# LLM Benchmarks - April 2025

Rinat Abdullin

22 August, 2025

# Table of Contents

1 New OpenAI models: o3-mini, o4-mini, GPT-4.1 ........ 4

2 Qwen3 - pushing state-of-the-art for local models ........ 5

3 Google Is Breathing Down the Neck: Gemini Flash 2.5 Preview and Pro 2.5 v2 ........ 6

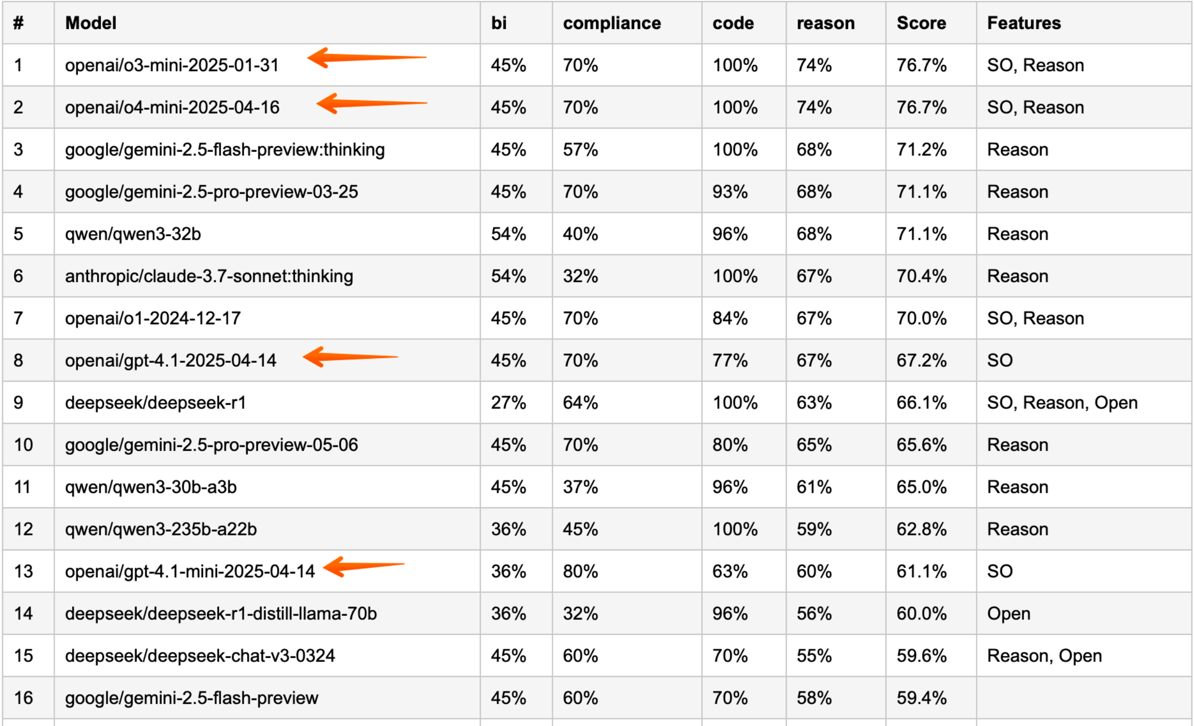
4 Practical insights from AI Coding - Embrace AI ........ 8

In this LLM Benchmark report we are going to cover:

- New OpenAI models - o3-mini, o4-mini and GPT-4.1
- New Qwen3 models - pushing SotA in local models
- New Google Gemini models - breathing down the neck of OpenAI
- Insights from AI Coding - Embrace AI program

# 1 New OpenAI models: o3-mini, o4-mini, GPT-4.1

New models from OpenAI landed nicely in our reasoning leaderboard. o3-mini and o4-mini scored at the top, while less expensive 4.1 (base and mini) took 8th and 13th places. There are not that many surprises here, except for the cost of running these models.



| # | Model | bi | compliance | code | reason | Score | Features |
|---|---|---|---|---|---|---|---|
| 1 | openai/o3-mini-2025-01-31 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 2 | openai/o4-mini-2025-04-16 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 3 | google/gemini-2.5-flash-preview:thinking | 45% | 57% | 100% | 68% | 71.2% | Reason |
| 4 | google/gemini-2.5-pro-preview-03-25 | 45% | 70% | 93% | 68% | 71.1% | Reason |
| 5 | qwen/qwen3-32b | 54% | 40% | 96% | 68% | 71.1% | Reason |
| 6 | anthropic/claude-3.7-sonnet:thinking | 54% | 32% | 100% | 67% | 70.4% | Reason |
| 7 | openai/o1-2024-12-17 | 45% | 70% | 84% | 67% | 70.0% | SO, Reason |
| 8 | openai/gpt-4.1-2025-04-14 | 45% | 70% | 77% | 67% | 67.2% | SO |
| 9 | deepseek/deepseek-r1 | 27% | 64% | 100% | 63% | 66.1% | SO, Reason, Open |
| 10 | google/gemini-2.5-pro-preview-05-06 | 45% | 70% | 80% | 65% | 65.6% | Reason |
| 11 | qwen/qwen3-30b-a3b | 45% | 37% | 96% | 61% | 65.0% | Reason |
| 12 | qwen/qwen3-235b-a22b | 36% | 45% | 100% | 59% | 62.8% | Reason |
| 13 | openai/gpt-4.1-mini-2025-04-14 | 36% | 80% | 63% | 60% | 61.1% | SO |
| 14 | deepseek/deepseek-r1-distill-llama-70b | 36% | 32% | 96% | 56% | 60.0% | Open |
| 15 | deepseek/deepseek-chat-v3-0324 | 45% | 60% | 70% | 55% | 59.6% | Reason, Open |
| 16 | google/gemini-2.5-flash-preview | 45% | 60% | 70% | 58% | 59.4% | |

