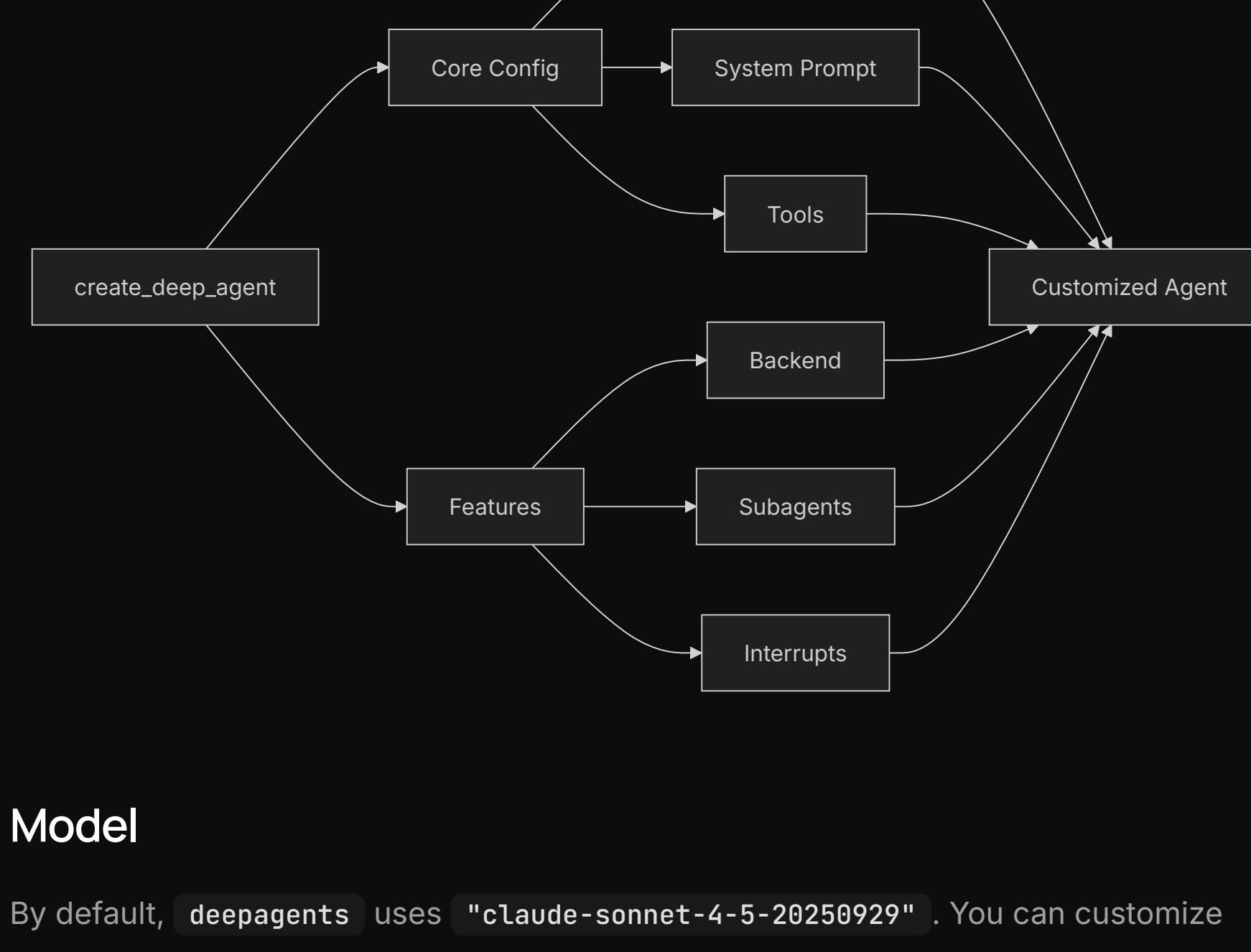


Get started

## Customize Deep Agents

Copy page

Learn how to customize deep agents with system prompts, tools, subagents, and more



### Model

By default, `deepagents` uses "claude-sonnet-4-5-20250929". You can customize this by passing any [LangChain model object](#).

```

from langchain.chat_models import init_chat_model
from deepagents import create_deep_agent

model = init_chat_model(
    model="gpt-3",
)
agent = create_deep_agent(
    model=model,
)
  
```

### System prompt

Deep agents come with a built-in system prompt inspired by Claude Code's system prompt. The default system prompt contains detailed instructions for using the built-in planning tool, file system tools, and subagents.

Each deep agent tailored to a use case should include a custom system prompt specific to that use case.

```

from deepagents import create_deep_agent

research_instructions = """\
You are an expert researcher. Your job is to conduct \
thorough research, and then write a polished report. \
"""

agent = create_deep_agent(
    system_prompt=research_instructions,
)
  
```

### Tools

Just like tool-calling agents, a deep agent gets a set of top level tools that it has access to.

```

import os
from typing import Literal
from tavity import TavityClient
from deepagents import create_deep_agent

tavity_client = TavityClient(api_key=os.environ["TAVILY_API_KEY"])

def internet_search(
    query: str,
    max_results: int = 5,
    topic: Literal["general", "news", "finance"] = "general",
    include_raw_content: bool = False,
):
    """Run a web search"""
    return tavity_client.search(
        query,
        max_results=max_results,
        include_raw_content=include_raw_content,
        topic=topic,
    )

agent = create_deep_agent(
    tools=[internet_search]
)
  
```

In addition to any tools that you provide, deep agents also get access to a number of default tools:

- `write.todos` – Update the agent's to-do list
- `ls` – List all files in the agent's filesystem
- `read_file` – Read a file from the agent's filesystem
- `write_file` – Write a new file in the agent's filesystem
- `edit_file` – Edit an existing file in the agent's filesystem
- `task` – Spawn a subagent to handle a specific task

Edit the source of this page on GitHub.

Connect these docs programmatically to Claude, VSCode, and more via MCP for real-time answers.

Was this page helpful?

Yes

No