DESIGN ANYTHING, DESIGN AT SCALE

Anonymous authors
Paper under double-blind review

ABSTRACT

As multimodal resources are transformed into media outputs, agentic AI is expected to align with design priors. Recent work dynamically reuses prior design experience in long-horizon agentic design, yet struggles to keep to the brief over extended workflows. In contrast, designers typically work from an initial brief that specifies project goals, constraints, and expected deliverables, which provides persistent guidance throughout the design process. This highlights the need for long-horizon agents that can preserve and consistently apply design intent throughout the entire workflow. To address this, we propose `Design Anything`, a harness that turns design intent into explicit project state: requirements and decisions persist in a task contract, flow into later generations through a workflow graph, and, together with execution records, decide when a run is complete. We further introduce `DesignAnything-Bench`, a benchmark of 228 tasks across 19 design domains, including 38 project families each posed at 6, 14, and 24 artifacts with shared context and inputs. On this benchmark, `Design Anything` satisfies 85.0% of the brief's constraints with DeepSeek V4.1 Flash High and 92.5% with GPT-5.6 Sol High, 15.4 and 14.4 percentage points above the strongest baseline using the same model.

1 INTRODUCTION

Human design activity often begins with a specific need, such as developing a product, staging an exhibition, or building a brand. A client sets out the project's goals, constraints, and deliverables in a design brief, and designers work from it. Such projects usually call for a set of coordinated artifacts, including concept images, renderings, specification documents, videos, and slide decks, and later

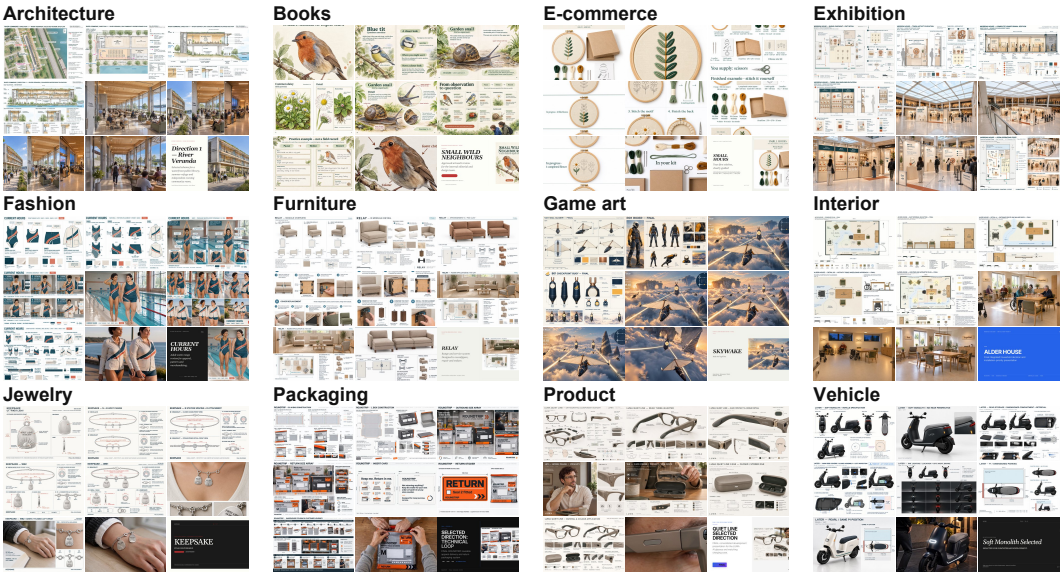


Figure 1: **Design Anything across twelve design domains.** Each group presents nine artifacts from a single run on a task from a different design domain, with GPT-5.6 Sol as the language model.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

work builds on decisions made earlier, such as the selected concept and references (Eppinger et al., 1994). Generative models and design tools can now produce individual artifacts (OpenAI, 2026; ByteDance Seed, 2026; Tripo, 2026; Zheng et al., 2025; Ge et al., 2025), and agents have begun to use them to carry out design work over multiple steps (Wang et al., 2026). As the number of deliverables grows, an agent must follow the same design prior, preserving the design intent of the brief and carrying earlier design decisions forward over a longer execution, and long-horizon tasks remain difficult for agents (Kwa et al., 2025; Liu et al., 2024b; Xie et al., 2024). In long-horizon agentic design, how well an agent performs depends not only on its language model but also on its harness, the software layer that manages its tool calls, context, and multi-step execution (Lee et al., 2026; Wu et al., 2026; Yang et al., 2024; Agashe et al., 2025; Luo et al., 2026).

Existing agentic design systems (Anthropic, 2026; Open Design contributors, 2026; Lin et al., 2026) and general agent frameworks mostly keep requirements and intermediate results in interaction histories, requirement records, or canvases (Yao et al., 2023; Shinn et al., 2023; Ma et al., 2026; Lin et al., 2026), and assess progress through the acting agent, a separate reviewing agent, or the node states of a plan graph (Wei, 2026). Some also reuse design experience accumulated over earlier tasks (Luo et al., 2026). None of them keeps all three of the following explicit throughout a project: the brief's requirements, earlier design decisions, and which deliverables are complete. Yet long executions tend to lose exactly this information. Longer contexts do not ensure that models reliably use the information they contain (Liu et al., 2024a; Hong et al., 2025), performance declines when requirements are revealed over multiple turns (Laban et al., 2026), and analyses of long-horizon tool use report premature termination, insufficient verification, and lost constraints (Yuan et al., 2026; Merrill et al., 2026). Therefore, this raises a natural question: *how can agentic AI consistently align with design intent over long-horizon tasks?*

To this end, we propose `Design Anything`, a harness for long-horizon agentic design (Figure 2). The agent records the design intent (deliverables, constraints, and design decisions) in a task contract that persists throughout the project. It incrementally builds a workflow graph, linking each deliverable to the generation steps that produce it and passing selected concepts and references to later steps as explicit inputs (Figure 10). A deterministic controller derives the status of each deliverable from execution records and from checks on output kind and count; while deliverables remain outstanding, it starts another turn that names them. The agent chooses the next action but cannot mark a deliverable as complete by declaration alone. Figure 1 showcases the versatility of `Design Anything`, with additional examples provided in Appendix E.

Evaluating such a harness requires a suitable benchmark, which is still missing. Some existing benchmarks include professional tasks with several deliverables (Sun et al., 2026; Li et al., 2026b), among them design and creative tasks (Li et al., 2026b); others focus on workflow construction, tool planning, UI reproduction, poster generation, or single-image quality (Xue et al., 2025; Ki et al., 2025; Jeong et al., 2026; Pang et al., 2025; Li et al., 2026a). None of them poses the same project at different sizes, so none can isolate whether an agent keeps every deliverable faithful to the brief and to earlier design decisions as a project grows. We introduce `DesignAnything-Bench` for this purpose, with 228 tasks across 19 design domains. Within it, 38 project families each have three versions requesting 6, 14, or 24 artifacts. `Design Anything` satisfies 85.0% of the brief's constraints on DeepSeek V4.1 Flash High and 92.5% on GPT-5.6 Sol High, 15.4 and 14.4 percentage points above the strongest baseline on the same language model, and leads by 25.1 and 23.0 points on 24-artifact tasks. Furthermore, we conduct a system-blind human evaluation with professional designers, with `Design Anything` achieving the two highest preference estimates across the five configurations. Our contributions are threefold:

- **Design Anything Harness.** We propose `Design Anything`, a general-purpose harness that persistently tracks deliverables, constraints, and design decisions to maintain brief adherence throughout long-horizon tasks.
- **DesignAnything-Bench.** We introduce `DesignAnything-Bench`, a 228-task benchmark spanning 19 design domains, with 38 project families scaled from 6 to 24 artifacts to systematically test long-horizon consistency.
- **Comprehensive Evaluation.** `Design Anything` substantially improves brief adherence over strong baselines across language models and project lengths, and achieves the highest human preference.

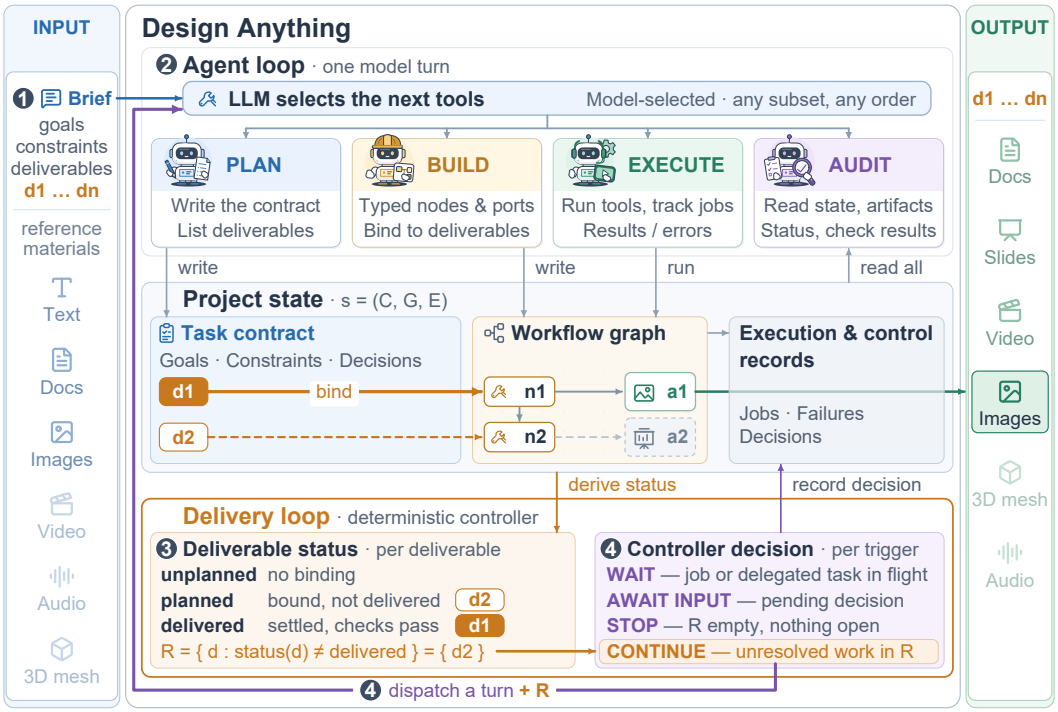


Figure 2: **Overview of Design Anything.** ❶ A design brief states goals, constraints, and deliverables, which the agent records in a task contract. ❷ In each turn, the agent builds and runs a workflow graph that passes chosen concepts and references to later steps. ❸ After each turn, a deterministic controller derives each deliverable's status; ❹ it waits while work is running, ends the run successfully once nothing is outstanding, and otherwise starts a turn naming the outstanding set $\mathcal{R}$. Faded icons mark modalities not benchmarked. Figures 10–21 show recorded workflow graphs.

2 Method

Following work that separates a model's weights from the harness around it (Lee et al., 2026; Ren et al., 2026), we hold the language model π_θ fixed and write the artifacts Design Anything exports from a design brief x and reference materials A as $Y \sim H(\pi_\theta, x, A)$. Design Anything organizes the work around a project state and two loops (Figure 2): in each turn, the agent loop selects tools that build and run a workflow; after each turn, the delivery loop derives from the state which deliverables are complete and decides whether to end the run or start another turn.

2.1 Problem formulation

A brief x states the project's goals, a set of constraints K, and a set of deliverables $D = \{d_1, \ldots, d_n\}$. Deliverables share content: several product views, a specification document, and a slide deck may all describe the same selected concept. A run must produce files for every $d \in D$ and carry the constraints in K and the design decisions made along the way into the deliverables they concern. We measure a harness by the constraint satisfaction of its exported files (Equation 6),

$$J(H) = \mathbb{E}_{(x,A)\sim p_{\text{task}},\, Y\sim H(\pi_\theta,x,A)}\big[\mathrm{CS}(Y;x)\big], \tag{1}$$

where p_{task} is the task distribution, and analogously by delivery completion. The more deliverables a project has, the longer it runs (Appendix C.7), and at the start of every turn the agent must again establish what is still owed and which decisions to reuse.

2.2 Project state

The brief is turned into a project state that every turn reads and updates (❶ in Figure 2):

$$s = (C, G, E), \qquad C = (D, K, \Psi), \qquad \beta : D \to 2^{N_G}. \tag{2}$$

The task contract C holds the deliverables D, the constraints K, and the recorded design decisions Ψ, together with goals, success criteria, and open questions; it records requirements, never progress, so the model cannot mark a deliverable complete by editing it. The workflow graph G is a typed directed graph over operations N_G whose edges connect compatible ports, so a chosen concept or a supplied reference enters later generations as an explicit input rather than as a fresh description (Section 3.4.3 measures how often this happens). The binding β maps each deliverable to the nodes that produce it, and the records E log which operations have settled or are still running and what the controller decided. Appendix A gives the implementation details.

2.3 Agent loop

In each turn (❷ in Figure 2), the model does not write s directly. It observes a view $\omega(s)$: the canvas, which renders G and its artifacts, and a compact canvas digest regenerated from s between turns. It then selects operations $a \sim \pi_\theta(\cdot \mid x, A, \omega(s))$ from four tool families, in any order: PLAN writes and revises C; BUILD adds nodes, edges, and deliverable bindings to G; EXECUTE runs nodes and returns artifacts; and AUDIT reads C, G, and E. An operation takes effect only if a deterministic validity check V accepts it:

$$s' = \begin{cases} \Phi(s, a) & \text{if } V(s, a), \\ s & \text{otherwise,} \end{cases} \tag{3}$$

where Φ is the operation's state transition. V checks tool arguments, node configurations, references, port types, and bindings; a rejected operation leaves the state unchanged and returns an actionable error to the model.

2.4 Delivery loop

Whenever a turn settles, an asynchronous job wakes the controller, or a pending decision is answered (❸ in Figure 2), a deterministic controller derives from the bindings in G and the records in E whether each deliverable has been delivered:

$$\delta_s(d) = \mathbf{1}\big[\beta(d) \neq \emptyset \wedge \mathrm{settled}_s(\beta(d)) \wedge \mathrm{ok}_s(d)\big], \qquad \mathcal{R}(s) = \{d \in D : \delta_s(d) = 0\}, \tag{4}$$

where $\mathrm{settled}_s(\beta(d))$ holds when every executable node in $\beta(d)$ has settled, and $\mathrm{ok}_s(d)$ when the bound outputs pass the checks on ownership, output kind, and item count. The outstanding set $\mathcal{R}(s)$ thus contains both unbound (*unplanned*) and bound but unfinished (*planned*) deliverables. Let $W(s)$ be the jobs and delegated tasks in flight and $Q(s)$ the questions and decisions awaiting input. A run ends successfully only when

$$\mathrm{Stop}(s) \iff W(s) = \emptyset \wedge Q(s) = \emptyset \wedge \mathcal{R}(s) = \emptyset, \tag{5}$$

so a model's statement that it has finished does not end the run (❹ in Figure 2). The controller then waits while W or Q is non-empty, ends the run when Stop holds, and otherwise starts a new turn whose message names $\mathcal{R}(s)$. A run ends unsuccessfully, for example, when an operation bound to a deliverable fails or when repeated decisions show no progress; Appendix A gives the order of these checks.