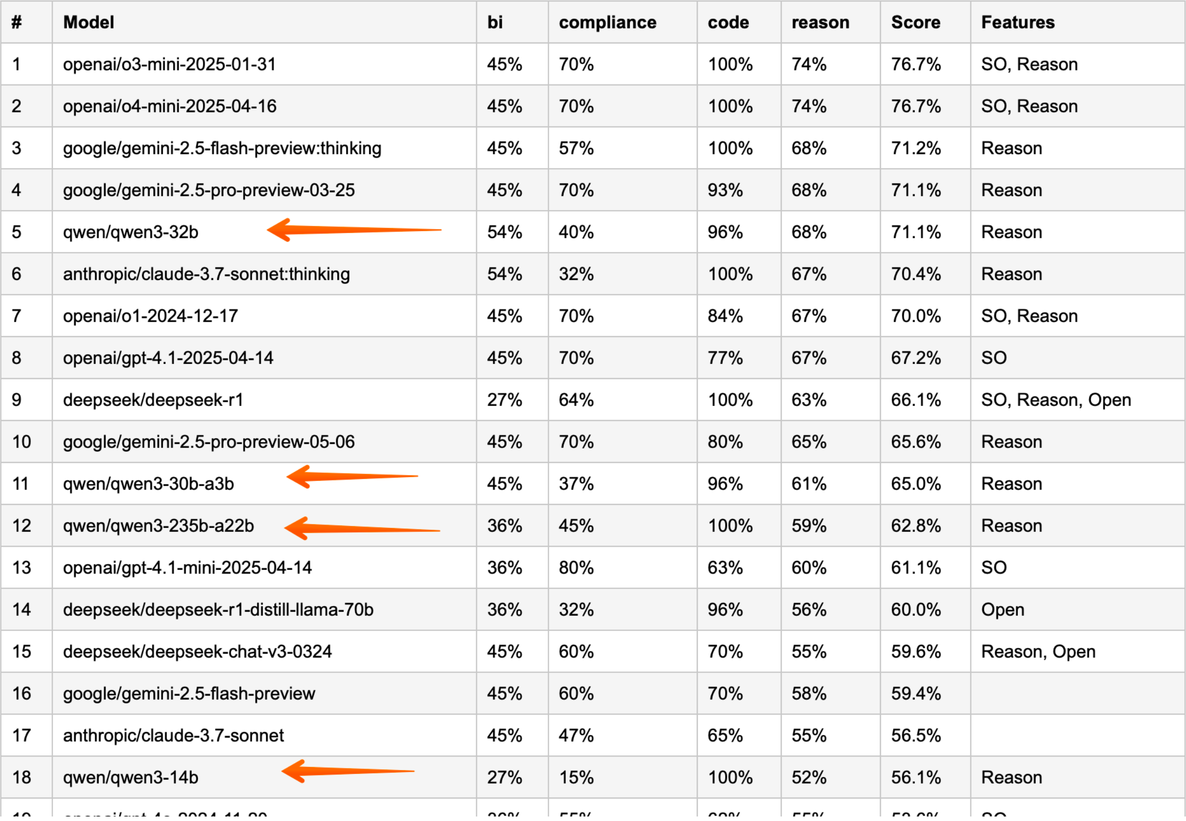
# 2 Qwen3 - pushing state-of-the-art for local models

Qwen models are a well-known “secret” of teams that need to run LLM on their premises. We heard about successful uses of Qwen 2.5 in many projects. This was represented by relatively high scores on our reasoning benchmark.

However **the latest release of Qwen3 models has pushed state-of-the-art even further**. Qwen3 models come in a variety of plain models and Mixture-of-Expert models. They may take longer to process problems, but their accuracy is comparable to the best cloud models. This makes them suitable for these enterprise jobs that require high accuracy and can run over night.

