

# Designing AI Infrastructure Experiences

Deepesh Sudhakar



# A high-level overview of multi-sprint AI platform work

Every initiative highlighted in this deck represents a multi-sprint design cycle requiring deep technical discovery, cross-functional alignment, and systems-level thinking. My role involved leading these projects from initial ambiguity to detailed product execution alongside engineering and product partners.

# Product areas covered across the AI platform ecosystem

Shaping the DigitalOcean AI platform into a connected ecosystem designed to support the full developer lifecycle from model selection to production monitoring.

Core Area	Focus Initiatives
Discovery	Model Studio + Model Catalog
Experimentation	Multimodal Playground
Deployment	Serverless Inference Enhancements
Evaluation	Agent Evaluations + Human-in-the-loop Feedback
Data & RAG	Knowledge Base Activity Logs + Auto Indexing

# Streamlining model discovery through a unified catalog

## Problem

Developers evaluating AI models needed a trusted, centralized environment to discover and compare options before testing.

## Design Action

I unified the Model Studio concept with a catalog-led experience, surfacing benchmark signals, model details, and direct comparison entry points.

## Why it Mattered

This approach created a "front door" for the AI experience, significantly lowering the friction for model adoption.

## Outcome / Signal

This initiative established a clearer discover-to-evaluate path and reduced the ambiguity inherent in choosing between competing LLMs.

The screenshot displays the 'Inference Hub' interface. On the left is a navigation sidebar with categories like PROJECTS, MANAGE, and various services. The main area shows a 'Model Catalog' with a search bar and a table of models. A modal window is open for the 'llama3.3-70b-instruct' model, showing its details and pricing.

Name	Type	Capabilities	Price	Availability
llama3.3 70b instruct	Text-to-Text /	Enhanced reasoning,	\$3.00/M input tokens	Serverless / Dedicated
Claude 3.7 Sonnet	AI	Anthropic Claude 3.7 Sonnet		Serverless / Dedicated
GPT OSS 120b		openai-gpt-oss-120b		Serverless / Dedicated
GPT 5		openai-gpt-5		Serverless / Dedicated
R1 Distill Llama 70b		deepseek-r1-distill-llama-70b		Serverless / Dedicated
GPT 5 mini		openai-gpt-5-mini		Serverless / Dedicated
Nemo Instruct	Text-to-Text / Multimodal	Enhanced reasoning, long-context chat, coding, analysis	\$3.00/M input tokens / \$15.00/M output tokens	Serverless / Dedicated

**Model Details for llama3.3-70b-instruct:**

- Model Details:** llama3.3-70b-instruct
- Capabilities:** Text Calling, Reasoning Model
- Model Details:** API Usage
- Description:** Llama 3.3 70B Instruct is a large instruction-tuned language model designed for high-quality reasoning, complex task execution, and conversational AI, delivering strong performance across coding, analysis, and multi-step problem solving at scale.
- License:** License Agreement
- Availability:** Serverless Inference (Instant on-command model execution layer), Dedicated Inference (Advanced infrastructure with dedicated GPUs)

# Lowering the barrier to experimentation with multimodal playgrounds

## Problem

Developers struggled to understand how different models behaved across text, image, and audio tasks before committing to an inference path.

## Design Action

Designed a flexible playground for side-by-side exploration, featuring modular modality inputs and real-time output previews.

## Why it Mattered

Experimentation is the bridge to production; a low-friction testing environment builds developer confidence in model reliability.

## Outcome / Signal

Successfully created a more intuitive bridge from discovery to deployment decisions by making system behavior transparent.

The screenshot displays the 'Inference Hub' interface, which is designed for comparing different AI models. The interface is split into two columns, each representing a different model: 'Llama 3 Instruct 8-B' on the left and 'Mistral Nemo' on the right. Each column has a dropdown menu at the top to select the model, followed by a text input area where users can enter prompts. Below the input area, there are chat bubbles showing the model's responses to various prompts such as 'What's an agent?', 'What's a knowledge base?', and 'How do I compare two models?'. The interface also includes a 'Compare Model' section at the top, a search bar, and a navigation menu on the left. At the bottom, there are two main options: 'Serverless Inference' (Instant on-demand model execution layer) and 'Dedicated Inference' (Always-on infrastructure with dedicated GPUs), each with a 'Create Key' or 'Deploy endpoint' button. The interface is clean and modern, with a dark blue sidebar and a white main content area.