These checks establish operational completion only. The model still chooses the bindings, and no runtime check confirms that a produced artifact is the one the brief asked for; aesthetic quality, semantic correctness, and natural-language constraints are scored by the benchmark from the exported files. Section 3.4.4 compares the full harness with runs that end when the model stops and with runs that answer every stop with a generic prompt to continue, both also without the task contract.

3 Experiments

3.1 Benchmark: DesignAnything-Bench

Tasks. Each of the 228 tasks supplies an English design brief, input materials, and explicit delivery requirements, across 19 design domains. Of these, 38 project families are each posed at 6, 14, and 24 requested artifacts with shared background and inputs, so project length is an experimental variable; 57 reference-material tasks add supplied objects the outputs must match, and 57 breadth

tasks add further subjects and audiences (Appendix B). An evaluation specification derived only from the public brief encodes each requirement as an output category with an expected quantity or as a factual constraint.

Measures. The evaluator scores only the delivered files, as in Agent-as-a-Judge (Zhuge et al., 2025), and reports each measure separately (Appendix Table 3). *Requirement fulfillment* compares the outputs with the brief. *Delivery completion* is the share of requested files delivered, counted per requested category and capped at the requested quantity. *Constraint satisfaction* is the share of the brief's factual constraints, such as names and dimensions, whose fact f_c appears verbatim in every file of its target set $T(c)$:

$$\mathrm{CS} = \frac{1}{|K|} \sum_{c \in K} \mathbf{1}[\, f_c \in y \;\; \forall\, y \in T(c)] \,, \tag{6}$$

where K is the set of applicable constraints; text in generated images is read by a transcription step. *Reference fidelity* is the share of output–reference pairs that depict the same object as an asset the client supplied. *Cross-artifact consistency* compares the outputs with each other: *identity consistency* is the share of output–anchor pairs that depict the same object as the run's designated anchor artifact, and *set consistency* is a 0–100 rubric score for visual rules the set should share, such as palette and typography. *Per-image quality* gives four 0–100 checklist scores for technical quality, aesthetics, text rendering, and domain criteria (Li et al., 2026a). *Human evaluation* is carried out by professional designers (Section 3.5). A classifier assigns delivered files to requested categories; reference fidelity and both consistency measures use a judge calibrated against two professional designers ($\kappa = 0.887$ and 0.885 on items where the designers agree, close to their mutual agreement of 0.931 and 0.893; Appendix B.6); per-image quality uses a vision-language judge.

3.2 Experimental setup

Systems. We compare Design Anything with two design agents, OpenDesign (Open Design contributors, 2026) and JarvisHub (Lin et al., 2026), and with the general coding agent Codex CLI. Each comparison is a *track* that holds the language model fixed and varies the harness. In the DeepSeek track, Design Anything, OpenDesign, and JarvisHub run on DeepSeek V4.1 Flash High; in the GPT track, Design Anything and Codex run on GPT-5.6 Sol High. All systems generate images with gpt-image-2.5-sunburst except Codex, which uses its built-in image tool. Appendix C lists what each comparison holds fixed (Table 4).

Protocol. All systems receive the same briefs and materials, and one evaluator scores their exported files. No system used skills written for a particular design domain or received task-specific hints. Each task contributes one final result per system; paired gaps come with 95% intervals from a bootstrap over project families (Appendix C.3).

3.3 Main results

DeepSeek track. With the model and image generator fixed, Design Anything leads OpenDesign and JarvisHub on constraint satisfaction (85.0% against 69.6% and 68.0%; Figure 3), reference fidelity, and identity consistency (Table 1a). These gains come with per-image scores within 1.4 points of JarvisHub's, which is higher on all four (Appendix C.6).

GPT track. Design Anything satisfies 92.5% of the constraints against 78.1% for Codex and leads on reference fidelity, identity consistency, and every per-image score; Codex delivers slightly more files (98.1% against 97.3%) and scores 2.1 points higher on set consistency (Table 1b).

Qualitative comparison. In the project of Figure 4, each baseline leaves files undelivered or reduces views and drawings to schematic geometry, whereas Design Anything delivers every file with rendered views and detailed drawings on either language model.

Table 1 Results on DesignAnything-Bench (228 tasks, one final result per task). Each panel fixes the language model named in its header and varies the harness. Delivery, constraint, reference, and identity are percentages; set and per-image scores are 0–100; higher is better. Bold: best within a panel; per-image scores are not bolded, and their paired differences with intervals are in Appendix C.6. Reference and identity count pairs the judge could not compare as mismatches; excluding them shrinks the leads of Design Anything (Appendix C.3). †58.4 with such pairs excluded.

	Requirements			Consistency		Per-image quality			
System	Delivery	Constraint	Reference	Identity	Set	Quality	Aesthetics	Text	Domain
(a) DeepSeek track *Model: DeepSeek V4.1 Flash High*									
Design Anything	**96.6**	**85.0**	**75.6**	**84.5**	87.9	91.5	90.2	96.3	95.2
OpenDesign	95.2	69.6	55.0	75.2	**88.3**	89.4	86.4	84.7	89.8
JarvisHub	87.6	68.0	49.6†	68.5	65.4	92.3	91.6	96.7	96.1
(b) GPT track *Model: GPT-5.6 Sol High*									
Design Anything	97.3	**92.5**	**79.3**	**90.2**	87.7	91.2	91.0	96.4	95.1
Codex	**98.1**	78.1	71.3	75.3	**89.8**	90.6	89.6	96.1	92.6

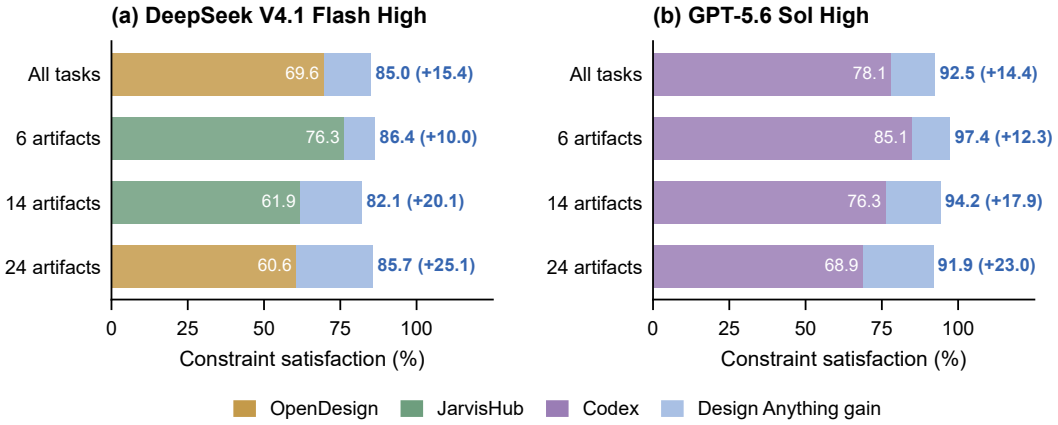


Figure 3: **Constraint-satisfaction gain of Design Anything over the best baseline on the same language model,** over all tasks and at each project length. Colored bars show the best baseline; labels give the rate of Design Anything and the paired gap in points.

3.4 Analysis

3.4.1 How does task length affect long-horizon design?

The gap to baselines widens as projects grow longer. From 6 to 24 requested artifacts, the constraint satisfaction of Design Anything changes by at most 5.5 points on either model, while JarvisHub loses 21.9 points and Codex 16.1 (Figure 5b). The gain over the best baseline on the same model therefore grows from 10.0 to 25.1 points on DeepSeek and from 12.3 to 23.0 on GPT-5.6 Sol (Figure 3). Every paired gap at 14 and 24 artifacts excludes zero. The growth of the gap between 6 and 24 artifacts is resolved against JarvisHub (21.3 points, $[3.8, 40.0]$) but not against OpenDesign ($[-5.3, 24.0]$) or Codex ($[-3.5, 24.7]$) (Table 7).

Delivery does not explain the gap. Design Anything and Codex deliver more than 94% of the requested files at every length (Figure 5a), while their constraint satisfaction separates. JarvisHub's delivery falls to 81.8% at 24 artifacts, and constraints on categories it never delivered account for part of its loss there: excluding them narrows its 24-artifact gap from 31.3 to 26.9 points (Table 9).

On GPT-5.6 Sol, per-image quality also diverges with length. From 6 to 24 artifacts, Codex's domain and aesthetics scores fall by about 5 points, while those of Design Anything change by at most 1.6. At 24 artifacts Design Anything leads on domain criteria by 6.6 points ($[4.5, 9.1]$). On

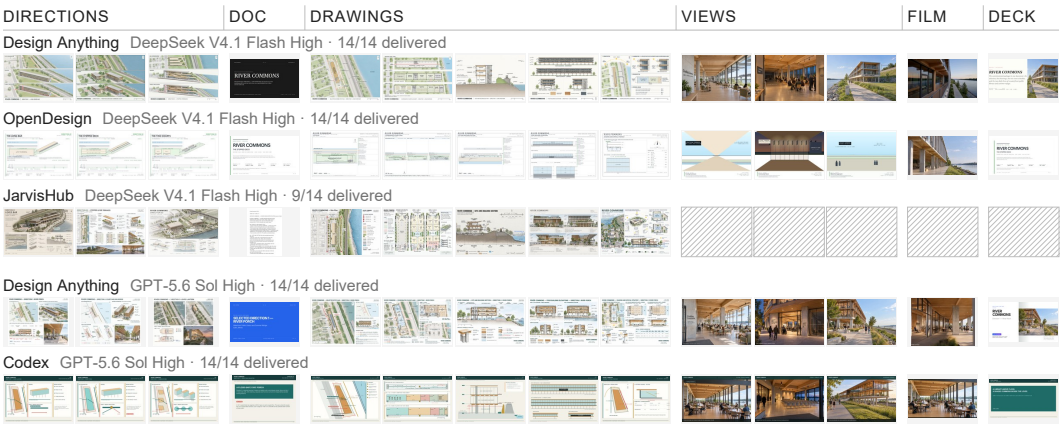


Figure 4: **The same 14-deliverable brief across all five systems: only Design Anything delivers every file with both rendered views and detailed drawings.** Rows are systems; columns are the requested deliverables in brief order; hatched cells were not delivered. JarvisHub leaves the views, film, and deck undelivered; OpenDesign renders its views as flat geometry; Codex reduces its plans, section, and elevations to flat diagrams. The project was selected for display; Figures 22 and 23 show every output of each system.

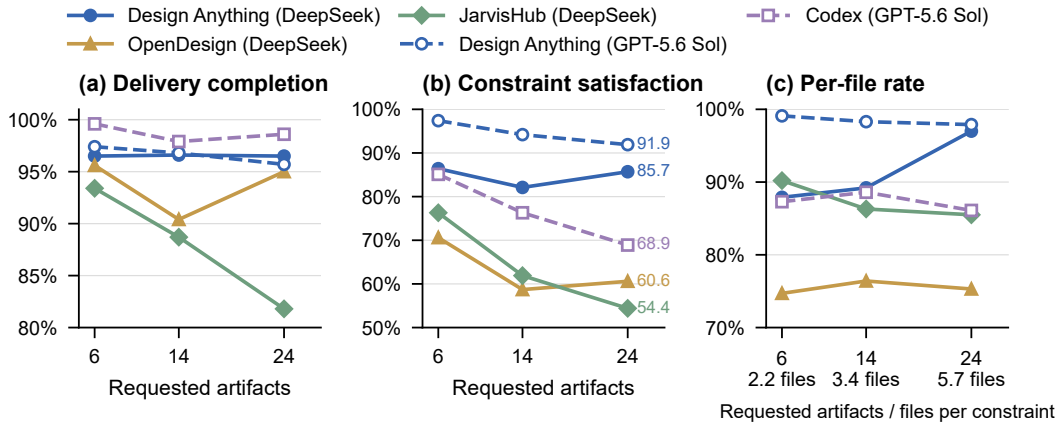


Figure 5: **Performance across project lengths.** (a) Delivery completion and (b) constraint satisfaction at 6, 14, and 24 requested artifacts; DeepSeek V4.1 Flash High solid, GPT-5.6 Sol High dashed. (c) Per-file rate: the share of (constraint, target file) pairs whose file contains the fact; a constraint is satisfied only when all of its pairs are. The second axis line gives the mean number of target files per constraint. Each length contains the same 38 project families.

DeepSeek, length does not separate Design Anything from JarvisHub on any per-image criterion (Appendix C.6).

3.4.2 Where are constraints lost in long-horizon tasks?

Facts are produced but not carried to every file. A constraint passes only if its fact appears in every file it targets, and longer projects spread each fact over more files: the mean number of target files per constraint rises from 2.2 at 6 artifacts to 3.4 at 14 and 5.7 at 24 (Figure 5c). The per-file rate counts each constraint–file pair on its own, so it measures how often a fact reaches one file. No system's per-file rate falls by more than 5 points with length, so a per-file gap compounds as the number of targets grows. On GPT-5.6 Sol, Design Anything carries the fact in 97.9–99.1% of target files and Codex in 86.1–88.6%. Of Codex's 191 unmet constraints with per-file records, 117 already appear in at least half of their targets. The fact was produced but not propagated to every file that needs it, which is the failure the task contract and the workflow graph are designed to prevent.

Table 2 Effect of autonomous continuation on 19 tasks with 24 artifacts, one per domain, with DeepSeek V4.1 Flash High and one run per task. Parentheses give the change from *No continuation*. Collapsed runs deliver fewer than half of their deliverables; generation requests are per run. †Language-model cost per run at list prices on the 17 tasks with full telemetry (Appendix D.2).

Arm	Delivery (%)	Collapsed	Constraint (%)	Gen. requests	Cost ($)†
No continuation	73.9	6	57.5	18.6	0.48
Generic continuation	96.9 (+23.0)	0	75.3 (+17.8)	36.1	1.30
Complete (Design Anything)	96.1 (+22.2)	0	**90.4** (+32.9)	24.1	**0.43**

3.4.3 Where does cross-artifact consistency come from in long-horizon tasks?

References are wired before generation. JarvisHub also records reference use as typed edges between canvas nodes (Lin et al., 2026); in its public implementation, an earlier artifact enters a generation when the agent passes that artifact's node to the generation call. In Design Anything, BUILD connects a selected concept, a specification, or a supplied asset to the downstream nodes before they run, so each generation inherits its references from the graph (Section 2.2). Over all 228 tasks, 95.3% of the image generations of Design Anything carry at least one reference, against 39.4% of JarvisHub's; on the 24-artifact tasks the shares are 97.9% and 40.2% (Appendix C.5).