We recommend paying special attention to these two models:

- Qwen3 32B (download on Hugging Face[1]) - 32.8B parameters, 32k native context and 131k extended context
- Qwen3-30B-A3B (download on Hugging Face[2]) - 30.5B parameters, but only 3.3B are activated per token (leading to faster inference)



| # | Model | bi | compliance | code | reason | Score | Features |
|---|---|---|---|---|---|---|---|
| 1 | openai/o3-mini-2025-01-31 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 2 | openai/o4-mini-2025-04-16 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 3 | google/gemini-2.5-flash-preview:thinking | 45% | 57% | 100% | 68% | 71.2% | Reason |
| 4 | google/gemini-2.5-pro-preview-03-25 | 45% | 70% | 93% | 68% | 71.1% | Reason |
| 5 | qwen/qwen3-32b | 54% | 40% | 96% | 68% | 71.1% | Reason |
| 6 | anthropic/claude-3.7-sonnet:thinking | 54% | 32% | 100% | 67% | 70.4% | Reason |
| 7 | openai/o1-2024-12-17 | 45% | 70% | 84% | 67% | 70.0% | SO, Reason |
| 8 | openai/gpt-4.1-2025-04-14 | 45% | 70% | 77% | 67% | 67.2% | SO |
| 9 | deepseek/deepseek-r1 | 27% | 64% | 100% | 63% | 66.1% | SO, Reason, Open |
| 10 | google/gemini-2.5-pro-preview-05-06 | 45% | 70% | 80% | 65% | 65.6% | Reason |
| 11 | qwen/qwen3-30b-a3b | 45% | 37% | 96% | 61% | 65.0% | Reason |
| 12 | qwen/qwen3-235b-a22b | 36% | 45% | 100% | 59% | 62.8% | Reason |
| 13 | openai/gpt-4.1-mini-2025-04-14 | 36% | 80% | 63% | 60% | 61.1% | SO |
| 14 | deepseek/deepseek-r1-distill-llama-70b | 36% | 32% | 96% | 56% | 60.0% | Open |
| 15 | deepseek/deepseek-chat-v3-0324 | 45% | 60% | 70% | 55% | 59.6% | Reason, Open |
| 16 | google/gemini-2.5-flash-preview | 45% | 60% | 70% | 58% | 59.4% | |
| 17 | anthropic/claude-3.7-sonnet | 45% | 47% | 65% | 55% | 56.5% | |
| 18 | qwen/qwen3-14b | 27% | 15% | 100% | 52% | 56.1% | Reason |

Qwen3 release pushes state-of-the-art in local language models and creates more pressure for the cloud providers.

Things don’t stop there, though.

[1] https://huggingface.co/Qwen/Qwen3-32B

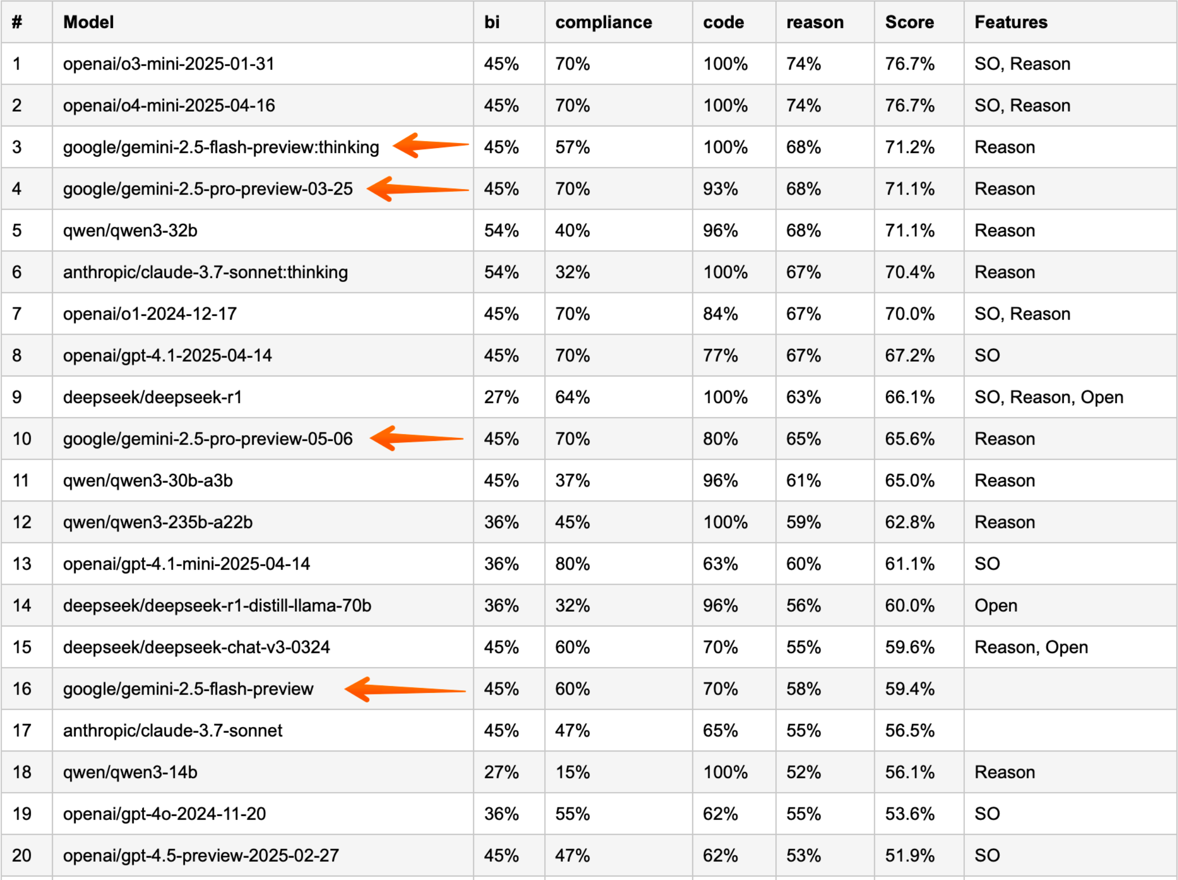
[2] https://huggingface.co/Qwen/Qwen3-30B-A3B

