 of 3 checks passed.
  PASS  "The login button is visible" — green Login button clearly visible (high)
  PASS  "No error messages are displayed" — no visible error alerts (high)
  FAIL  "A banner ad is present at the top" — no banner ad visible (high)
```

https://github.com/nullp2ike/visual-reasoning

Behind the Scenes: System Prompt

How ai.check transforms statements into structured visual QA results

INPUTS

```
screenshot  //page image
statements[]
"Login button is visible"
"No error messages shown"
"Banner ad at top"
```

SYSTEM PROMPT

Role Assignment

"You are a visual QA assistant"

Evaluation Instructions

1. Evaluate each statement independently
2. Derive pass = AND of all results
3. Summarize reasoning
4. Report issues for failures only

JSON SCHEMA

```
{
  "pass": boolean,
  "reasoning": string,
  "issues": [...],
  "statements": [
    { statement, pass,
      reasoning, confidence }
  ]
}
```

Key Prompt Concepts

Role Definition

Sets the LLM as a visual QA assistant evaluating screenshots precisely

Evaluation Instructions

Statements first, then derive pass as logical AND of all results

Confidence Scoring

High / medium / low confidence per statement based on visual clarity

Structured Issues

Failed statements emit priority, category, description, suggestion

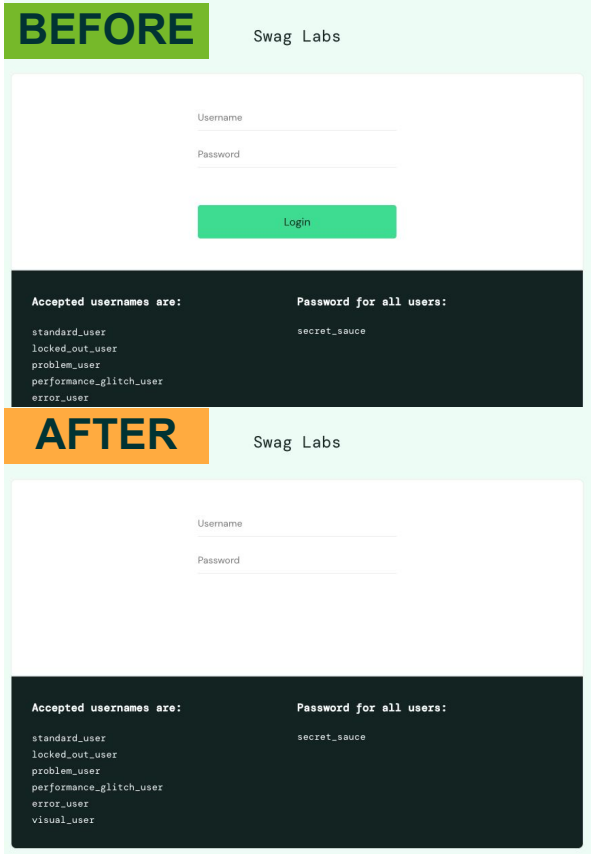
Visual Regression with AI

diff.e2e.ts

```
const result = await ai.compare(
  image('before.png'),  // with login button
  image('after.png')   // without login
);
expect(result.pass).toBe(true);
```

JSON OUTPUT

```
{
  "pass": false,
  "reasoning": "Login button is missing in the
current version",
  "changes": [{
    "description": "Green Login button removed",
    "severity": "critical"
  }]
}
```



https://github.com/nullp2ike/visual-reasoning

Video Assertion Support

https://github.com/nullp2ike/visual-reasoning

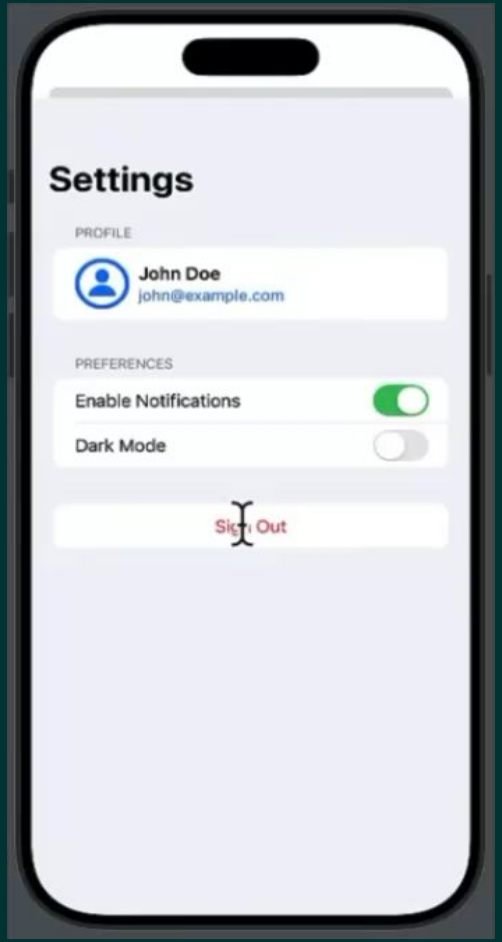
post-deploy.e2e.ts