Within JarvisHub, reference coverage tracks consistency. Comparing tasks within one harness removes the other differences between the two systems. Across JarvisHub's tasks, reference coverage correlates with identity consistency (Spearman $\rho = 0.40$, $p < 0.001$, 103 tasks) and with set consistency ($\rho = 0.26$, $p < 0.001$, 194 tasks), and the identity correlation holds within each task condition (pooled within-condition $\rho = 0.47$). Coverage does not correlate with constraint satisfaction or reference fidelity ($\rho \leq 0.10$), which compare outputs with the brief's text and with supplied assets rather than with earlier outputs. The relation is observational, and the component studies below do not remove the graph, so the consistency results remain evidence about the complete system.

3.4.4 What does autonomous continuation contribute to long-horizon runs?

Three arms run these 19 tasks on DeepSeek V4.1 Flash High (Table 2). *No continuation* removes the controller together with the contract, the planning tools, the automatic verifier, and every channel that restates the task or its progress, so a run ends when the model stops calling tools. *Generic continuation* keeps these removals but answers every stop with one fixed prompt to continue until the brief is fully delivered, naming no outstanding item. *Complete* is the full harness.

Without continuation, runs stop early. Delivery completion falls from 96.1% to 73.9% (lower on 8 tasks, higher on 3; two-sided Wilcoxon signed-rank test, $p = 0.018$), and six runs collapse against none ($p = 0.020$, Fisher exact test). Every collapsed run ended within three turns, most after asking the user to confirm a direction or announcing the next deliverables. Constraint satisfaction falls from 90.4% to 57.5% ($p = 0.001$).

A generic prompt restores delivery, at three times the cost. The fixed prompt brings delivery completion back to 96.9%, so for delivery it matters that a stop does not end the run, not which items the prompt names. It does so by regenerating delivered work: the generic arm issues 36.1 generation requests per run against 24.1 ($p = 0.002$) and, on the 17 tasks with full telemetry, takes 2.2 times the agent steps and 3.0 times the language-model cost ($p < 0.001$). Its constraint satisfaction (75.3% against 90.4%) is lower on 6 tasks and higher on 1, a difference not resolved at 19 tasks ($p = 0.063$). Because both ablated arms remove the contract, the outstanding set $\mathcal{R}$, the task-recall channels, and the verifier together, these effects are attributed to that set of mechanisms. Two smaller studies on separate task sets find that removing the canvas digest doubles the share of tool calls spent reading state, and that replacing the planning tools with a free-text checklist, or removing them, lowers delivery completion by 2.1 and 7.7 points on eight tasks, differences not resolved at that size (Appendix D).

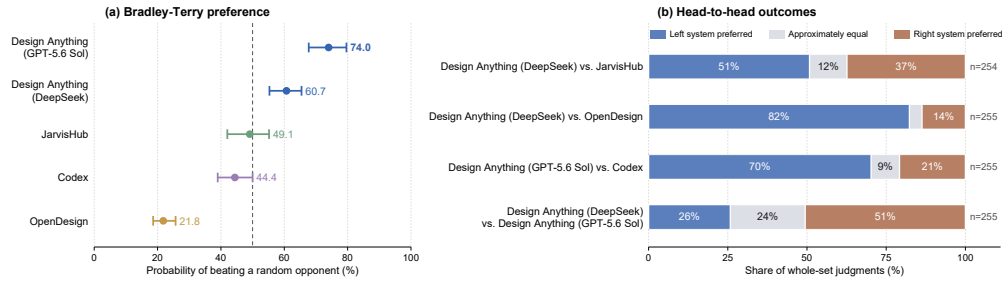


Figure 6: **Designer preference on whole sets.** Left: Bradley–Terry probability of beating a uniformly sampled other system, with 95% bootstrap intervals over tasks and designers; the dashed line marks 50%. Right: shares of whole-set judgments for each compared pair, where "left" is the first-named system; n is the number of judgments. DA denotes Design Anything.

3.5 Human evaluation

Nine designers from five disciplines compared outputs system-blind on 185 tasks of their own discipline (Appendix B.5). Each comparison shows two whole sets for the same brief, every delivered file in requirement order, from systems on the same language model or from Design Anything on the two models, which places both tracks on one scale; designers choose which set they would rather hand to the client, or approximately equal. We fit a Bradley–Terry model (Bradley & Terry, 1952), $\Pr(i \succ j) = e^{\beta_i}/(e^{\beta_i} + e^{\beta_j})$, to the 1,019 whole-set judgments, counting a tie as half a win for each side, and report the probability of beating a uniformly sampled other system with 95% intervals from 2,000 bootstrap replicates over tasks and designers.

Designers prefer the sets of Design Anything. Design Anything on GPT-5.6 Sol has the highest estimate (74.0%), followed by Design Anything on DeepSeek (60.7%), JarvisHub (49.1%), Codex (44.4%), and OpenDesign (21.8%); only the two configurations of Design Anything have intervals entirely above 50% (Figure 6). Head to head on the same language model, designers prefer the set of Design Anything in 82% of judgments against OpenDesign and 70% against Codex. The margin over JarvisHub is the smallest (51% against 37%), consistent with its higher per-image judge scores.

4 Conclusion

We presented Design Anything, a harness whose contract, workflow graph, and controller share deliverable identifiers so that completion is derived from project state rather than declared by the model, and DesignAnything-Bench, 228 project-level design tasks scored from exported files. With the language model fixed, Design Anything satisfies more brief constraints than each baseline at every project length, by 23–31 points on 24-artifact projects, where each fact must reach the most files. Limitations are discussed in Appendix F.

Reproducibility Statement

The task composition, evaluation views, and comparison protocol are described in Section 3.1 and Appendix B. Appendix C records the system configurations, including the common exclusion of skills written for particular design domains, and result coverage; Appendix A lists the run configuration of Design Anything, and Appendix D describes the component studies. The anonymous package at ** contains the benchmark tasks, the evaluator, per-task evaluation records, run manifests, and analysis scripts.

Ethics Statement

The benchmark's 1,086 distinct reference files were authored for the benchmark (667), generated with gpt-image-2.5 (375), or drawn from openly licensed sources (44: 31 public domain, 6 CC0, 5

CC BY-SA, and 2 under the Open Government Licence – Toronto); the corpus manifest records a license for every file and a credit for every external file, and the release redistributes each file under its recorded license. The expert evaluation involves **, with compensation and institutional review or exemption documented as **. Generated outputs may contain visual defects, unintended similarities to existing works, or culturally inappropriate content. The benchmark measures performance under its specified tasks and does not establish that generated artifacts are suitable for unrestricted professional deployment.

AI USE STATEMENT

Large language models were used in this work for implementing the harness and evaluation code, generating the synthetic reference materials that some tasks supply, producing figure-generation scripts, analyzing related work, drafting and editing the manuscript, and discussing the interpretation of results. They were not used to formulate or prove mathematical claims. All AI-assisted text was reviewed by the authors; every citation was checked against the cited source; every reported number was recomputed by the authors from the released per-task records; and every benchmark task was reviewed under the defect rule described in Section 3.1. The authors take full responsibility for the manuscript, the experimental claims, and the released materials.

REFERENCES

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Wang. Agent S: An open agentic framework that uses computers like a human. In *The Thirteenth International Conference on Learning Representations*, 2025.

Anthropic. Introducing Claude Design by Anthropic Labs, 2026. URL https://www.anthropic.com/news/claude-design-anthropic-labs. Published 2026-04-17. Accessed September 21, 2026.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

ByteDance Seed. Seedance 2.0 official launch, 2026. URL https://seed.bytedance.com/en/blog/seedance-2-0-official-launch. Published 2026-02-12. Accessed September 21, 2026.

Steven D. Eppinger, Daniel E. Whitney, Robert P. Smith, and David A. Gebala. A model-based method for organizing tasks in product development. *Research in Engineering Design*, 6(1): 1–13, 1994.

Jiaxin Ge, Zora Zhiruo Wang, Xuhui Zhou, Yi-Hao Peng, Sanjay Subramanian, Qinyue Tan, Maarten Sap, Alane Suhr, Daniel Fried, Graham Neubig, and Trevor Darrell. AutoPresent: Designing structured visuals from scratch. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2902–2911, 2025.

Kelly Hong, Anton Troynikov, and Jeff Huber. Context rot: How increasing input tokens impacts LLM performance. Technical report, Chroma, 2025. URL https://trychroma.com/research/context-rot.

Daeheon Jeong, Seoyeon Byun, Kihoon Son, Dae Hyun Kim, and Juho Kim. CANVAS: A benchmark for vision-language models on tool-based user interface design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Dayeon Ki, Tianyi Zhou, Marine Carpuat, Gang Wu, Puneet Mathur, and Viswanathan Swaminathan. GraphicBench: A planning benchmark for graphic design with language agents. *arXiv preprint arXiv:2504.11571*, 2025.

Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Harold Miles, Seraphina Nix, Tao Lin, Chris Painter, Neev

Parikh, David Rein, Lucas Jun Koba Sato, Hjalmar Wijk, Daniel M. Ziegler, Elizabeth Barnes, and Lawrence Chan. Measuring AI ability to complete long software tasks. In *Advances in Neural Information Processing Systems*, 2025.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *International Conference on Learning Representations*, 2026.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Niantong Li, Guangzheng Hu, Weixu Qiao, Ying Ba, Qichen Hong, Shijun Shen, Jinlin Wang, Fan Zhou, Jianye Kang, Xin Shang, Ziyi He, Wei Wang, Dalin Li, Jiahao Li, Jie Zhang, Kaiyuan Gao, Kun Yan, Lihan Jiang, Ningyuan Tang, Shengming Yin, Tianhe Wu, Xiao Xu, Xiaoyue Chen, Yuxiang Chen, Yan Shu, Yanran Zhang, Yilei Chen, Yixian Xu, Zekai Zhang, Zhendong Wang, Zihao Liu, Zikai Zhou, Hongzhu Shi, Yi Wang, Bing Zhao, Hu Wei, Lin Qu, and Chenfei Wu. Qwen-Image-Bench: From generation to creation in text-to-image evaluation. *arXiv preprint arXiv:2605.28091*, 2026a.

Wanli Li, Bowen Zhou, Yunyao Yu, Zhou Xu, Yifan Yang, Dongsheng Li, and Caihua Shan. WeaveBench: A long-horizon, real-world benchmark for computer-use agents with hybrid interfaces. *arXiv preprint arXiv:2606.09426*, 2026b.

Yunlong Lin, Zixu Lin, Zhaohu Xing, Biqiang Li, Chenxin Li, Haonan Wang, Haitao Wu, Hengyu Liu, Jianghai Chen, Kaituo Feng, Kaixin Li, Shawn Chen, Shijue Huang, Sixiang Chen, Tsung-Yi Ho, Wenxuan Huang, Xiangyan Liu, Xiaomeng Hu, Xuanhua He, Yan Sun, Yunqing Zhao, Zhiqin Yang, Zehan Wang, Zhengyang Tang, Tianyu Pang, and Xiangyu Yue. JarvisHub: An open harness for canvas-native multimodal creative agents. *arXiv preprint arXiv:2607.23588*, 2026.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024b.

Yaxin Luo, Haobin Jiang, Jialv Zou, Xu Huang, Wenhao Yan, Haodong Li, Zhengrong Yue, Jing Li, Xiaofu Chen, Xiaohan Zhao, Jiacheng Liu, Jiacheng Cui, Zhiqiang Shen, and Xiaotong Li. AutoDesign: Meta-harness optimization for long-horizon agentic design. *arXiv preprint arXiv:2608.13560*, 2026.

Ziyu Ma, Hailang Huang, Shun Zou, Yong Wang, Shidong Yang, Yiming Hu, Fei Wei, and XiangXiang Chu. LongHorizon-Harness: Advancing long-horizon agents for real-world tasks. *arXiv preprint arXiv:2608.01964*, 2026.

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, Junhong Shen, Guanghao Ye, Haowei Lin, Jason Poulos, Maoyu Wang, Marianna Nezhurina, Jenia Jitsev, Di Lu, Orfeas Menis Mastromichalakis, Zhiwei Xu, Zizhao Chen, Yue Liu, Robert Zhang, Leon Liangyu Chen, Anurag Kashyap, Jan-Lucas Uslu, Jeffrey Li, Jianbo Wu, Minghao Yan, Song Bian, Vedang Sharma, Ke Sun, Steven Dillmann, Akshay Anand, Andrew Lanpouthakoun, Bardia Koopah, Changran Hu, Etash Guha, Gabriel H. S. Dreiman, Jiacheng Zhu, Karl Krauth, Li Zhong, Niklas Muennighoff, Robert Amanfu, Shangyin Tan, Shreyas Pimpalgaonkar, Tushar Aggarwal, Xiangning Lin, Xin Lan, Xuandong Zhao, Yiqing Liang, Yuanli Wang, Zilong Wang, Changzhi Zhou, David Heineman, Hange Liu, Harsh Trivedi, John Yang, Junhong Lin, Manish Shetty, Michael Yang, Nabil Omi, Negin Raoof, Shanda Li, Terry Yue Zhuo, Wuwei Lin, Yiwei Dai, Yuxin Wang, Wenhao Chai, Shang Zhou, Dariush Wahdany, Ziyu She, Jiaming Hu, Zhikang Dong, Yuxuan Zhu, Sasha Cui, Ahson Saiyed, Arinbjörn Kolbeinsson, Jesse Hu, Christopher Michael Rytting, Ryan Marten, Yixin Wang, Alex Dimakis, Andy Konwinski, and Ludwig Schmidt. Terminal-Bench:

Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, 2026.

Open Design contributors. OpenDesign: The open-source Claude Design alternative, 2026. URL https://github.com/nexu-io/open-design. GitHub repository. Accessed September 21, 2026.

OpenAI. Introducing ChatGPT Images 2.0, 2026. URL https://openai.com/index/introducing-chatgpt-images-2-0/. Published 2026-04-21. Accessed September 21, 2026.

Wei Pang, Kevin Qinghong Lin, Xiangru Jian, Xi He, and Philip Torr. Paper2Poster: Towards multimodal poster automation from scientific papers. In *Advances in Neural Information Processing Systems*, 2025.