# 3 Google Is Breathing Down the Neck: Gemini Flash 2.5 Preview and Pro 2.5 v2

We all remember how Google was late in the game in the past. OpenAI was much better than first Gemini models, while praised Gemini Ultra - never came out.

Google has managed to gradually shift the tides since then. While it still doesn’t beat the top OpenAI models, it consistently delivers new models within the TOP-10 range, while beating OpenAI on price and stability.

Gemini-2.5-Flash Preview in thinking mode is currently the best Google model on our benchmark. Gemini 2.5 Pro Preview scores 4th and 10th places. Gemini 2.5 Flash in non-thinking mode is at 16th place.



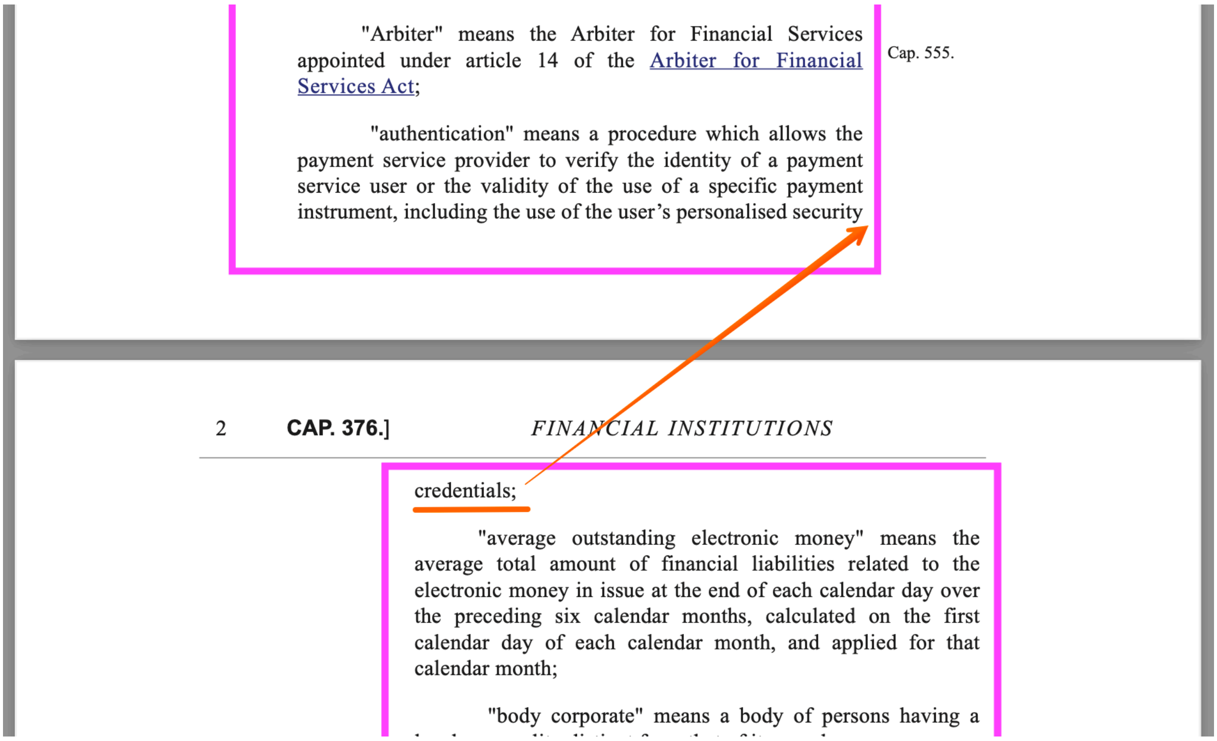
| # | Model | bi | compliance | code | reason | Score | Features |
|---|---|---|---|---|---|---|---|
| 1 | openai/o3-mini-2025-01-31 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 2 | openai/o4-mini-2025-04-16 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 3 | google/gemini-2.5-flash-preview:thinking | 45% | 57% | 100% | 68% | 71.2% | Reason |
| 4 | google/gemini-2.5-pro-preview-03-25 | 45% | 70% | 93% | 68% | 71.1% | Reason |
| 5 | qwen/qwen3-32b | 54% | 40% | 96% | 68% | 71.1% | Reason |
| 6 | anthropic/claude-3.7-sonnet:thinking | 54% | 32% | 100% | 67% | 70.4% | Reason |
| 7 | openai/o1-2024-12-17 | 45% | 70% | 84% | 67% | 70.0% | SO, Reason |
| 8 | openai/gpt-4.1-2025-04-14 | 45% | 70% | 77% | 67% | 67.2% | SO |
| 9 | deepseek/deepseek-r1 | 27% | 64% | 100% | 63% | 66.1% | SO, Reason, Open |
| 10 | google/gemini-2.5-pro-preview-05-06 | 45% | 70% | 80% | 65% | 65.6% | Reason |
| 11 | qwen/qwen3-30b-a3b | 45% | 37% | 96% | 61% | 65.0% | Reason |
| 12 | qwen/qwen3-235b-a22b | 36% | 45% | 100% | 59% | 62.8% | Reason |
| 13 | openai/gpt-4.1-mini-2025-04-14 | 36% | 80% | 63% | 60% | 61.1% | SO |
| 14 | deepseek/deepseek-r1-distill-llama-70b | 36% | 32% | 96% | 56% | 60.0% | Open |
| 15 | deepseek/deepseek-chat-v3-0324 | 45% | 60% | 70% | 55% | 59.6% | Reason, Open |
| 16 | google/gemini-2.5-flash-preview | 45% | 60% | 70% | 58% | 59.4% | |
| 17 | anthropic/claude-3.7-sonnet | 45% | 47% | 65% | 55% | 56.5% | |
| 18 | qwen/qwen3-14b | 27% | 15% | 100% | 52% | 56.1% | Reason |
| 19 | openai/gpt-4o-2024-11-20 | 36% | 55% | 62% | 55% | 53.6% | SO |
| 20 | openai/gpt-4.5-preview-2025-02-27 | 45% | 47% | 62% | 53% | 51.9% | SO |