```
const result = await ai.check(video, [
  'You are signed out.'
]);

expect(result.pass).toBe(true);
```

JSON OUTPUT


```
{
  "pass": true,
  "reasoning": "Toast is visible during sign out flow.",
  "issues": [],
  "statements": [{
    "statement": "\"You are signed out.\" shown in toast",
    "pass": true,
    "confidence": "high",
    "timestampSeconds": 5.5
  }]
}
```

Accessibility Support

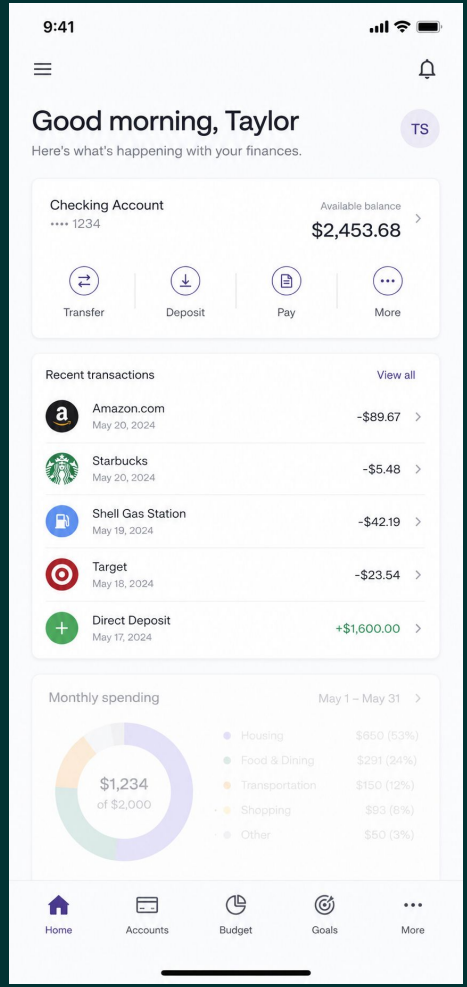
a11y.e2e.ts

```
const result = await ai.accessibility(screenshot, {
  checks: [Accessibility.CONTRAST,
           Accessibility.COLOR_ALONE],
});