# Simplifying the path to production with serverless inference

## Problem

Developers required a faster path to configure and deploy inference without being bogged down by infrastructure complexity.

## Design Action

I refined the setup and configuration flows, focusing on API guidance and clear deployment usage screens.

## Why it Mattered

Reducing setup friction allows developers to focus on application logic rather than server management.

## Outcome / Signal

Improved the clarity of the inference setup journey, making the DigitalOcean experience feel more robust and production-ready for enterprise use.

The screenshot displays the DigitalOcean GenAI Platform interface. On the left is a navigation sidebar with categories like PROJECTS, MANAGE, and App Platform. The main content area is titled 'GenAI Platform' and features a 'Serverless Inference' tab. Below this, there are sections for 'Model Endpoint', 'Usage', 'Model Access Keys', 'CHAT COMPLETIONS', and 'MODEL LIST'. The 'CHAT COMPLETIONS' section shows a code editor with Python code for interacting with the Gradient SDK. The 'MODEL LIST' section shows a code editor with Python code for listing models. A 'Create Model Access Key' button is visible in the 'MODEL ACCESS KEYS' section.

Search by resource name or public IP (Cmd+I) Create Design Onboarding Estimated costs: \$0.00

### GenAI Platform

Agents Serverless Inference Knowledge bases Guardrails Model provider keys

#### Model Endpoint

About endpoints

With a model endpoint you can use models from industry-leading providers without hosting or maintaining them yourself with token-based billing.

[Try and compare models in Model Playground](#)

#### Usage

Supported parameters

#### CHAT COMPLETIONS

Chat completions create a model response for the chat conversation supplied in the request. See a list of [supported models](#).

#### MODEL LIST

Returns a list of available models with their corresponding id values, which you use as the model parameter in your inference requests.

#### Model Access Keys

Grants you access to any model hosted by DigitalOcean through a model endpoint.

[Create Model Access Key](#)

No model access keys added.

[About model access keys](#)

#### CHAT COMPLETIONS

Gradient SDK llama3.3-70b-instruct

```
import os
from gradient import Gradient

inference_client = Gradient(
    inference_key=os.environ.get(
        "GRADIENT_MODEL_ACCESS_KEY"
    ),
)

inference_response =
inference_client.chat_completions.create(
    messages=[
        {
            "role": "user",
            "content": "What is the capital of
France?"
        }
    ],
    model="llama3.3-70b-instruct",
)

print(inference_response.choices[0].message.content)
```

Copy

#### MODEL LIST

Gradient SDK

```
from gradient import Gradient

client = Gradient()
models = client.models.list()
print(models.links)
```

Copy

# Turning black-box agent results into observable workflows

## Problem

Teams building autonomous agents lacked structured ways to inspect and evaluate behavior beyond manual judgment.

## Design Action

Designed evaluation dashboards that help users review outputs, score performance, and compare agent traces over time.

## Why it Mattered

Systematic evaluation is critical for AI safety and performance iteration.

## Outcome / Signal

Framed agent quality as a reviewable product workflow rather than a hidden result, allowing for more predictable software development.

The screenshot shows a web application interface for setting up an evaluation test case. The interface is divided into a left sidebar and a main content area. The sidebar contains navigation options: PROJECTS, MANAGE, App Platform, Agent Platform, Model Studio (highlighted), Functions, Kubernetes, Volume Back Storage, Databases, Spaces Object Storage, Container Registry, Images, Networking, Monitoring, and Add-Ons. Below these are sections for 'By DigitalOcean' (Billing, Support, Settings) and 'API' (NEW). At the bottom of the sidebar are links for Marketplace, Product Docs, and What's New.