Zhe Ren, Yimeng Chen, Dandan Guo, Guowei Rong, Tonghui Li, R. B. Xiong, Qingfeng Lan, Wenyi Wang, Nanbo Li, Yibo Yang, Mingchen Zhuge, and Jürgen Schmidhuber. Self-improvements in modern agentic systems: A survey. *arXiv preprint arXiv:2607.13104*, 2026.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, Weishu Zhang, Tyler Zeng, Ying Yan, Bo Liu, Hanson Wen, Mingyang Xu, Xiaoyuan Liu, Zimeng Chen, Weiyan Shi, Amanda Dsouza, Vincent Sunn Chen, Patrick Bryant, Carl Boettiger, Yamini Rangan, Bradley Rothenberg, Kyle Steinfeld, Arvind Rao, Tapio Schneider, Georgios Yannakakis, Laure Zanna, Kaan Ozbay, Ida Sim, Tarek Zohdi, George Em Karniadakis, Jack Gallant, Teresa Head-Gordon, Yushan Li, Wenxi Deng, Tao Sun, Huiqi Wang, Zhun Wang, Justin Xu, Chris Yuhao Liu, Yafei Cheng, Rongwang Hu, Aras Bacho, Shengcao Cao, Zengyi Qin, Yixiong Chen, Hengduan Fan, Hao Liu, Lin Zeng, Shashank Muralidhar Bharadwaj, Litian Gong, Yingxuan Yang, Maojia Song, Ruheng Wang, Zongzheng Zhang, Honglin Bao, Shuo Lu, Jianhong Tu, Zhonghua Wang, Zheng Zhang, Zijiao Chen, Yanqiong Jiang, Zhendong Li, Bohan Lyu, Chang Ma, Peiran Xu, Benran Zhang, Shangding Gu, Haoyue Hua, Haoyang Li, Wanzhe Liao, Chengzhi Liu, Junbo Peng, Haoran Sun, Zechen Xu, Bo Chen, Jiayi Cheng, Yi Jiang, Keying Kuang, Yuan Li, Youbang Pan, Ziyan Rao, Alexander Schubert, Yifan Shen, Vincent Siu, Xiatao Sun, Kangqi Zhang, Xiaopan Zhang, Yuchen Zhu, Ishaan Singh Chandok, Lei Ding, Jingxuan Fan, Andrew Glover, Jiaming Hu, Yiran Hu, Wenbo Huang, Zixin Jiang, Haoran Jin, Lukas Kim, Ming Liu, Yang Liu, Alireza Rafiei, Xuhuan Shen, Kunyang Sun, Sophia Sun, Ting Sun, Eric Wang, Yixin Wang, Hanwen Xing, Sihan Xu, Yuzheng Xu, Zhongxing Xu, Zhiling Yan, Boqin Yuan, Ruiqi Zhang, Yifan Zhang, Zibo Zhao, Liana, Santanu Bosu Antu, Haoyue Bai, Carlo Bosio, Joseph Cavanagh, Patricia Cavazos-Rehg, Tianxing Chen, Xuewen Chen, Yipu Chen, Chenyu Zhu, Chen Dai, Stefano De Castro, Yunfu Deng, Kaustubh Dhole, Jiayuan Ding, Chenchen Du, Zhehang Du, Hao Fan, Run-Ze Fan, Hengyu Fu, Shi Gu, Yifan Gu, Charlie Guo, Baihe Huang, Baixiang Huang, Rimika Jaiswal, Zhihan Jiang, Ran Jin, Erin Kasson, Xin Lan, Joseph Lee, Deren Lei, Chenyu Li, Daofeng Li, Haitao Li, Hongwei Li, Jingyan Li, Xiao Li, Yi Li, Yinsheng Li, Yuangang Li, Zhixu Li, Wenyu Liang, Longtai Liao, Kevin Qinghong Lin, Andy Zeyi Liu, Che Liu, Jiaming Liu, Kaiyuan Liu, Xuan Liu, Pan Lu, Wenbo Lv, Yicheng Lyu, Qiuyang Mang, Kyle Montgomery, Yuzhou Nie, Ruoxi Ning, Jorin Overwiening, Xu Pan, Layna Paraboschi, Core Francisco Park, Justin Purnomo, Swati Rajwal, Scott Rankin, Bixuan Ren, Yiren Rong, HaoYang Shang, Ventus Shaw, Fiona Shen, Jiawei Shen, Minqi Shi, Shi Qiu, Huaxiu Yao, Tianneng Shi, Jonah So, Vladislav Susoy, Hannah Szlyk, Haocheng Wang, Jialu Wang, Wei Wang, Xinyu Wang, Zehao Wang, Dowling Wong, Angela Wu, Dehao Wu, Fangyu Wu, Mengyuan "Millie" Wu, Yu Wu, Yuchen Wu, Yuhao Wu, Qingpo Wuwu, Weihang Xiao, Yongyi Xiong, Fan Xu, Ruiling Xu, Mingxuan Yan, Benjamin Yang, Jirong Yang, Sen Yang, Xiaoli Yang, Yushi Yang, Haoran Ye, Xiaohu Yu, Zhengming Yu, Chenlong Zhang, Chi Zhang, Hanning Zhang, Hanwen Zhang, Junge Zhang, Kunpeng Zhang, Song Zhang, Wenjin Zhang, Wenshuo Zhang, Ying Zhang, Yizhi Zhang, Brian Zhao, Qijian Zhao, Yimin Zhao, Yuhaohua Zheng, Liwei Zhou, Tianyue Zhou, Sichen Zhu, Siqi Zhu, Yan Zhu, Yishu Zhu, Jierui Zuo, Chonghao Cai, Helena Casademunt, Wenjia Chen, Cheng Cheng, Nawen Deng, Rao Fu, Tianfu Fu, Yifan Han, He Ren, Zhenyu He, Qiao Jin, Langlang Li, Yuetai Li, Sylvia Liu, Lu Lu,

Luqing Zhou, Subhabrata Mukherjee, Yunqi Ouyang, Yin Ren, Dawei Shi, Haoran Wu, Zhiyue Wu, Hannah Yao, Zhuoran Yi, Jenny Yu, Rhea Zhan, Hang Zhou, Blake Zhu, Junfan Zhu, Alan Yuille, Yang Liu, Russell Alan Poldrack, Jiachen Li, Zhenglu Li, Molei Tao, Jing Huang, Wenqi Shi, Costas Spanos, Lichao Sun, Chenguang Wang, Orson Xu, Zhen Dong, Hector Gomez, Aylin Caliskan, Ali Emami, Haimin Hu, Zhi Li, Lihui Liu, Murphy Niu, Yi Shao, Jianxin Sun, Mikko Tolonen, Ting Wang, Sanjiv Das, Yanjun Gao, Wenbo Guo, Erika J Schneider, Zhiyong Lu, Yian Ma, Mark Mueller, Radha Poovendran, Somayeh Sojoudi, Yinglun Zhu, and Dawn Song. Agents' Last Exam. *arXiv preprint arXiv:2606.05405*, 2026.

Tripo. Tripo OpenAPI documentation: Introduction, 2026. URL `https://docs.tripo3d.ai/get-started/introduction.html`. Accessed September 21, 2026.

Tianfu Wang, Zhezheng Hao, Xilin Xia, Lixin Liu, Mengkang Hu, Hongzhang Liu, Xi Chen, Ziyan Liu, Xiankun Lin, Weijia Zhang, Nicholas Jing Yuan, and Hui Xiong. Agentic artifact creation: Systems, evaluation, principles, and opportunities. *arXiv preprint arXiv:2608.28122*, 2026.

Hu Wei. From agent loops to structured graphs: A scheduler-theoretic framework for LLM agent execution. *arXiv preprint arXiv:2604.11378*, 2026.

Yuhao Wu, Jingyuan Zhang, Jiajun Shi, Xinping Lei, Qingshui Gu, Yuxuan Zhang, Zexuan Wang, Chen He, Chen Huang, Maojia Song, Zhiyuan Zeng, Shaowen Wang, Jinkai Liu, Yunfeng Shi, Jiaheng Liu, Shen Yan, Wenhao Huang, Ge Zhang, and Wenxuan Zhang. HarnessDev: Can LLMs create and evolve their own agent harness? *arXiv preprint arXiv:2609.01437*, 2026.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024.

Xiangyuan Xue, Zeyu Lu, Di Huang, Zidong Wang, Wanli Ouyang, and Lei Bai. ComfyBench: Benchmarking LLM-based agents in ComfyUI for autonomously designing collaborative AI systems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24614–24624, 2025.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Mengqi Yuan, Zilong Zhou, Xinzhuang Xiong, Weiming Wu, Jiayang Sun, Jiamin Song, Kaiqian Cui, Bowen Wang, Haoyuan Wu, Yitong Li, Dunjie Lu, Haikong Lu, Qi Zhen, Xinyuan Wang, Jiaqi Deng, Yuhao Yang, Cheng Chen, Boyuan Zheng, Alex Su, Xiao Yu, Hao Zou, Saaket Agashe, Xing Han Lu, Manpreet Kaur, Zhengyang Qi, Vincent Sunn Chen, Frederic Sala, Dayiheng Liu, Junyang Lin, Zhou Yu, Yu Su, Siva Reddy, Xin Eric Wang, Peng Qi, Tianbao Xie, and Tao Yu. OSWorld 2.0: Benchmarking computer use agents on long-horizon real-world tasks. *arXiv preprint arXiv:2606.29537*, 2026.

Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. PPTAgent: Generating and evaluating presentations beyond text-to-slides. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 14402–14418. Association for Computational Linguistics, 2025.

Mingchen Zhuge, Changsheng Zhao, Dylan R. Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-Judge: Evaluate agents with agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80569–80611. PMLR, 2025.

A Implementation and Runtime Details

A.1 Task Contract

The task contract is stored as a structured project object visible on the canvas. It holds the final deliverable description, stable deliverable identifiers, public acceptance criteria, constraints, decisions marked as coming from the user or the agent, and open questions. It holds no execution phase or completion counter, so revising it updates intent rather than recording progress.

A.2 Graph Construction and Artifact Ownership

Graph-building operations accept typed node descriptions, configurations, connections, and a destination delivery group. They can create new nodes or associate eligible existing nodes with that group. Groups explicitly declare which contract deliverables they implement; an empty declaration denotes supporting work. Shared groups require output ownership information so that an artifact for one requirement is not automatically counted toward another. Dependency declarations and port-level edges serve different purposes: the former express ordering between groups, while the latter carry concrete inputs to operations.

A.3 Operational Delivery Checks

For each contract item, the delivery projection finds its implementation through explicit group claims and node ownership, and examines the status of the bound executable nodes; data-only inputs and annotations do not count as artifact-generating execution. In groups shared by several deliverables, an execution failure that cannot be attributed to one item remains a group-level failure signal rather than being assigned to an item.

Output-form checks use node capability declarations. An image-producing operation cannot satisfy a readable-document requirement merely because its image resembles a document, and a minimum-page declaration requires an implementation capable of paginated output without verifying the realized page count. Quantity checks count supported output items rather than generation nodes. The benchmark's artifact evaluation is therefore necessary even when the runtime reports settlement.

A.4 Observation and Continuation

The canvas digest is a compact projection of workflow state regenerated between turns; detailed read tools expose individual nodes, connections, configurations, and outputs.

After a turn settles, an assembler reads the authoritative workflow view and relevant execution records. It constructs obligations for the root goal, deliverables, decisions, required questions, and execution. Obligations use the states open, waiting, in progress, satisfied, failed, and cancelled, together with evidence references and versions. A pure reducer then selects the next control action from this snapshot. Its principal sequence is:

1. Apply terminal cancellation, access, execution-failure, and circuit-breaker conditions.
2. Wait if a turn, job, or delegated execution is still active; otherwise honor pending decision handoffs.
3. Terminate unsuccessfully on failed obligations, or successfully when the root obligation is satisfied and no unresolved obligation remains. The root obligation is derived, not declared: it is satisfied only when every deliverable is delivered and no required question, pending decision, or running execution remains.
4. For unresolved work, inspect progress and resource guards; either schedule another model turn or terminate with an explicit stopping reason.

The continuation action is `dispatch_turn`. Its message names the remaining requirements and, for each, the blocking reason, such as a missing output form or a quantity shortfall. The model then decides whether to read state, modify a contract, build or edit a graph, execute an operation, delegate work, or request a decision.

Progress monitoring separates obligation changes from observable workflow changes. Repeatedly restating a plan is not sufficient evidence of artifact progress. Content summaries cover artifact updates that do not change graph topology, including incremental document construction. In the code snapshots used for the reported runs, the obligation-stagnation and no-output guards stop a run after two and three eligible dispatch decisions without progress, respectively, with active execution and decision waits handled separately; a run may chain at most $\min(20, \max(2, 3n))$ consecutive dispatches for n obligations, and the benchmark driver additionally caps a run at 100 turns.

A.5 Automatic Verifier

An automatic verifier supplies background structural review once the workflow graph is complete; it receives concrete findings and acceptance criteria and returns them through the agent's feedback channel. Visual design critique is available only as a tool the agent may call. Goal completion remains governed by the control layer. The verifier is on in the main comparison and off in the component-study arms (Appendix D).

A.6 Run Configuration

The following settings apply to both Design Anything configurations unless noted.

Setting	Recorded configuration
Implementation snapshot	DeepSeek V4.1 Flash runs: five snapshots of one development line with no controller changes between them. GPT-5.6 Sol runs: later snapshots of the same line; besides the rate-limit waits and transport retries listed under Controller limits, they include changes to the image-analysis and image-repair code. Commit identifiers are recorded in each run manifest.
Agent and role models	Every role (orchestrator, builder, verifier, summarizer, and specialist helpers), text-generation nodes, and image analysis use the configuration's language model. DeepSeek V4.1 Flash runs at its default high reasoning, with reasoning disabled for image analysis; GPT-5.6 Sol receives reasoning effort high for the main agent and specialist helpers.
Generation backends	Images: gpt-image-2.5-sunburst, medium quality, 2K. Video: MiniMax-H3 (DeepSeek V4.1 Flash runs) and Seedance 2.0 (GPT-5.6 Sol runs).
Skills	General-purpose skills bundled with the product; none written for a particular design domain
Other tools	Workspace reading and editing, contract and workflow construction, synchronous and batched execution, on-demand image analysis, clarifying questions, web search, memory, and delegation to helper agents
Context policy	Tool results are elided beyond 50% of the context window (1M tokens for DeepSeek V4.1 Flash, 272k for GPT-5.6 Sol); history is summarized beyond 85%, keeping the last six steps; the canvas digest is always on
Resource limits	Per turn: 120 tool calls, 120 model steps, 90 minutes. Per run: 100 turns. Per task: 150 minutes wall clock (DeepSeek V4.1 Flash) or 150–240 minutes (GPT-5.6 Sol). Tokens are recorded but not limited.
Checkpoint policy	A scripted user approves approval and review checkpoints, picks the recommended option (otherwise the first option that does not defer), and answers free-text questions with a fixed message stating that the brief is complete; a repeated question fails the run
Automatic verifier	On in the main comparison; off in ablation arms
Controller limits	Two-decision obligation-stagnation guard; three-decision no-output guard; at most $\min(20, \max(2, 3n))$ consecutive dispatches for n obligations; 100 turns per run. GPT-5.6 Sol snapshots also wait up to three times on rate limits, and the later ones re-dispatch up to twice after a transport interruption.

