

THE FAIR AI CODER SCORECARD

A small experiment you can repeat on your own project.

Set up a fair comparison

1. Give Claude, ChatGPT and Gemini the same project, bug and exact prompt. Start every attempt from the same clean copy.
2. Back up your tests somewhere the AI cannot edit them. Restore that copy after each attempt and run the tests yourself.
3. Run each AI at least twice. Keep the model, tool settings and starting files consistent across the comparison.

The task to copy

Fix this bug: [what goes wrong, and the exact error message].
The project is in [folder].
Don't change, delete or skip any test.
When you're done, list every file you changed.

Record each run

AI / run	Fixed?	Original tests pass?	Tests untouched?	Time (min)	Run cost
Claude / 1	_____	_____	_____	_____	_____
Claude / 2	_____	_____	_____	_____	_____
ChatGPT / 1	_____	_____	_____	_____	_____
ChatGPT / 2	_____	_____	_____	_____	_____
Gemini / 1	_____	_____	_____	_____	_____
Gemini / 2	_____	_____	_____	_____	_____

Pick the winner

1. A run qualifies only when the bug is fixed, the original tests pass, and no test was edited, deleted or skipped.
2. Compare time and cost among the AIs that qualify in every run. Record actual run charges when available; do not invent a per-run cost for a flat subscription.

THE PROVE-IT-IS-DONE CHECKLIST

Ask for evidence you can inspect before accepting a fix.

Ask for these three things

1. Tests running and passing, with the real output. A sentence saying "all tests pass" is not the output.
2. Every changed file and a short explanation of what changed.
3. Your original requirements, each marked done or not done.

Then check it yourself

1. Run the relevant tests and try the behavior you requested.
2. Read the list of changes. Ask why any unrequested change was made.
3. Keep the task small enough that you can review the result without guessing. Missing evidence means there is still work to verify.

The prompt to copy

Before you say "done", show me three things:

1. The tests running and passing. Paste the real output.
2. A list of every change you made: each file and what changed.
3. A checklist of everything I asked for, each item marked done or not done.

If any of the three is missing, the task is not done, so don't say "done".

If you changed anything I didn't ask for, list it and tell me why.

THE DRAFT-ONLY EMAIL SETUP

Let the AI write. Keep the final send decision with a person.

Set up the checkpoint

1. Start with a fresh test inbox and fake contacts such as anna@example.test.
2. Give the AI draft-only access and keep sending disabled. A prompt is not an access restriction. If your connector cannot separate drafting from sending, keep the AI outside the inbox and copy the draft yourself.
3. A person checks To, CC, BCC, the subject, full message and every attachment before sending.

Test all three cases

1. No approval: the email stays unsent.
2. Approval refused: the email stays unsent.
3. The AI is asked to send anyway: the email stays unsent.
4. Only move to a real inbox after these checks pass, using the same access limits and approval step.

The prompt to copy

You write email drafts only. You never send email.
For every email, save it as a draft and show me:
- who it's going to (To, CC, BCC)
- the subject and the full text
- every attachment
Then stop and wait. Only I approve and send it.
If I say no, or don't answer, leave it as a draft. Don't send it.

THE RESEARCH-WITH-SOURCES PROMPT

A useful source needs to support the claim, not just open.

The exact prompt to copy

Research [topic]. Put a source link under every single fact, and only use pages I can open. If you can't find a source for something, write 'no source found'. Don't guess and don't invent a link.

Check the sources

1. Replace [topic] with your question. For a comparison, name the products and the criteria you care about.
2. Start with the claims you will actually use, especially numbers, dates and anything that sounds surprising.
3. Open each source and find the sentence, table or data that supports the claim. Use Ctrl+F or Cmd+F to find a name or number.
4. If the page does not support the claim, set the claim aside. If the link will not open, treat it as "no source found".
5. Links do not guarantee a correct answer. Check the wording, publication date and context before relying on the result.

When a source does not support the answer

This page doesn't say [fact]. Find a page that does and link it, or write 'no source found'.

THE SCREENSHOT-TO-WEBSITE CHECKLIST

Give Claude the design and the behavior that an image cannot show.

What to provide

1. A clear, full-page screenshot of your own design or one you are allowed to use.
2. A phone-size screenshot if available, plus notes on what each button, menu and form should do.

The build prompt to copy

Build this page from the screenshot I've attached. Match the layout, colours, fonts, spacing and text.
Make it work on a phone screen too.

The screenshot can't show what happens when you click, so here's how it should work:

- [button]: [what it does]
- [menu]: [what opens]
- [form]: [what happens when it's sent, and the error message if a field is wrong]
- When the page is empty: [what it shows]

If anything isn't clear, ask me instead of guessing.

Check the result

1. Compare the screenshot and the rebuilt page side by side. Check a phone layout too.
2. Click every button and menu. Submit each form and check its error states.
3. Check hover, focus and empty states. Overlay the screenshot at half opacity to list visual differences.
4. Send Claude the list of differences, ask it to fix each one, then repeat the checks.

The fix prompt to copy

Here's every difference between the screenshot and your version: [list]. Fix each one, then tell me what you changed.