The main content area has a search bar at the top with the text 'Search by resource name or public IP (Cmd+B)'. To the right of the search bar are buttons for 'Create', a notification bell, and a help icon. The top right corner shows 'Design Onboarding' with 'Estimated costs: \$0.00' and a 'DO' logo.

The main content area is titled 'Set up an evaluation test case'. Below the title is a subtitle: 'Evaluations are an excellent way to determine if your agent is ready for production scenarios, test against hard-to-predict scenarios, and measure agent performance improvement as you make changes. Test cases are reusable templates against which your agent is evaluated, and can be used for any agent in its evaluation workspace.' There is a link for 'About agent evaluations'.

Below this is a section titled 'What Qualities Do You Want To Evaluate Your Agents For?'. The text says: 'Select at least one category to evaluate your agent against, or select from the full list of available metrics. Then, select at least one star metric.'

There are two panels for selecting metrics. The left panel is titled 'General Agent Quality' and includes a description: 'Evaluate agent for hallucinations, answer quality, and safety and security scenarios.' It has a table with the following metrics and star ratings:

METRIC	STAR METRIC
<input checked="" type="checkbox"/> Correctness (general hallucinations)	☆
<input checked="" type="checkbox"/> Instruction following	☆
<input checked="" type="checkbox"/> Ground truth faithfulness <small>Includes BLEU and ROUGE-1</small>	☆
<input type="checkbox"/> Prompt perplexity	☆
<input type="checkbox"/> Personally identifiable information detection	☆
<input type="checkbox"/> Toxicity	☆
<input type="checkbox"/> Tone	☆
<input type="checkbox"/> Sexism	☆
<input type="checkbox"/> Prompt injection detection	☆
<input checked="" type="checkbox"/> User goal progress	☆
<input checked="" type="checkbox"/> User goal completion	☆

The right panel is titled 'RAG and Tool Use Correctness' and includes a description: 'Evaluate how accurate, relevant, and successful your agent is regarding delivering responses related to its attached resources and context.' It has a table with the following metrics and star ratings:

METRIC	STAR METRIC
<input checked="" type="checkbox"/> Context adherence (context hallucinations)	★
<input checked="" type="checkbox"/> Response-context completeness	☆
<input checked="" type="checkbox"/> Retrieved context relevance	☆
<input type="checkbox"/> Retrieved chunk relevance	☆
<input type="checkbox"/> Retrieved chunk usage	☆
<input type="checkbox"/> Add use percentage	☆
<input type="checkbox"/> Route selection and success	☆
<input type="checkbox"/> Route errors	☆

At the bottom of the left panel is a link for 'About general agent quality metrics'.

# Capturing human judgment to improve model reliability

## Problem

AI systems need human feedback loops to correct behavior and improve accuracy, but feedback capture is often intrusive.

## Design Action

Developed lightweight feedback and review patterns that capture actionable signals without disrupting the developer's primary workflow.

## Why it Mattered

Trust in AI is built on the ability to course-correct model outputs through human expertise.

## Outcome / Signal

Introduced clearer pathways for turning subjective user judgment into structured data points for model fine-tuning.

The image displays a 'Conversation log stream' interface. At the top, there's a header with 'Demos / robs-agent / Conversation log stream'. Below it, a table lists several 'request' entries with timestamps from 8/6/25. A 'Traces' button is highlighted in the top navigation. A chat window titled 'Customer support agent' is overlaid on the right, showing a conversation. The user asks 'When does the shop open?' and the agent replies 'The shop opens at 8am Tuesday to Friday and 9am Saturday and Sunday.' The user then asks 'Can I buy coffee in bulk?' and the agent replies 'Yes, we offer coffee in bulk, whole bean or ground.' A 'Feedback' button is visible in the chat window, and a 'Thumb Annotation' tooltip shows a thumbs-down icon with the text 'took too long'. The background interface includes a search bar, filters, and a 'Log Stream Insights' button.