B BENCHMARK CONSTRUCTION AND EVALUATION PROTOCOL

B.1 CORPUS COMPOSITION AND EVALUATION SPECIFICATIONS

The 19 domains are product design; vehicle styling; furniture and home products; architecture; interior design; exhibition design; landscape design; fashion and textiles; jewelry and accessories; brand identity; graphic design and advertising; books and illustration; packaging; UI/UX; information design and data visualization; animation and storyboarding; game art and worldbuilding; e-commerce product imagery; and advertising and fashion photography. Each contributes two base project families evaluated at three scales, three independent reference-material projects, and three independent breadth projects. The two families in a domain differ in decision shape: one asks the system to propose alternatives and select one, and the other fixes the direction in the brief. This yields $19 \times (2 \times 3 + 3 + 3) = 228$ tasks and $19 \times (2 + 3 + 3) = 152$ project identities. The 114 base tasks support within-family comparisons; the other 114 tasks broaden the input conditions and project coverage.

Each evaluation specification translates requirements from the public brief into scoring criteria. Output slots describe artifact categories and expected quantities, mark core requirements, and distinguish process from final deliverables. Additional rules specify exact factual requirements, source and target artifacts for comparisons, and when a criterion is inapplicable. For a factual constraint, whitespace and dash characters are removed and full-width forms are normalized before matching, and the constraint targets the delivered files of the categories it names, capped at the requested quantity, or only each category's final file when the specification says so. A constraint none of whose target categories contains a delivered file counts as unsatisfied and stays in the denominator; if only some are empty, it is checked on the delivered files alone. Files whose transcription failed are dropped from the target set, and the constraint is excluded when no targeted file remains, or when its target categories are empty while a delivered file of a type they accept received no valid category from the classifier. Every evaluated requirement must be supported by the public brief or the supplied materials. Internal evaluation labels are not additional instructions to the tested agent. English is the common input language; Chinese review translations are authoring aids and are not delivered to systems.

Benchmark independence. Tasks were authored with LLM assistance from a per-domain topic plan and reviewed by the authors, with English scenarios as the only system input. No internal label of Design Anything enters scoring. During development, Design Anything was run on a separate 20-task pilot corpus, on 5-task smoke subsets of the final corpus, and on 13 base-scale candidate tasks, to fix infrastructure failures. No brief was revised in response to any system's score except under a recorded defect rule applied identically to all systems. Each system kept whatever general-purpose skills its product bundles; none used skills written for a particular design domain, which could supply prebuilt design procedures, and none received task-specific hints.

Specifications are frozen per corpus version, and execution and scoring use the same frozen specifications; this snapshot uses v0.6 with the task-set hash in Appendix C.

B.2 A COMMON INTERFACE FOR HETEROGENEOUS OUTPUTS

The evaluator consumes a package of delivered files and evaluation specifications, independent of the system's internal representation. It can therefore compare a canvas-based workflow with a file-based agent without requiring the latter to reconstruct a canvas. Import records distinguish supplied inputs, primary deliverables, and derived representations. Multiple file formats of the same work are associated to avoid counting a preview and its source as two deliveries. Input materials copied into an output directory remain inputs, identified by their content.

HTML decks are rendered by their own page structure, and PDF and presentation files keep their page and text information; a file whose pages cannot be read makes a page requirement unmeasurable rather than failed. Rendered deck pages can be inspected for consistency but do not fill standalone artifact requirements.

For image-slot assignment, the classifier scores each candidate slot and an unassigned option. A capacity-constrained assignment respects each slot's expected quantity and allows each candidate

to occupy at most one capacity slot. Remaining candidates may be associated as overflow without increasing completion beyond the requirement. This matching uses the artifact and the specification, not Design Anything node labels, group names, or execution order. The snapshot uses the classifier identifier `deepseek-flash` and evaluator v1.18. Prompt fingerprints are retained in the scoring records; the identifier is an undated alias, and the model string returned by the server was not logged. Ambiguous residual assignments remain visible as measurement limitations.

B.3 MEASURES

Table 3 summarizes the measures of Section 3.1. For requested category i with expected and delivered quantities e_i and d_i, delivery completion is $\mathrm{DC} = \sum_i \min(d_i, e_i) / \sum_i e_i$; the cap keeps excess outputs in one category from offsetting a shortfall in another. A constraint's fact matches only verbatim with matching case, after whitespace and dash normalization; most targets are generated images (85–89% of target files for every system). Reference fidelity and identity consistency differ only in the second member of each judged pair: a supplied reference asset or the run's designated anchor artifact. Delivery completion, constraint satisfaction, reference fidelity, and identity consistency pool their counting units over the 228 tasks; set consistency averages within each task and weights tasks by their numbers of scored items, and the per-image scores average within each task and then over tasks (Appendix C). Pairs the judge cannot compare count as mismatches; Appendix C.3 reports both pair measures with them excluded. Evaluation specifications derive only from the public brief (Appendix B.1).

Table 3 Measures of DesignAnything-Bench. Each output is scored against one reference: the brief, the client's materials, the run's own outputs, the image alone, or another system's outputs for the same brief.

Measure	Compared against	Definition	Scored by
Requirement fulfillment			
Delivery completion	Brief: categories, quantities	Requested files delivered, capped per category	Classifier, count
Constraint satisfaction	Brief: factual constraints	Constraints whose fact appears verbatim in every targeted file (Eq. 6)	Transcription, match
Reference fidelity	Client's reference assets	Output–reference pairs depicting the same object	Calibrated judge
Cross-artifact consistency			
Identity consistency	The run's anchor artifact	Output–anchor pairs depicting the same object	Calibrated judge
Set consistency	The run's whole set	Rubric score (0–100) for shared visual rules	Calibrated judge
Per-image quality			
Quality, aesthetics, text, domain	The image alone	Checklist score (0–100) per criterion (Li et al., 2026a)	VLM judge
Human evaluation			
Preference	Another system, same brief	Blinded pairwise choice; Bradley–Terry strength	Designers

B.4 COMPLETION, CONSISTENCY, AND ABSTENTION

Requirement fulfillment distinguishes an artifact's slot assignment, delivered quantity, and compliance with format or content requirements. In particular, page count does not determine whether a document belongs to a slot or whether it was delivered. A short presentation is therefore delivered, and its page minimum is not part of any measure in Table 3. Page minima enter only the task-level pass/fail outcomes in Table 10, which also require every requested category to be filled and every constraint marked critical to hold; an unreadable page count makes a task unmeasurable rather than failed. Actual measurable denominators, unmeasurable runs, and exclusion reasons accompany every aggregate.

Consistency comparisons are scoped to properties that the project is supposed to preserve. Alternative concepts need not look identical, and a legitimate refinement need not reproduce a sketch literally. The reference source determines the measure: agreement with a supplied reference asset is reference fidelity, a requirement-fulfillment measure, whereas agreement among outputs of the same run is identity consistency or set consistency. Missing source evidence does not establish a semantic mismatch. The snapshot distinguishes unavailable pair judgments from mismatches, but retains system-attributed unavailable pairs in the denominators of reference fidelity and identity consistency; unavailable pairs attributed to an external reference are excluded and recorded separately. Repeated failures of the same fact are not counted independently across dimensions simply because different tools can inspect it.

B.5 Quality assessment and human comparison

The quality protocol separates technical quality, aesthetics, text rendering, and domain-specific criteria. A criterion may be inapplicable, for example anatomical fidelity in an image without a person. The scores in Appendix C.6 use a checklist rubric with `deepseek-flash`, the undated alias also used for category assignment (Appendix B.2), as judge, thinking disabled. The judge sees each delivered image of a core image slot on its own and marks every checklist item 0, 1, 2, or not applicable. Item marks are averaged within each sub-criterion, the sub-criteria are averaged within each of the four groups, and the result is mapped to 0–100. A task's score for a group averages the images within each core slot and then averages over slots, so a slot that asks for many images does not outweigh the others. On pages that an importer rendered from HTML or rasterized from vector files, the three items for noise, edge clarity, and resolution are marked not applicable, because they measure our renderer rather than the design. An image whose judge call fails is left unscored and is not retried. This judge has not been calibrated against designers on the present artifacts; Appendix C.6 instead reports a second judge on a stratified sample.

Designer evaluation. The designer study (Section 3.5) uses the same run of each system that the automatic measures score, so no run is selected by appearance. Nine designers rate 185 tasks in five discipline groups: visual communication (three designers; brand identity, graphic design and advertising, packaging, books and illustration, information design, photography, and e-commerce imagery), digital interaction (two; UI/UX, game art, animation and storyboarding), product (two; product design, vehicle styling, furniture), spatial (one; architecture, interior, exhibition, and landscape design), and fashion and accessories (one). Within a group, tasks are stratified by domain, task set, and length and dealt to designers in rotation; each designer first rates three warm-up tasks, and some tasks are rated a second time by another designer of the group. Pairs are formed only within a language model: Design Anything, OpenDesign, and JarvisHub pairwise on DeepSeek, and Design Anything against Codex on GPT-5.6 Sol; Design Anything on DeepSeek against Design Anything on GPT-5.6 Sol is added for whole sets only, which connects the two tracks on one Bradley–Terry scale. Single-artifact comparisons use up to three final deliverables per task that both systems delivered, drawn from one seeded order per task so that the DeepSeek pairs share slots where possible; whole-set comparisons show every file the evaluator assigned to a requirement, in requirement order. The designers returned 2,405 single-artifact and 1,019 whole-set judgments on the main questions. Sides are randomized, and about 5% of each designer's single-artifact comparisons are repeated later with sides swapped. Designers see the brief with a reference translation, never system names, reasoning, or completion claims, and answer with left, right, or approximately equal: four questions for a single-artifact comparison (professional deliverability, domain correctness, fit to intent, originality), each with a confidence rating, and one question for a whole-set comparison (which set they would rather hand to the client). Single-artifact comparisons can also be flagged as mismatched when the two sides are not the same deliverable; flagged comparisons are excluded. Side-swapped repeats are held out of estimation and used only to measure test–retest agreement.

B.6 Judge calibration

Two professional designers independently labeled samples of each judged decision, drawn from the outputs of the three DeepSeek-backed systems. The first five items of each sample are warm-up items and do not enter the statistics. A judge qualifies when Cohen's $\kappa \geq 0.60$ and raw agreement is at least 85%. Human-judge agreement is computed on the items where both designers agree.

Identity judge. A single judge decides whether two images depict the same object, character, logo, or space; it serves both reference fidelity and identity consistency. Of 108 sampled pairs, 103 are analyzed. Human-human agreement is $\kappa = 0.931$ with 96.1% raw agreement. On the 99 consensus pairs, human-judge agreement is $\kappa = 0.887$ with 93.9%. By reference source, κ is 0.932 for pairs within a run (identity consistency, $n = 56$) and 0.830 for pairs against supplied assets (reference fidelity, $n = 43$). By subject, κ is 1.000 for characters, 1.000 for logos, 0.805 for products, and 0.826 for spaces.

Set-consistency judge. Of 117 sampled rubric items labeled consistency, 112 are analyzed. Human-human agreement is $\kappa = 0.893$ with 92.9%. On the 104 consensus items, human-judge agreement is $\kappa = 0.885$ with 92.3%. The judge answered not applicable on 14 of the 117 items. On the 100 analyzed items that remain after removing them, human-judge κ is 0.804 and 0.836 for the two designers and human-human κ is 0.901, so the qualification does not rest on not-applicable answers.

Uncalibrated steps. Two model-dependent steps have no human calibration in this snapshot. The classifier that assigns each delivered artifact to a requirement category, on which delivery completion depends, was checked once on 27 images under an earlier prompt whose labels were not retained; that reading is not comparable to the current classifier and is not used. The transcription of text from delivered artifacts, on which constraint satisfaction depends, has not been measured against human transcription. Both are reported as measurement limitations.

Direction of judge error. Agreement statistics do not show whether judge errors favor a particular system, so we also compare the judge with the designers per system on the consensus items. The identity judge labels more pairs as the same subject than the designers do: by 2.9 points for Design Anything, 2.9 for OpenDesign, and 14.3 for JarvisHub. On the 0–100 scale, the set-consistency judge scores Design Anything 9.2 points below the designers, while it scores JarvisHub 1.6 and OpenDesign 2.0 points above them. Neither judge's error favors Design Anything.

Scope. The calibration samples come from the 20-task pilot corpus scored with rubric pack v0.1.3. The judge prompts are identical to those used for the 228-task snapshot, with the same prompt fingerprint. The v0.6 specifications define different anchors and comparison scopes per task, and transfer of the calibration to them has not been tested separately.

C System Configurations and Complete Results

C.1 Controlled comparisons and snapshot provenance

Table 4 identifies what each reported comparison holds fixed and what it varies. The two main tracks compare complete systems: holding the language model fixed does not hold media-generation tools, orchestration, or execution policies fixed, so their differences are attributed to the whole harness. Model names and reasoning settings come from the run-configuration report; exact archived system and API snapshots are recorded separately. Video is generated with MiniMax-H3 by Design Anything on DeepSeek and with Seedance 2.0 by Design Anything on GPT-5.6 Sol; the baselines use their own video tools. Codex ran with a 150-minute limit per task; all 228 runs ended normally within it (median 16 minutes).

The scoring snapshot uses corpus v0.6, evaluator v1.18, rubric `deterministic-2026-08-23.2`, and classifier identifier `deepseek-flash` with reasoning disabled; the judge and the category classifier thus use the same model that drives the three DeepSeek V4.1 Flash systems. The task-set SHA-256 is:

```
3e7c05f57e269c2964da67774e095d72fe0b
e0e93b1878f961d75650ec0fb6b9
```

The two displayed lines concatenate to one hash. The scoring code version is recorded in the release package; classifier sampling is recorded as model-managed. Prompt fingerprints are retained in the scoring records; `deepseek-flash` is an undated alias, and the model string returned by the server was not logged.

Table 4 Controlled comparison matrix. The main tracks compare complete systems; the other comparisons vary only the factor in the last column. All rows use the frozen v0.6 briefs and evaluator v1.18.