expect(result.pass).toBe(true);
```

JSON OUTPUT


```
{
  "pass": false,
  "reasoning": "Primary content readable; some
    small labels appear too light to read.",
  "issues": [{
    "priority": "major",
    "description": "Secondary text and chart labels
      low contrast on the spending card.",
    "suggestion": "Darken light gray and pale purple
      text against the white background."
  }]
}
```

https://github.com/nullp2ike/visual-reasoning

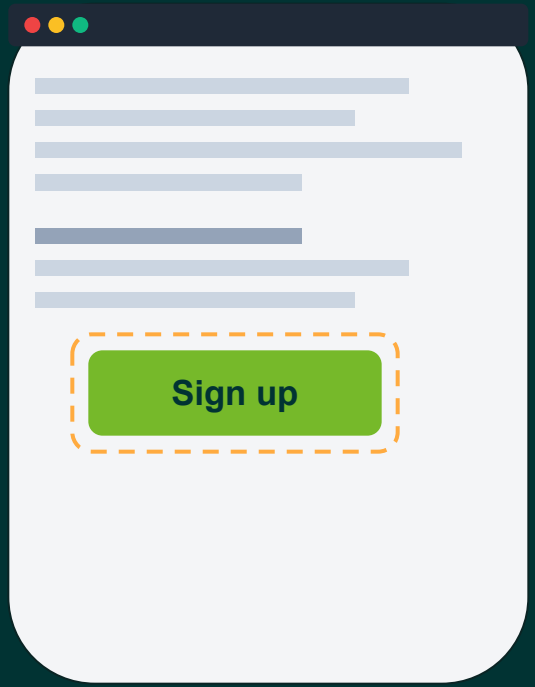
Select Only the Element You Care About

Send the model only the element you care about — sharper answers, lower cost, faster runs.

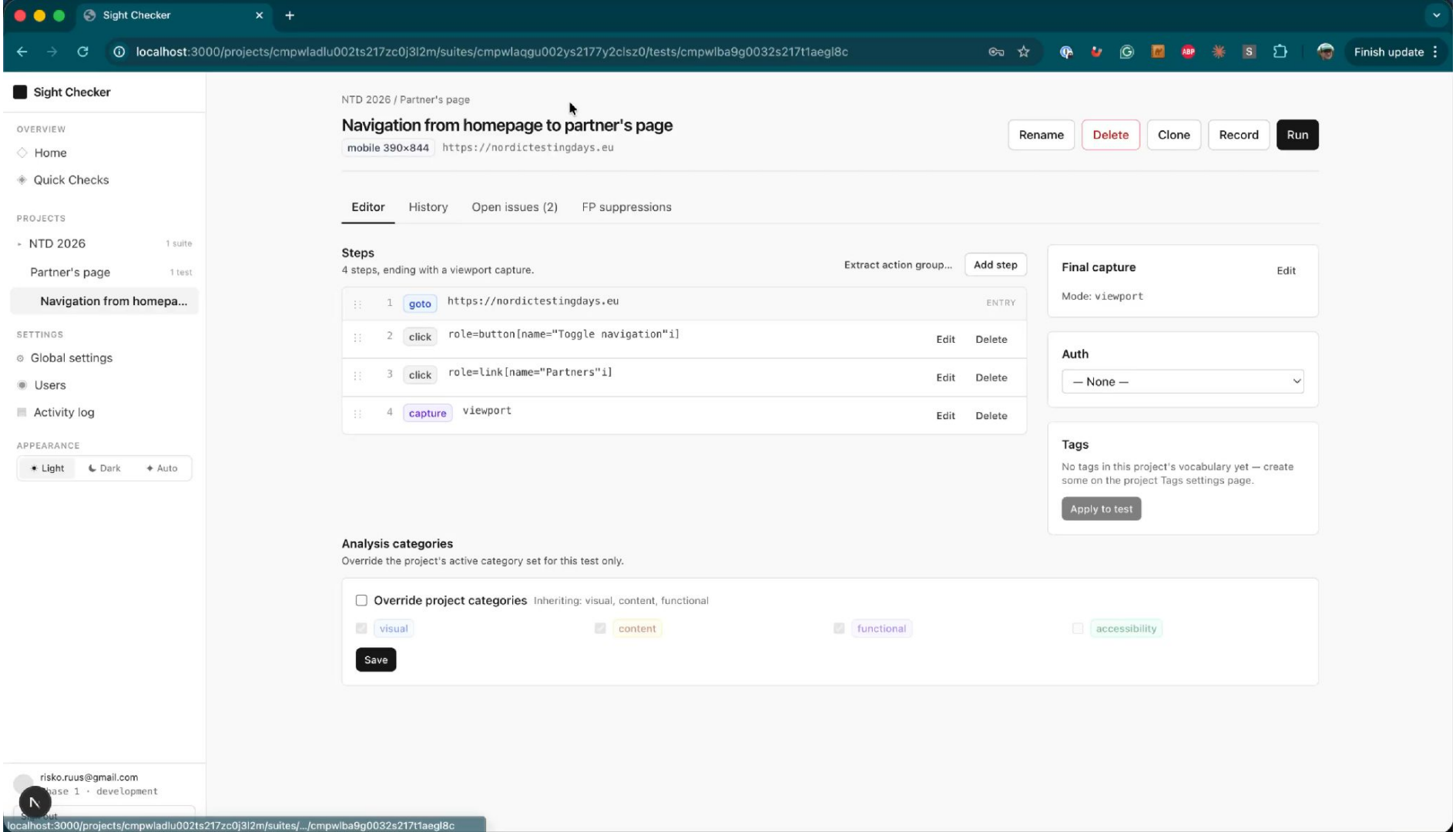
- **Accuracy** — less visual noise, fewer distractors.
- **Cost** — far fewer image tokens per call.
- **Speed** — less data to upload and process.

crop-element.ts