However benchmarks are only an approximation of the real world. So let me give you two additional data points.

First of all, experienced engineers (with access to Mistral / Anthropic / ChatGPT Pro and Gemini) are starting to gradually **rely more on Gemini 2.5 Pro as their tool of choice in AI Coding.** They praise quality of the answers and reliable work with larger context. For example, it is common to copy essential chunk of the source code into the prompt (up to 50k tokens) and then work interactively with that chat until the overall context reaches 200k-500k.

Another example is about **complex document understanding**. Within the last months we've heard from our peers in AI Research that "if you need to handle enterprise documents reliably - use Google Gemini LLM".

We went on to verify that. In one of the evaluations, we load a compliance PDF into a Graph database for further analysis. The tricky part is that such documents can span hundreds of pages, so they have to be loaded page by page. Another tricky part is that the documents are deeply nested, and it takes extra care to properly attribute clauses across the page breaks.



"Arbiter" means the Arbiter for Financial Services appointed under article 14 of the Arbiter for Financial Services Act;

Cap. 555.

"authentication" means a procedure which allows the payment service provider to verify the identity of a payment service user or the validity of the use of a specific payment instrument, including the use of the user's personalised security

2 **CAP. 376.]** *FINANCIAL INSTITUTIONS*

credentials;

"average outstanding electronic money" means the average total amount of financial liabilities related to the electronic money in issue at the end of each calendar day over the preceding six calendar months, calculated on the first calendar day of each calendar month, and applied for that calendar month;

"body corporate" means a body of persons having a

Models from Anthropic, Mistral, and even OpenAI fail to handle the task of "stitching" content across pages. They get confused easily even within a small context, misrepresenting the original text.

Google Gemini 2.5 Pro, on the other hand, handles the same task (given the same inputs) reliably. This makes it a good contender to OpenAI models when implementing document-heavy AI workflows.