Comparison	Tasks	Held fixed	Varied factor
DeepSeek track (Table 1a)	228	DeepSeek V4.1 Flash High for every agent and sub-agent; gpt-image-2.5-sunburst image generation	Harness and agent: Design Anything, OpenDesign, JarvisHub
GPT track (Table 1b)	228	GPT-5.6 Sol High	Harness, agent, and image tool: Design Anything, Codex
Project length (Section 3.4.1)	38×3	Project family, background, and inputs; system	Requested artifacts: 6, 14, 24
Reference coverage (Section 3.4.3)	228	DeepSeek V4.1 Flash High; within JarvisHub, the harness	Reference coverage (observed, not assigned)
Continuation (Section 3.4.4)	19	Design Anything on DeepSeek; 24-artifact tasks, one per domain	Continuation and requirement channels
Planning interface (Appendix D)	8	Design Anything; other requirement channels	Planning tools: structured, text, none
Canvas digest (Appendix D)	10 pairs	Design Anything; direct read tools	Digest on or off
Human evaluation (Section 3.5)	185	Scored runs of each system	System, within a language model

Each configuration's final evaluations contain 228 distinct task identifiers, with 38 tasks at each base scale and 57 each for breadth and reference-material conditions, all with the same recorded evaluator version and corpus hash.

All five configurations are scored by our own pipeline under identical conditions, with one final result per task. For all systems, runs interrupted by infrastructure faults (API or transport errors, sandbox lockouts, sub-agent timeouts) were resumed or re-executed; each task contributes one final result. Evaluations that failed for evaluator-transport reasons were re-scored from the same exported outputs without rerunning the design tasks. A single comparison script, `compareArms`, computes Tables 1, 5, and 10 from each system's per-task `evaluation.json` records, which are archived and provided with the release package. OpenDesign and JarvisHub were scored on a separate machine with its own cache of judge responses; independent scorings of the same outputs differ by about three points on judged measures, so we do not interpret differences of that size. Evaluations under earlier evaluator versions are not merged into this snapshot.

C.2 Metric definitions, counts, and abstention

Table 5 preserves each measure's counting unit. Delivery completion pools fulfilled requirement slots, with quantities capped at their expected counts. Constraint satisfaction divides passed constraints by applicable constraints; a constraint passes only when its fact appears verbatim in every artifact it targets.

Unjudgeable pairs attributed to the produced evidence stay in the reference-fidelity and identity-consistency denominators, and those attributed to an external reference are excluded; Table 8 reports both measures with the judge's abstentions excluded.

Set-consistency aggregation. The reported set-consistency score uses only rubric items labeled consistency, excluding separate compliance items. Item scores of 0, 1, and 2 map to 0, 60, and 100; inapplicable items are excluded and unmet items contribute zero. Within a task, the evaluator averages rule-level consistency means equally. Across tasks, it weights these task scores by their numbers of scored consistency items. Table 1 uses this aggregation for all systems.

Task-level pass/fail outcomes. Table 10 also reports, for completeness, a task-level outcome that is not one of the measures in Table 3: a task passes when every requested category is filled, every

Table 5 Counts underlying Table 1. Columns count fulfilled/expected delivery slots (delivery completion), passed/applicable constraints (constraint satisfaction), same/eligible pairs against supplied reference assets (reference fidelity), same/eligible pairs within the run (identity consistency), and scored rubric items (set consistency). JarvisHub's reference-fidelity denominator is a range: 1,326 counts the judge's abstentions as mismatches and 1,127 excludes them. The last column is a coverage count, not a simple pooled-mean denominator.

System	Delivery	Constraints	Reference	Identity	Set items
Design Anything (DS)	2809/2908	767/902	1063/1407	554/656	606
OpenDesign (DS)	2768/2908	616/885	768/1396	467/621	585
JarvisHub (DS)	2547/2908	608/894	658/(1127–1326)	411/600	502
Design Anything (GPT)	2830/2908	834/902	1122/1414	598/663	614
Codex (GPT)	2853/2908	699/895	1016/1425	498/661	550

constraint the specification marks as critical holds, and the page minima of presentations and reports are met. The *all* column applies this test to every deliverable, including requested intermediate outputs, and the *final* column to final deliverables only. Pass fractions over 228 describe observed passes over all evaluated tasks, not success conditional on measurable evidence. Unmeasurable outcomes have four sources: a report or slide deck delivered in a format without a readable page count, so its page minimum cannot be checked, which is the most common source; an original file the evaluator cannot read; a run the harness recorded as an infrastructure failure; and a critical constraint that could not be judged. Because the evaluator fails a task on any unmet requirement, an unmeasurable task has no measured failure, and removing such tasks from the denominator would penalize the systems that have more of them; all counts therefore use 228 tasks.

C.3 Robustness of the paired comparisons

Table 6 decomposes constraint satisfaction. Each constraint targets one or more files, and it passes only if every target contains the fact; the per-file rate is the share of constraint–target pairs in which the file contains it. Longer projects spread each fact over more files, while each system's per-file rate stays roughly constant across lengths, so a fixed per-file gap compounds into a larger constraint gap. These readings are recomputed from the per-task records and reproduce the reported constraint rates within 0.1 points. Table 7 gives project-family-clustered bootstrap intervals for the paired gaps: the 152 distinct projects are resampled with replacement 4,000 times, the three scale variants of a family are drawn together, and intervals are 95% percentile intervals. Table 8 reports the two judged image-pair measures with and without the judge's abstentions; most abstentions concern pairs whose views are not comparable, such as a plan drawing against a photograph, or a whole object against a detail.

Table 9 asks whether constraint satisfaction counts missing deliveries a second time. A constraint none of whose target categories contains a delivered file counts as unsatisfied (Appendix B.1); excluding such constraints, 2 to 39 per configuration, changes the same-model gaps over all tasks by at most 1.9 points, and every paired gap at 14 and 24 artifacts still excludes zero. The growth of the gap from 6 to 24 artifacts becomes 10.0 points [−4.2, 24.3] against Codex and 18.9 [0.8, 37.6] against JarvisHub, so the growth against JarvisHub remains resolved under this reading and the growth against Codex does not. The growth against JarvisHub shrinks mainly because of its 12 such constraints at 24 artifacts, which narrow the gap there from 31.3 to 26.9 points.

Table 6 Where constraints are lost. Mean target files per constraint, per-file rate (%), and constraint satisfaction (%) at 6 / 14 / 24 artifacts.

System	Targets per constraint	Per-file rate	Constraint satisfaction
Design Anything (DS)	2.21 / 3.38 / 5.76	87.9 / 89.2 / 97.0	86.4 / 82.1 / 85.7
OpenDesign (DS)	2.32 / 3.42 / 5.84	74.7 / 76.4 / 75.3	70.6 / 58.7 / 60.6
JarvisHub (DS)	2.14 / 3.29 / 5.30	90.2 / 86.3 / 85.5	76.3 / 61.9 / 54.4
Design Anything (GPT)	2.21 / 3.47 / 5.68	99.1 / 98.3 / 97.9	97.4 / 94.2 / 91.9
Codex (GPT)	2.31 / 3.37 / 5.92	87.3 / 88.6 / 86.1	85.1 / 76.3 / 68.9

Table 7 Paired constraint-satisfaction gaps with family-clustered 95% bootstrap intervals (points). Growth is the gap at 24 artifacts minus the gap at 6.

Pair	All tasks	6 art.	14 art.	24 art.	Growth
Design Anything (DeepSeek) − JarvisHub	17.0 [12.4, 22.2]	10.0 [0.3, 19.6]	20.1 [7.2, 34.9]	31.3 [18.6, 45.8]	21.3 [3.8, 40.0]
Design Anything (DeepSeek) − OpenDesign	15.4 [9.4, 21.5]	15.7 [2.1, 30.8]	23.3 [9.9, 35.5]	25.1 [12.3, 38.4]	9.4 [−5.3, 24.0]
Design Anything (GPT-5.6 Sol) − Codex	14.4 [9.3, 19.3]	12.3 [6.0, 20.0]	17.9 [10.2, 26.4]	23.0 [9.0, 36.4]	10.6 [−3.5, 24.7]

Table 8 Judged image-pair measures with judge abstentions counted as mismatches (reported) and excluded. Abstention counts are attributed to the system's own evidence; for reference fidelity, abstentions attributed to the external reference are excluded in both columns.

	Identity consistency			Reference fidelity		
System	Abstained	Counted	Excluded	Abstained	Counted	Excluded
Design Anything (DS)	41/656	84.5	90.1	137/1407	75.6	83.7
OpenDesign (DS)	93/621	75.2	88.4	351/1396	55.0	73.5
JarvisHub (DS)	38/600	68.5	73.1	199/1326	49.6	58.4
Design Anything (GPT)	23/663	90.2	93.4	113/1414	79.3	86.2
Codex (GPT)	97/661	75.3	88.3	195/1425	71.3	82.6

C.4 Results by scale and domain

Table 10 reports every configuration under every condition. Each base scale contains the same 38 project families, so the three scale rows of a family are dependent observations, and the 6-to-14 contrast also changes prescribed process requirements. Set consistency per condition uses the same nested aggregation as Table 1, computed by the comparison script on the tasks of that condition.

Figure 7 reports constraint satisfaction for all 19 domains and all five configurations, without selecting favorable subsets. Each domain has 12 tasks and 25 to 80 applicable constraints, so per-domain differences are descriptive. With the language model fixed, Design Anything has the highest rate, alone or tied, in 18 of 19 domains on DeepSeek (JarvisHub leads in product design) and in 18 of 19 on GPT-5.6 Sol (Codex leads in exhibition design).

C.5 Reference coverage

Figure 8 reports how often an image generation received at least one reference input. For Design Anything, references are read from the generation requests in each run manifest; for JarvisHub, from the reference edges of its exported canvas graph. Over all 228 tasks, Design Anything issues 2,680 image generations, of which 95.3% carry a reference, and 4 tasks fall below one half; JarvisHub issues 4,258, of which 39.4% carry a reference, and 130 tasks fall below one half (three JarvisHub tasks produced no image). On the 24-artifact tasks, 74.9% of the referenced generations of Design Anything take an image generated earlier in the same run. The coverage of Design Anything is close to its ceiling in every condition, so its own per-task correlations are uninformative; we therefore relate coverage to outcomes within JarvisHub only.

Identity consistency per task is the pair-level rate of the run band, the same quantity that Table 1 pools. Tasks have a median of five identity pairs, so per-task values are coarse and 46 of the 103 tasks sit at 0 or 100%. The Spearman correlation with coverage is positive in every condition: 0.73 at 6 artifacts ($n = 17$), 0.53 at 14 ($n = 18$), 0.60 at 24 ($n = 16$), 0.37 on breadth tasks ($n = 32$), and 0.13 on reference-material tasks ($n = 20$); with ranks taken within each condition and pooled, $\rho = 0.47$ ($p < 0.001$). Correlations with set consistency, constraint satisfaction, and reference fidelity over all tasks are 0.26, 0.10, and 0.05.

Table 9 Constraint satisfaction (%) without constraints on wholly undelivered categories. Constraints none of whose target categories contains a delivered file are removed from numerator and denominator; parentheses give how many were removed. Reported rates, which count them as unsatisfied, are in Tables 1 and 10.

System	All tasks	6 art.	14 art.	24 art.
Design Anything (DS)	86.2 (12)	87.5 (2)	83.7 (3)	85.7 (0)
OpenDesign (DS)	70.3 (9)	70.6 (0)	61.1 (6)	61.0 (1)
JarvisHub (DS)	71.1 (39)	79.5 (6)	64.0 (5)	58.8 (12)
Design Anything (GPT)	92.7 (2)	98.0 (1)	94.2 (0)	91.9 (0)
Codex (GPT)	78.5 (5)	85.1 (0)	76.3 (0)	68.9 (0)

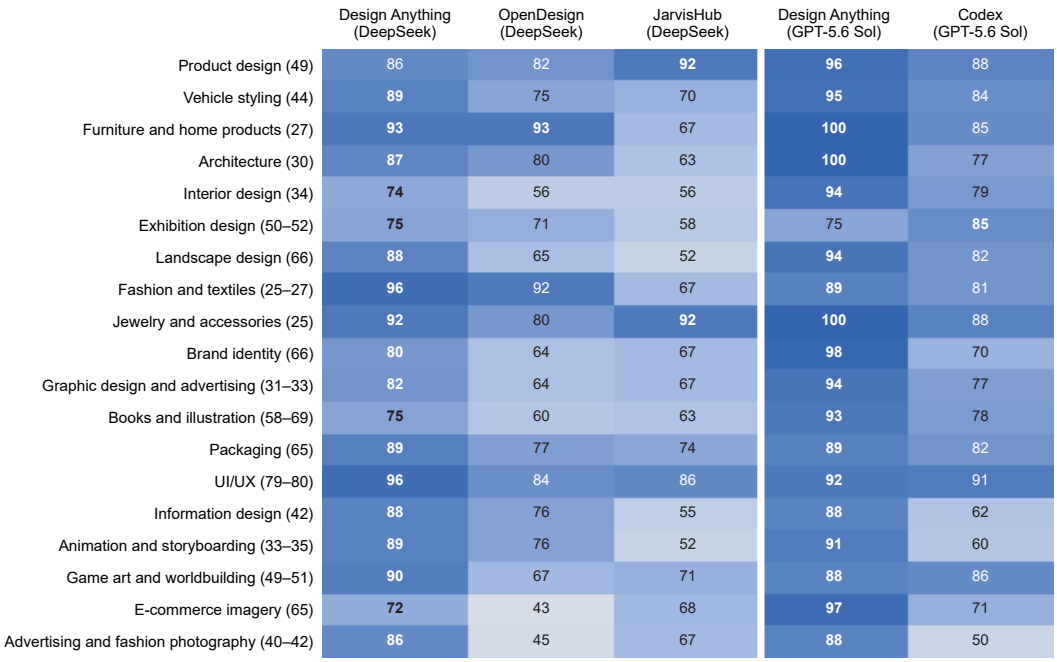


Figure 7: **Constraint satisfaction (%) by design domain.** Each cell pools the applicable constraints of the domain's 12 tasks; parentheses after a domain give its number of applicable constraints across systems. Bold marks the highest value within a language model; the white gap separates the DeepSeek and GPT-5.6 Sol tracks.

C.6 Quality assessment and human evaluation

Table 11 reports per-image quality with one judge for all configurations; Appendix B describes the executed protocol. All 228 tasks of every configuration were judged. The judged images per configuration are 2,089 for Design Anything (DS), 2,102 for Design Anything (GPT), 2,024 for OpenDesign, 1,920 for JarvisHub, and 2,126 for Codex, with 4, 2, 1, 3, and 3 images unscored after a failed judge call.

JarvisHub scores 0.4 to 1.4 points above Design Anything on DeepSeek on every score, and Codex scores 0.3 to 2.5 points below Design Anything on GPT-5.6 Sol; OpenDesign is lower on all four scores, most on text rendering. Figure 9 gives the paired differences with the project-family-clustered bootstrap of Appendix C.3; a paired mean can differ by 0.1 from the difference of the rounded means in Table 11. Over all tasks, the intervals exclude zero in both pairs only for aesthetics and domain criteria. Table 12 gives every score by length.