```
// WebdriverIO
const cta = await $('~signupCta');
await cta.saveScreenshot('./cta.png');
// Playwright
const cta = page.getByRole('button', { name: 'Sign up' });
await cta.screenshot({ path: 'cta.png' });
// Hand the element image to the model
await ai.check('cta.png', ['Button says Get started']);
```



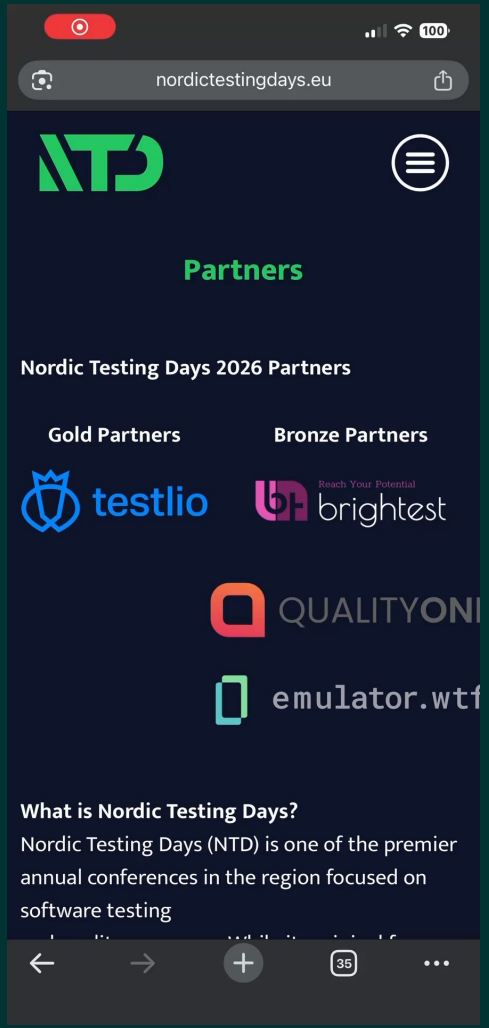
Crop → only this element reaches the model


Sight Checker
localhost:3000/projects/cmpwladlu002ts217zc0j3l2m/suites/cmpwlaqgu002ys2177y2clsz0/tests/cmpwlba9g0032s217t1aegl8c
Finish update
Sight Checker
OVERVIEW
Home
Quick Checks
PROJECTS
NTD 2026
1 suite
Partner's page
1 test
Navigation from homepa...
SETTINGS
Global settings
Users
Activity log
APPEARANCE
Light
Dark
Auto
risko.ruus@gmail.com
base 1 · development
NTD 2026 / Partner's page
Navigation from homepage to partner's page
mobile 390×844
https://nordictestingdays.eu
Rename
Delete
Clone
Record
Run
Editor
History
Open issues (2)
FP suppressions
Steps
4 steps, ending with a viewport capture.
Extract action group...
Add step
1 goto https://nordictestingdays.eu ENTRY
2 click role=button[name="Toggle navigation"i] Edit Delete
3 click role=link[name="Partners"i] Edit Delete
4 capture viewport Edit Delete
Final capture
Edit
Mode: viewport
Auth
— None —
Tags
No tags in this project's vocabulary yet — create some on the project Tags settings page.
Apply to test
Analysis categories
Override the project's active category set for this test only.
Override project categories
Inheriting: visual, content, functional
visual
content
functional
accessibility
Save
localhost:3000/projects/cmpwladlu002ts217zc0j3l2m/suites/.../cmpwlba9g0032s217t1aegl8c

Empowering Manual Testing — Mobile Companion

Visual checks on real devices — a shortcut, your voice, and an LLM verdict on your phone.

- **Device shortcut** — iOS Shortcut or Android quick-tile fires a screenshot of whatever the tester is looking at.
- **Shortcut for the shortcut** — bind it to the phone's action button or to a back double/triple-tap gesture.
- **Voice the assertion** — tester speaks the check: "check the content copy, critical issues only."
- **Sent for analysis** — screenshot + transcribed prompt POSTed to the LLM service; verdict returns in seconds.
- **Companion app log** — pass/fail, reasoning, and the captured image stream into a session log, ready to be submitted as a bug.



Key Takeaways

1. LLMs bring semantic understanding to visual testing — they capture meaning, not just pixels
2. Multiple AI providers offer different strengths — use tiered strategies to balance speed, cost, and accuracy
3. LLMs are **non-deterministic** — keep deterministic tests alongside visual AI checks
4. Integration is straightforward — add AI checks as assertions in existing test suites
5. Start small, measure impact, and don’t be afraid to build tools

Thank you

Questions? Let's keep the conversation going.

GET IN TOUCH

EMAIL
risko.ruus@gmail.com

LINKEDIN
linkedin.com/in/risko-ruus