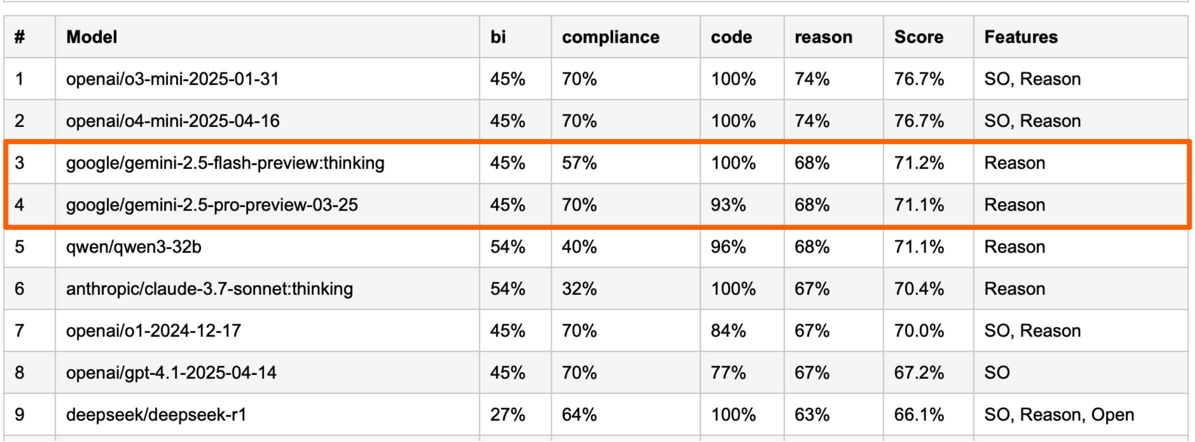
> An important clarification: we are talking about using Gemini 2.5 Pro models through the AI studio or an API. Other Google Gemini AI products that build on top - are currently underwhelming.

# 4 Practical insights from AI Coding - Embrace AI

As a part of the development program at TimeToAct Austria we are running Embrace AI initiative - an experiment in training senior-level software engineers to better understand and use modern AI coding tools. This program is also integrated into our AI Research process, helping to strengthen it with insights from the wider community of experts.

As we've mentioned earlier, the **first insight from this process is that practitioners are gradually shifting from OpenAI and Sonnet 3.7 models to Google Gemini 2.5 Pro** in their everyday tasks. This affects even those developers that used to stand by Sonnet 3.5 for a long time.

The reason for this shift is the effective combination of context size (the amount of context the model can reliably handle), speed, quality, and cost offered by Gemini models.



| # | Model | bi | compliance | code | reason | Score | Features |
|---|---|---|---|---|---|---|---|
| 1 | openai/o3-mini-2025-01-31 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 2 | openai/o4-mini-2025-04-16 | 45% | 70% | 100% | 74% | 76.7% | SO, Reason |
| 3 | google/gemini-2.5-flash-preview:thinking | 45% | 57% | 100% | 68% | 71.2% | Reason |
| 4 | google/gemini-2.5-pro-preview-03-25 | 45% | 70% | 93% | 68% | 71.1% | Reason |
| 5 | qwen/qwen3-32b | 54% | 40% | 96% | 68% | 71.1% | Reason |
| 6 | anthropic/claude-3.7-sonnet:thinking | 54% | 32% | 100% | 67% | 70.4% | Reason |
| 7 | openai/o1-2024-12-17 | 45% | 70% | 84% | 67% | 70.0% | SO, Reason |
| 8 | openai/gpt-4.1-2025-04-14 | 45% | 70% | 77% | 67% | 67.2% | SO |
| 9 | deepseek/deepseek-r1 | 27% | 64% | 100% | 63% | 66.1% | SO, Reason, Open |

This shift is not mainstream - it affects only experienced engineers that are not bound to a specific AI provider and are free to choose the best tools for the job.