# Building trust through transparent knowledge base activity

## Problem

Users indexing data for RAG applications lacked visibility into job success, failures, and why specific items were skipped.

## Design Action

Designed comprehensive activity logs that visualize indexing progress, job-level outcomes, and detailed failure states.

## Why it Mattered

Transparency in data processing reduces troubleshooting time and builds trust in the underlying knowledge engine.

## Outcome / Signal

Made indexing behavior more inspectable, allowing users to manage large-scale data ingestion with significantly reduced ambiguity.

The screenshot displays the GenAI Platform interface. On the left is a navigation sidebar with categories like PROJECTS, MANAGE, App Platform, GenAI Platform, Droplets, Functions, Kubernetes, Volumes, Block Storage, Databases, Spaces, Object Storage, Container Registry, Images, Networking, Monitoring, Add-Ons, By DigitalOcean, Billing, Support, and Settings. The main content area shows the 'knowledge-base-name' page with tabs for Overview, Data Source, Activity, and Settings. The 'Activity' tab is active, displaying 'LATEST INDEXING DETAILS' with a 'Partially Indexed' status. Below this is a summary table and a list of 'All Activity' logs.

Activity	Value
Newly Indexed	987 of 1,234 (79%)
Failed	47
Skipped	200
Indexing event cost	\$60.75 / 675M tokens

**All Activity**

- Partially Completed Indexing Job**  
Indexed 987 of 1234 files/URLs (79%)  
\$60.75 (675M tokens @ \$0.09/1M)  
Today at 12:44 PM [Name](#) [Download Details](#) [Re-run](#)
- Completed Indexing Job**  
Indexed 1234 of 1234 files/URLs (100%)  
\$60.75 (675M tokens @ \$0.09/1M)  
August 8 at 12:44 PM [Name](#) [Download Details](#)
- Stopped Indexing Job**  
Indexed 987 files/URLs  
\$60.75 (675M tokens @ \$0.09/1M)  
August 5 at 1:44 AM [Name](#) [Download Details](#)
- Failed Indexing Job**

# Automating knowledge maintenance for continuous AI accuracy

## Problem

Maintaining an up-to-date knowledge base was a manual burden, leading to stale data in AI-driven answers.

## Design Action

Designed automated scheduling and management patterns for recurring indexing, including state-based controls like pause and resume.

## Why it Mattered

Automation transforms maintenance into a background workflow, ensuring AI reliability while preserving user control.

## Outcome / Signal

Defined a clearer operational model for keeping AI knowledge sources current without manual intervention.

The screenshot displays the Agent Platform interface. On the left is a navigation sidebar with categories like PROJECTS, MANAGE, and various services. The main area shows the 'knowledge-base-name' configuration page with tabs for Overview, Data sources, Activity, and Settings. A modal dialog titled 'Create Indexing Schedule' is open, providing an 'Info' section with key points: every indexing job incurs charges, users are only charged for detected changes, and they should review details before scheduling. Below this, users can select days (Mon, Wed, Thurs) and a trigger time (12pm:00 UTC). A summary states the schedule will run on those days at the specified time. 'Cancel' and 'Create Indexing Schedule' buttons are at the bottom.

# Driving platform impact through design leadership

My work across DigitalOcean's AI initiatives is grounded in three core leadership pillars:



## Systems Thinking

I connected discovery, evaluation, and deployment into a coherent platform narrative rather than siloed features.



## Product Judgment

I balanced aggressive roadmap goals with technical constraints and the need for a polished, user-centric experience.



## Execution Leadership

I led cross-functional design efforts across multiple product surfaces, ensuring high-fidelity delivery in a fast-paced AI environment.

*Together, these pillars ensure that design is not just a visual layer, but a strategic driver of product success.*

# Selected AI platform work, with deeper case studies available

Let's discuss the product context, design trade-offs, and full case studies in detail.

This deck provides a high-level summary of selected DigitalOcean AI/ML platform work. For a deeper look at the design process, user research findings, and specific metrics, I would be happy to walk through full case studies during a scheduled video call.

Thank you for your time.