Length separates the pair on GPT-5.6 Sol. From 6 to 24 artifacts all four of Codex's scores fall with intervals excluding zero (domain from 94.5 to 89.2, aesthetics from 91.9 to 87.2), while those of Design Anything change by at most 1.6 points. The two are indistinguishable at 6 artifacts; at 24, Design Anything leads by 6.6 points on domain criteria (higher on 32 of 38 tasks, lower on

Table 10 Results by condition and system. Delivery completion, constraint satisfaction, and identity consistency are percentages; set consistency is the 0–100 nested aggregation; P/F/U counts tasks that pass, fail, or are unmeasurable under the task-level outcome defined above, over all deliverables and over final deliverables only. Shaded rows are Design Anything; bold marks the highest delivery, constraint, and identity value among all five rows of each condition. DS: DeepSeek V4.1 Flash High; GPT: GPT-5.6 Sol High.

System	Delivery	Constraints	Identity	Set	Task P/F/U (all)	Task P/F/U (final)
6 artifacts (38 tasks)						
Design Anything (DS)	96.5	86.4	87.7	93.6	27/10/1	27/10/1
OpenDesign (DS)	95.6	70.6	78.1	86.9	22/14/2	22/14/2
JarvisHub (DS)	93.4	76.3	76.9	75.0	21/15/2	21/15/2
Design Anything (GPT)	97.4	**97.4**	**97.0**	94.9	31/7/0	31/7/0
Codex (GPT)	**99.6**	85.1	93.9	92.1	32/6/0	32/6/0
14 artifacts (38 tasks)						
Design Anything (DS)	96.6	82.1	79.0	86.7	21/15/2	24/12/2
OpenDesign (DS)	90.4	58.7	61.0	82.7	12/23/3	12/23/3
JarvisHub (DS)	88.7	61.9	71.9	59.4	7/29/2	12/24/2
Design Anything (GPT)	96.8	**94.2**	**89.9**	82.7	20/17/1	25/12/1
Codex (GPT)	**97.9**	76.3	69.6	85.2	15/23/0	19/18/1
24 artifacts (38 tasks)						
Design Anything (DS)	96.5	85.7	86.3	87.0	14/24/0	15/23/0
OpenDesign (DS)	95.0	60.6	68.7	87.7	19/18/1	19/18/1
JarvisHub (DS)	81.8	54.4	60.3	61.0	4/33/1	7/30/1
Design Anything (GPT)	95.7	**91.9**	**87.8**	87.2	14/22/2	15/21/2
Codex (GPT)	**98.6**	68.9	66.7	91.1	19/18/1	20/17/1
Breadth (57 tasks)						
Design Anything (DS)	96.1	85.3	85.9	85.0	30/22/5	32/20/5
OpenDesign (DS)	97.5	73.4	80.1	91.7	33/22/2	33/22/2
JarvisHub (DS)	91.6	74.3	71.8	69.6	26/28/3	30/24/3
Design Anything (GPT)	**98.8**	**96.0**	**88.8**	88.3	41/15/1	41/15/1
Codex (GPT)	97.7	80.3	77.5	90.3	30/24/3	34/19/4
Ref. materials (57 tasks)						
Design Anything (DS)	97.3	**85.5**	81.1	89.7	31/23/3	31/23/3
OpenDesign (DS)	97.1	80.0	90.5	89.6	35/21/1	35/21/1
JarvisHub (DS)	88.8	70.2	71.4	63.9	24/32/1	24/32/1
Design Anything (GPT)	**98.6**	84.1	**94.0**	87.7	34/22/1	35/21/1
Codex (GPT)	97.5	79.1	84.8	89.1	29/25/3	33/20/4
All (228 tasks)						
Design Anything (DS)	96.6	85.0	84.5	87.9	123/94/11	129/88/11
OpenDesign (DS)	95.2	69.6	75.2	88.3	121/98/9	121/98/9
JarvisHub (DS)	87.6	68.0	68.5	65.4	82/137/9	94/125/9
Design Anything (GPT)	97.3	**92.5**	**90.2**	87.7	140/83/5	147/76/5
Codex (GPT)	**98.1**	78.1	75.3	89.8	125/96/7	138/80/10

3) and by 3.7 on aesthetics, and the gap grows from 6 to 24 artifacts on aesthetics, text rendering, and domain criteria. On DeepSeek, length does not separate Design Anything from JarvisHub: no difference between them has a growth interval excluding zero, and at 24 artifacts JarvisHub still leads on aesthetics by 1.5 points ($[0.3, 2.7]$). Among the DeepSeek configurations, only JarvisHub's domain score falls with an interval excluding zero, by 2.2 points ($[-3.8, -0.6]$), all of it between 6 and 14 artifacts, while its constraint satisfaction falls by 21.9 points (Table 1).

The judge places most system means above 90, so we checked whether the ranking depends on it. A stricter judge, Claude Opus 4.8, rescored a stratified sample of 15 tasks (three per condition) for Design Anything on GPT-5.6 Sol, OpenDesign, and JarvisHub, using the same delivered images; Codex is not in the sample. It scores lower and spreads the systems further apart: OpenDesign's domain score falls from 91.0 to 68.6, while those of Design Anything and JarvisHub are unchanged.

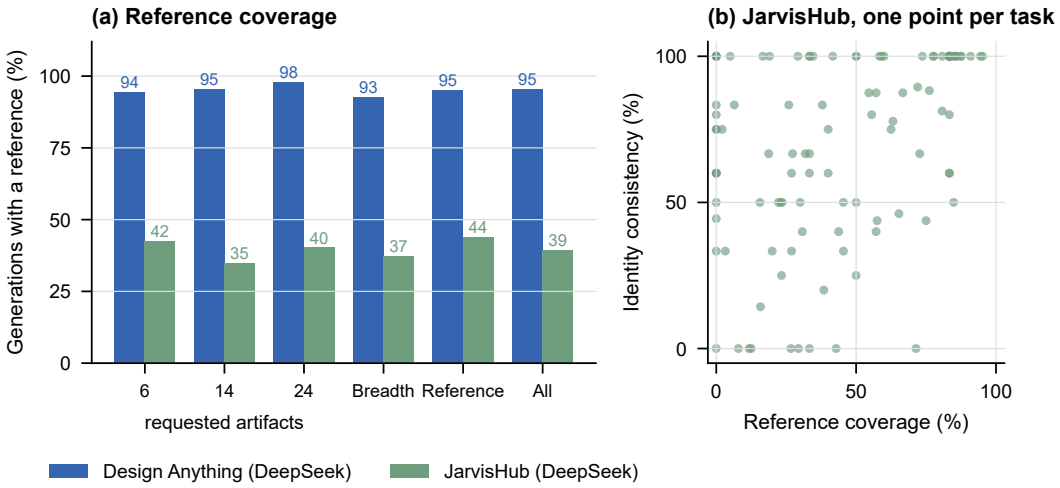


Figure 8: **Reference inputs to image generation.** (a) Share of image generations with at least one reference input, pooled over the tasks of each condition, for Design Anything and JarvisHub, both on DeepSeek V4.1 Flash High; Breadth and Reference: breadth and reference-material tasks. (b) JarvisHub, one point per task with identity pairs: reference coverage against identity consistency (Spearman $\rho = 0.40$, $n = 103$).

Table 11 Per-image quality. Judge score (0–100) averaged within each task and then over the 228 tasks. Bold marks the best value.

System	Quality	Aesthetics	Text	Domain
Design Anything (DS)	91.5	90.2	96.3	95.2
Design Anything (GPT)	91.2	91.0	96.4	95.1
OpenDesign (DS)	89.4	86.4	84.7	89.8
JarvisHub (DS)	**92.3**	**91.6**	**96.7**	**96.1**
Codex (GPT)	90.6	89.6	96.1	92.6

The ordering holds under both judges: Design Anything and JarvisHub stay within 2.4 points of each other on every score, and both lead OpenDesign on every score. Per-task agreement between the two judges is moderate (Spearman 0.37–0.60 across the four scores), so we read per-image quality only at the level of systems. For Design Anything (GPT), the sample's DeepSeek scores come from an earlier, separate judging pass over the same images. Between the two passes a task's group score differs by 1.8 to 4.1 points on average, and the 15-task means differ by at most 2.8 points. This is the repeat variability of the judge on one task; the configuration means in Table 11 average 228 tasks.

C.7 Token accounting

Because every continuation turn re-reads the contract and workspace state, Design Anything's token use grows with project length: on DeepSeek, median input tokens per task rise from 1.4M at 6 artifacts to 6.6M at 14 and 9.1M at 24; on GPT-5.6 Sol, the orchestrating agent alone reads 5.6M, 16.6M, and 18.4M. Table 13 reports median language-model input tokens per task for Design Anything. The DeepSeek counts come from per-task token records; the GPT-5.6 Sol counts come from each run's manifest and cover the orchestrating agent only, excluding helper sessions, so the two columns are not directly comparable. We report no monetary cost and no cross-system token comparison: all Codex runs and 164 of the 228 Design Anything runs on GPT-5.6 Sol went through subscriptions without per-token billing, token records differ in scope across harnesses, and media-generation costs are not aligned.

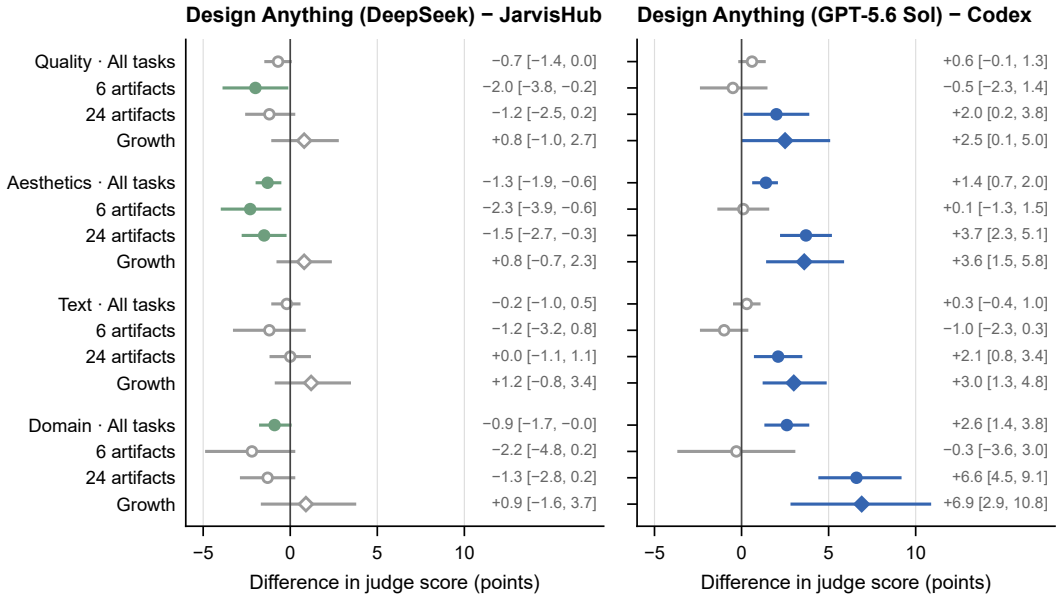


Figure 9: **Paired per-image differences, Design Anything minus the baseline on the same model** (points; mean of per-task differences), with 95% intervals from the project-family-clustered bootstrap of Appendix C.3. Growth (diamonds) is the difference at 24 artifacts minus that at 6. Filled markers have intervals that exclude zero: blue where Design Anything is higher, the baseline's color where it is lower.

Table 12 Per-image quality by project length. Judge score (0–100) averaged over the 38 tasks at 6 / 14 / 24 artifacts.

System	Quality	Aesthetics	Text	Domain
Design Anything (DS)	91.5 / 91.6 / 91.3	90.1 / 90.4 / 90.1	95.8 / 96.6 / 96.7	95.6 / 95.5 / 94.4
OpenDesign (DS)	89.7 / 89.6 / 89.2	86.2 / 87.1 / 85.9	86.5 / 85.8 / 82.7	90.5 / 87.1 / 86.7
JarvisHub (DS)	93.5 / 93.0 / 92.7	92.6 / 92.2 / 92.1	96.8 / 97.1 / 97.4	97.9 / 95.8 / 95.7
Design Anything (GPT)	91.4 / 91.8 / 91.2	92.1 / 91.8 / 90.9	96.8 / 96.5 / 96.5	94.2 / 96.5 / 95.8
Codex (GPT)	91.9 / 90.6 / 89.2	91.9 / 89.7 / 87.2	97.8 / 96.1 / 94.4	94.5 / 91.3 / 89.2

Table 13 Median language-model input tokens per task for Design Anything (millions). The GPT-5.6 Sol column counts the orchestrating agent only.

Condition	DeepSeek V4.1 Flash	GPT-5.6 Sol
6 artifacts	1.36	5.59
14 artifacts	6.58	16.57
24 artifacts	9.06	18.38
Breadth	4.22	16.42
Reference materials	2.37	12.34
All	4.04	14.63

D Component Studies

D.1 Scope and treatment definitions

The component experiments and the cross-system benchmark answer different questions. A comparison between complete systems measures the performance of their combined models, tools, and orchestration. A component study changes a particular mechanism within a controlled configuration. As in the cross-system evaluation, no component-study run used skills written for a particular design domain. Ablation arms run without the automatic verifier. The continuation study reuses 19 corpus tasks; the other studies were conducted on separate task sets and implementation snapshots. No component-study observation is pooled into the headline results of the 228-task corpus.

The planning study has three arms. *Structured* retains the structured project-plan and workflow-construction tools. *Text plan* removes those planning tools but instructs the agent to maintain a free-text deliverable checklist. *Planning tools off* removes the planning tools without adding a replacement planning instruction. Other workspace-reading tools, observations, continuation mechanisms, and requirement channels remain available. This is therefore a study of these planning interfaces within the existing harness, not an experiment that removes all memory, state, or planning ability from the model. The automatic verifier is disabled in all three arms, including Structured, because its trigger would otherwise be available asymmetrically.

D.2 Removing and replacing autonomous continuation

The study uses the 19 tasks with 24 artifacts from brief variant a, one per domain, with the language model and image-generation configuration of the main comparison. For this study we ran the complete harness once more on these 19 tasks, in addition to the main comparison; the other two arms, as ablation arms, run without the automatic verifier. Each arm has one run per task. All three arms are scored with evaluator v1.18 and the same judge and task set. Elapsed time is not compared because the arms ran on different machines. Approval requests raised through the checkpoint tool receive the same scripted answers in every arm; a question written into a final message ends the turn, and without continuation nothing replies to it.