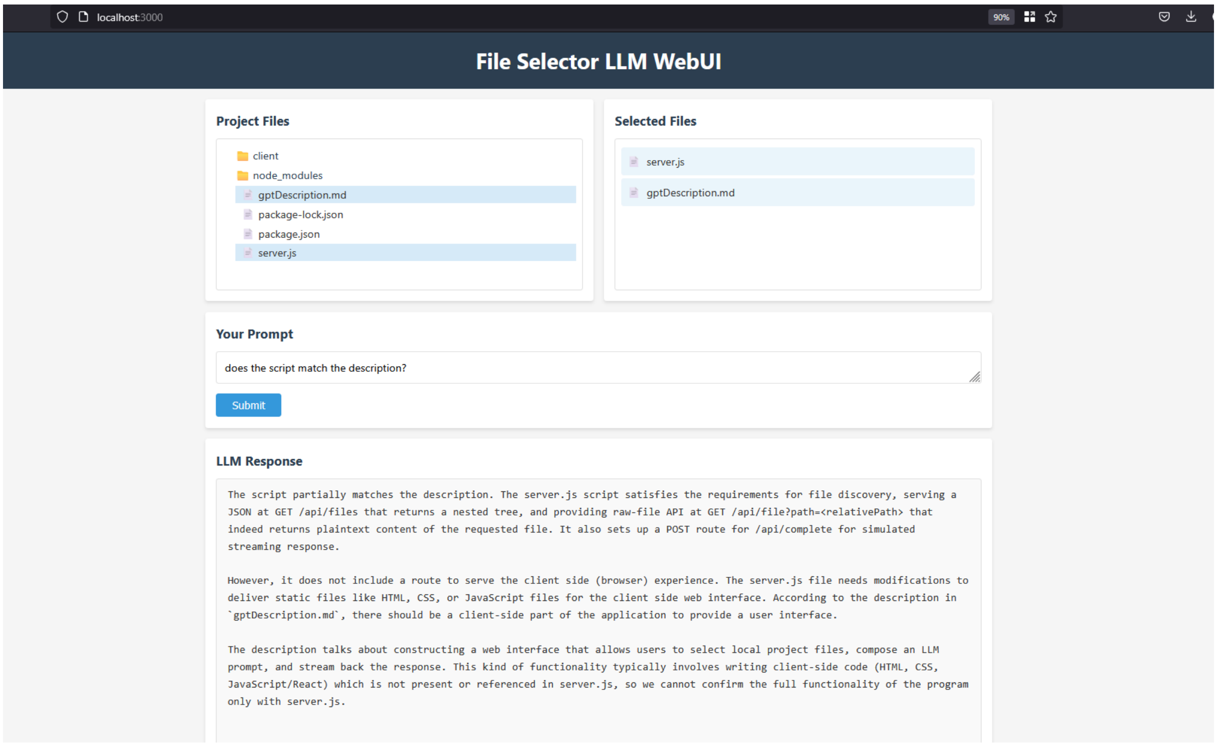
Second insight involves tooling - chats vs complex multi-agent IDEs and environments. To put things into the context, let me provide you with an example of one exercise from our Embrace AI program.

> *The task is following. Please implement a WebUI tool that can send a request to OpenAI/Gemini/LLM of your choice, while appending contents of files into the prompt. You should be able to pick the files that you want to append.*
>
> ***Requirements**:*
>
> - *Tool is passed a directory as argument on startup (e.g. `node server.js ../../projects/demo-project`)*
> - *When loading, it will list all files (recursively) in the left pane*
> - *When user clicks on the file - it is added to the right pane*
> - *If user clicks on the file in the right pane - it is removed*
> - *When user types a prompt and clicks on "Submit", then text of these files is appended to the prompt and sent to LLM.*

> - *Response from LLM is streamed back.*
>
> ***NOT Requirements:***
>
> - *multi-turn chat conversations or follow-up questions.*
> - *any persistence. Reloading the page may lose all information.*

Software engineers were allowed to use any combination of tooling to accomplish this, starting from chats to the most advanced IDEs and Coding agents. We allowed them to experiment, only asking to measure the time.

The fastest solution involved using Anthropic Claude with tools to access powershell and directory. It took 30 minutes and used quite a lot of prompts and tokens under the hood.



does the script match the description?

```
The script partially matches the description. The server.js script satisfies the requirements for file discovery, serving a
JSON at GET /api/files that returns a nested tree, and providing raw-file API at GET /api/file?path=<relativePath> that
indeed returns plaintext content of the requested file. It also sets up a POST route for /api/complete for simulated
streaming response.

However, it does not include a route to serve the client side (browser) experience. The server.js file needs modifications to
deliver static files like HTML, CSS, or JavaScript files for the client side web interface. According to the description in
`gptDescription.md`, there should be a client-side part of the application to provide a user interface.

The description talks about constructing a web interface that allows users to select local project files, compose an LLM
prompt, and stream back the response. This kind of functionality typically involves writing client-side code (HTML, CSS,
JavaScript/React) which is not present or referenced in server.js, so we cannot confirm the full functionality of the program
only with server.js.
```

Other solutions took 1.5h to 2h to implement the code, they involved human control and a bit of prodding the agents.

Writing the same code from the scratch without AI would probably take a few hours, so the best solution with AI provided a noticeable productivity boost. Median solution with AI took the same time it would take an experienced developer to code this app.

However, then we flipped the question and pushed the challenge further:

