Starting April 29, 2025, Gemini 1.5 Pro and Gemini 1.5 Flash models are not available in projects that have no prior usage of these models, including new projects. For details, see Model versions and lifecycle (/vertex-ai/generative-ai/docs/learn/model-versions#legacy-stable).

Give clear and specific instructions

Giving the model instructions on what to do is an effective and efficient way to customize model behavior. Be clear, concise, and direct in the instructions, providing the needed context and outlining the desired output. Instructions can be as simple as a list of step-by-step instructions or as complex as mapping out a user's experience and mindset.

How to write effective instructions

Effective instructions apply the following principles:

- Tell the model what to do.
- Be clear and specific.
- Specify any constraints or formatting requirements for the output.

Example

For example, suppose you own a cheeseburger restaurant and you want to use a model to help you learn about which menu items are the most popular. You want the model to format transcripts of customer orders in JSON so that you can quickly identify menu items. The following tables demonstrate the different responses the model provides when given generic instructions and specific instructions that follow the instruction principles:

eneric instructions
:

```
ct the transcript in JSON.
cript:
```

```
er 1 (Customer): Hi, can I get a cheeseburger and large fries, please?
er 2 (Restaurant employee): Coming right up! Anything else you'd like to add to your
?
er 1: Hmmm, maybe a small orange juice. And could I get the fries with ketchup on th

er 2: No problem, one cheeseburger, one large fries with ketchup on the side, and a
e juice. That'll be $5.87. Drive through to the next window please.
```

---

se:

```
{
  "speakers": [
    {
      "speaker": "Customer",
      "utterances": [
        "Hi, can I get a cheeseburger and large fries, please?"
      ]
    },
    {
      "speaker": "Restaurant Employee",
      "utterances": [
        "Coming right up! Anything else you'd like to add to your order?"
      ]
    },
    {
      "speaker": "Customer",
      "utterances": [
        "Hmm, maybe a small orange juice. And could I get the fries with ketchup on
        side?"
      ]
    },
    {
      "speaker": "Restaurant Employee",
      "utterances": [
        "No problem, one cheeseburger, one large fries with ketchup on the side, an
        small orange juice. That'll be $5.87. Drive through to the next window plea
      ]
    }
  ]
}
```

(gem

In the example, generic instructions lead to a response with more JSON fields than what you need to identify the menu items customers are ordering. Improve the instructions by making more specific output constraints, as demonstrated in the following table:

ecific instructions

:

```
ct the items from this transcript in JSON and separate drinks from food.
cript:
er 1 (Customer): Hi, can I get a cheeseburger and large fries, please?
er 2 (Restaurant employee): Coming right up! Anything else you'd like to add to your
?
er 1: Hmmm, maybe a small orange juice. And could I get the fries with ketchup on th
,
er 2: No problem, one cheeseburger, one large fries with ketchup on the side, and a
e juice. That'll be $5.87. Drive through to the next window please.
```

se:

```
{
  "drinks": [
    "orange juice"
  ],
  "food": [
    "cheeseburger",
    "large fries"
  ]
}
```

(gem

By applying the instruction principles, this example went from a response that contained too much data to one that contained exactly the data required for this use case. Instructions that use the instruction princples can help you guide the model to deliver the most helpful response for your use case.

What's next

- Explore more examples of prompts in the Prompt gallery (/vertex-ai/generative-ai/docs/prompt-gallery).

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the Apache 2.0 License (https://www.apache.org/licenses/LICENSE-2.0). For details, see the Google Developers Site Policies (https://developers.google.com/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated 2025-07-30 UTC.