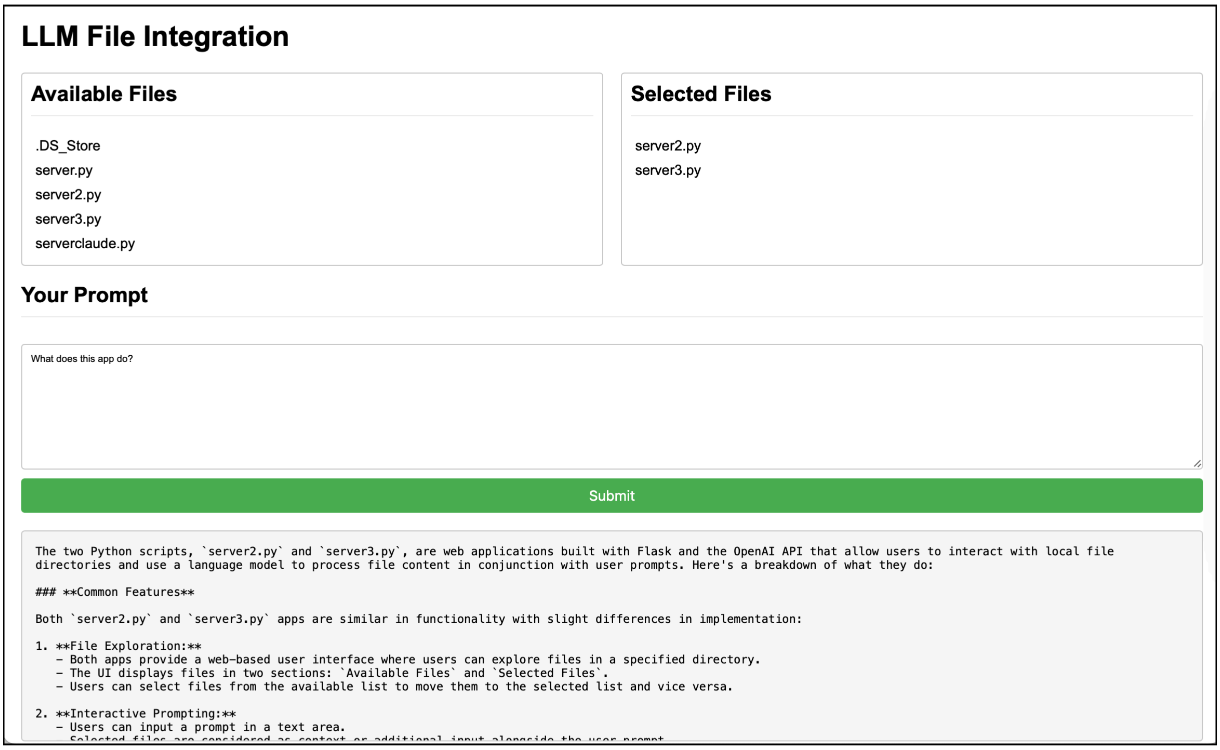
> *Write a prompt, that, once copied to any decent chat LLM model, will produce the code implementing the tool from previous kata straight away (without any agents or complex IDEs? It has to work reliably and across multiple models range of models.*

> *The purpose of the challenge is to build an intuition about how models work and why sometimes you don't need complex tools/agents to accomplish a task.*

The purpose of this exercise was to demonstrate that modern AI tooling gets in the way more often than not, and that understanding the fundamentals of coding with AI can let engineers get better results much faster.

Our own solution to the challenge **took 15 minutes and required 2 steps**:

1. Copy-paste kata to ChatGPT o1 pro, request to ask questions clarifying the task, then capture in succinct prompt. This pattern is called Prompt Distillation in our books.
2. The resulting prompt was 432 tokens (or 1833 characters) long. It produced working solution from the first try, when copied to GPT-o1 pro, GPT 4.5 and Claude 3.7.



What does this app do?

```
The two Python scripts, `server2.py` and `server3.py`, are web applications built with Flask and the OpenAI API that allow users to interact with local file
directories and use a language model to process file content in conjunction with user prompts. Here's a breakdown of what they do:

### **Common Features**

Both `server2.py` and `server3.py` apps are similar in functionality with slight differences in implementation:

1. **File Exploration:**
   - Both apps provide a web-based user interface where users can explore files in a specified directory.
   - The UI displays files in two sections: `Available Files` and `Selected Files`.
   - Users can select files from the available list to move them to the selected list and vice versa.

2. **Interactive Prompting:**
   - Users can input a prompt in a text area.
   - Selected files are considered as context or additional input alongside the user prompt.
```

Another engineer used Gemini 2.5 Pro for Prompt Distillation, arriving to the similar result in 15 minutes (although his final Gemini-generated prompt was 1191 tokens or 4598 characters).

This looks pretty impressive, right? Our AI Research community took the challenge to heart and managed to push it even further. One engineer manually wrote a prompt that is just 70 tokens (313 characters) long and worked on Gemini 2.5 Pro and Sonnet 3.7. It also worked almost out-of-the-box (required one follow-up question) on GPT-4o. The prompt looked like this:

> *Write a node.js/web app - user types a prompt, chooses local text files content which added to the prompt, sees OpenAI gpt-4o LLM response. App shows dir files recursively (dir name is a command*

> *line param of node.js server), chosen files, prompt input, submit button. Click on chosen files removes it from chosen*

Another engineer wrote in 7 minutes a prompt (194 tokens+298 custom instructions) that produced valid result on Claude 3.7.

Our conclusions to all that are:

- Modern AI Coding tools can bring real productivity gains, when used properly.
- Practice and experience can beat fancy tooling. In fact, complex AI tools can often get in the way of productivity.
- Claude 3.7 Sonnet and Gemini 2.5 Pro are currently preferred by the AI Coding practitioners.