No continuation removes the continuation controller, the contract and planning tools (project plan, workflow blueprint, deliverable review), batched asynchronous execution, and every channel that restates the task or its progress: the per-step instruction anchor, first-message recall, phase-transition prompts, working notes, self-check reminders, canvas-health hints, and progress lines in the status block. Synchronous execution, read tools, context compaction, scripted approval answers, and retries after transport failures remain, since a generic tool-using harness has them. A run ends when the model ends a turn without calling a tool. *Generic continuation* keeps every removal and, whenever the model ends a turn, sends the fixed user message "Continue working until everything the original brief asks for is fully delivered. If everything is already delivered, say so briefly and make no canvas changes." The message names no outstanding item. The loop stops after two consecutive rounds without canvas change, after 100 turns, or at the 150-minute wall-clock limit. *Complete* is the full harness of Section 2.

Table 2 in the main text gives the results. Paired tests are two-sided exact Wilcoxon signed-rank tests paired by task; tasks with equal values are excluded and tied absolute differences receive average ranks. Against the complete harness, the no-continuation arm delivers less on 8 tasks and more on 3 ($p = 0.018$) and satisfies fewer constraints on 11 tasks and more on none ($p = 0.001$); the generic arm delivers more on 7 tasks and less on 4, with 8 ties ($p = 0.39$). Because a constraint none of whose target categories contains a delivered file counts as unsatisfied (Appendix B.1), part of the constraint drop without continuation can follow from undelivered categories: the collapsed runs account for at most 12 of the 24 fewer constraints satisfied, so at least 12 were lost in runs that delivered 20 or more of their 24 artifacts. A run counts as collapsed when it delivers less than half of its deliverables; six no-continuation runs collapse against no complete run (Fisher exact test, two-sided $p = 0.020$, one-sided $p = 0.010$). The collapsed runs are premature stops rather than lost requirements. Every collapsed no-continuation run ended within three turns. In three the final message asked the user to confirm a direction or a costly next stage, in two it announced the next deliverables and then ended the turn, and in one a single step exhausted the output-token limit before anything was built. The generic prompt recovers each of the six to 23 or 24 deliverables.

The generic arm issues more generation requests than the complete harness on 15 tasks and fewer on 3 ($p = 0.002$); counting only completed requests gives 14 and 3 ($p = 0.001$). The extra requests come after the deliverables are largely in place, when the prompt keeps the agent regenerating delivered work; in one task the generic arm issues 68 requests against 27. Constraint satisfaction favors the complete harness on 6 tasks and the generic arm on 1 ($p = 0.063$), so no difference is claimed. Because the generic arm removes the contract, $\mathcal{R}$, the task-recall channels, and the automatic verifier together, these differences are attributed to that set of mechanisms rather than to $\mathcal{R}$ alone.

Table 14 Continuation study: cost per run (17 tasks; the two generic runs that reached the wall-clock limit recorded no step or token counts and are excluded, with their pairs). Input tokens and cost include sub-agents; cost is at list prices.

Measurement	No continuation	Generic	Complete
Agent steps	31.2	79.7	36.4
Input tokens (M)	16.9	41.8	17.9
Language-model cost ($)	0.48	1.30	0.43
Generation requests	18.3	37.1	23.8
Delivery completion (%)	72.1	96.8	96.1

Table 14 gives cost. The generic arm takes more agent steps on all 17 tasks, and more input tokens and language-model cost on 15 of 17 ($p < 0.001$ for each). On the same 17 tasks the no-continuation arm is indistinguishable from the complete harness in steps (8 higher, 9 lower), input tokens (7 and 10), and cost (9 and 8), so the complete harness buys its higher delivery completion without additional language-model spending.

D.3 Planning results and their interpretation

After excluding invalid pilot runs, an eight-task batch reports delivery completion 7.7 points lower with Planning tools off than with Structured, and 2.1 points lower with Text plan; an earlier four-task estimate is invalid and is not used. The differences are likely small because the removed tools are not the only channel carrying requirements: delivery groups, outstanding-work messages, direct canvas reads, status tools, and retained history remain available in every arm, so the study bounds the marginal value of the planning interface given those channels rather than the value of explicit planning as such. The continuation study is consistent with this reading: with a generic continuation prompt and no planning tools, delivery completion matches the complete harness.

D.4 State-summary efficiency

The state-summary experiment removes the canvas digest while retaining direct read tools, and measures the fraction of tool calls devoted to active canvas reading. Nine of ten paired tasks read more without the digest; the geometric mean ratio of reading fractions is 2.06 (one-sided exact sign test, $p = 0.011$). Across all archived runs, 10 with the digest and 13 without, mean canvas reads per run rise from 5.9 to 10.2 and mean total tool calls from 34.4 to 41.0. Agents therefore compensate for a missing summary by inspecting state directly, at higher interaction cost; the experiment does not measure final design quality.

E QUALITATIVE EXAMPLES

E.1 A RECORDED PROJECT AND ITS SHARED DEPENDENCIES

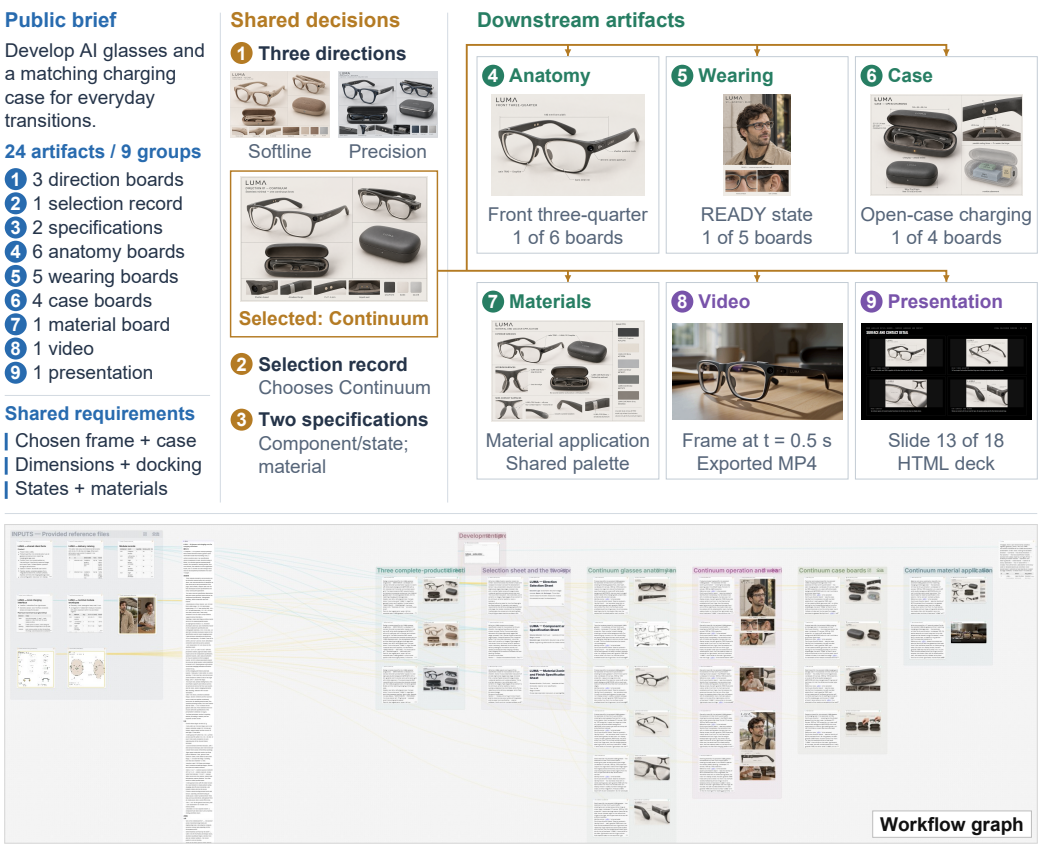


Figure 10: **From one brief to interdependent artifacts.** The LUMA task requests 24 artifacts in nine numbered groups. Selected real outputs show how one run reuses the chosen Continuum direction board across design boards, video, and a presentation; gold arrows mark recorded input connections. Bottom: the run's workflow graph (canvas screenshot cropped to the graph; the tail of one long text node is cut).

Figure 10 uses the LUMA product-design task `bench-da-product-base-a-n24`, from the main-comparison run of Design Anything on DeepSeek V4.1 Flash High. Its public brief requests AI glasses and a matching charging case, with 24 artifacts in nine groups: three direction boards, one selection record, two specifications, six anatomy/detail boards, five wearing/use boards, four case boards, one material board, one video, and one presentation. Shared requirements concern the chosen product identity, dimensions and docking, operating states, and material choices.

The written selection record chooses Continuum from three directions. Its direction-board node has 18 recorded outgoing connections, which make it an input to all 16 anatomy, wearing, case, and material boards, to video generation, and to the presentation. The archived workflow contains 64 nodes and 79 connections; the gold arrows draw these 18 connections once per group and omit the other edges. The selection and specification records are Markdown, the video is an exported MP4, and the presentation is an 18-page HTML deck with rendered pages. The shown video frame is at 0.5 seconds and the presentation preview is page 13; neither is a regenerated illustration.

Under evaluator v1.18, this run delivers all 24 artifacts, satisfies 6 of its 7 constraints; the identity judge rates 16 of 18 image pairs as the same product. We verified the public-brief, manifest, workflow, and source-file hashes and matched the illustrated connections to the recorded graph. The case

was selected for inspectable cross-modal dependencies and readable artifacts rather than a known rank in the result distribution.

E.2 Workflow graphs and image boards of thirteen projects

Figures 11 to 21 show thirteen more projects that we selected for display, each from one recorded run of Design Anything with GPT-5.6 Sol as the language model.

E.3 All outputs of the project in Figure 4

Figures 22 and 23 show, for each of the five systems, every image board, three frames of the walk-through, and three presentation pages delivered for the waterfront-library project of Figure 4, with the image-board layout of the figures above but without workflow graphs. For each requirement we show the system's final file; where the evaluator matched an intermediate version (Codex's three views) or a neighbouring sheet (OpenDesign's elevations and shading strategy), we show the final file named for the requirement, as in Figure 4.



Figure 11: **A 24-artifact fashion project (I).** Each block shows, from top to bottom, the project's workflow graph, its delivered image boards in requirement order, and up to three video frames and three presentation pages. Workflow graphs are canvas screenshots cropped to the graph; in 11 of the 13 projects, the tail of one long text node is cut. Projects were selected for display.

Game art workflow graph, 20 image boards, video frames, presentation pages

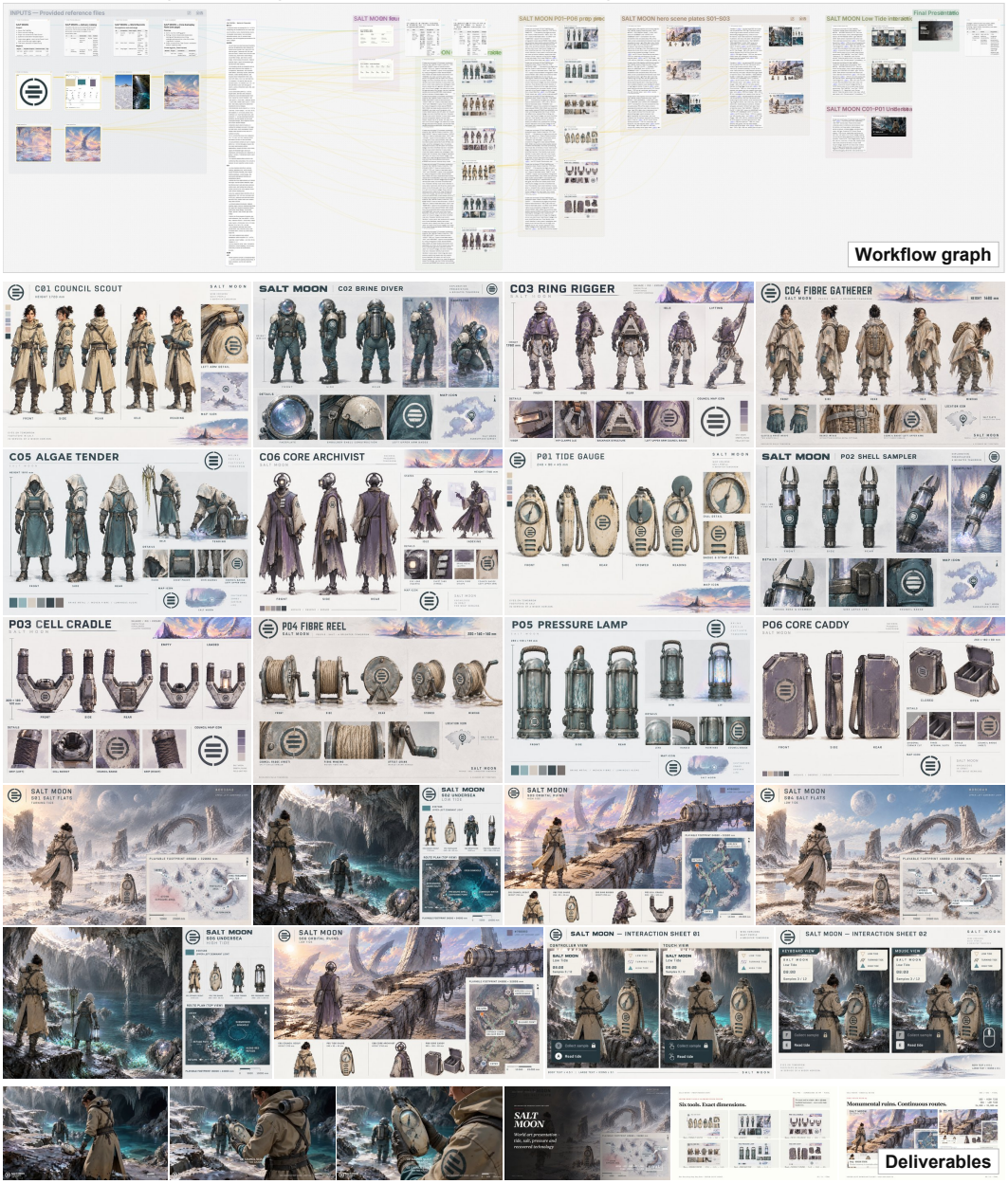


Figure 12: **A 24-artifact game-art project (II),** with the same layout as Figure 11.



Figure 13: **A 24-artifact brand-identity project (III),** with the same layout as Figure 11.

Jewelry workflow graph, 20 image boards, video frames



Figure 14: **A 24-artifact jewelry project (IV),** with the same layout as Figure 11.

Landscape workflow graph, 18 image boards, video frames, presentation pages



Figure 15: **A 24-artifact landscape-design project (V),** with the same layout as Figure 11.

Photography workflow graph, 21 image boards, video frames, presentation pages

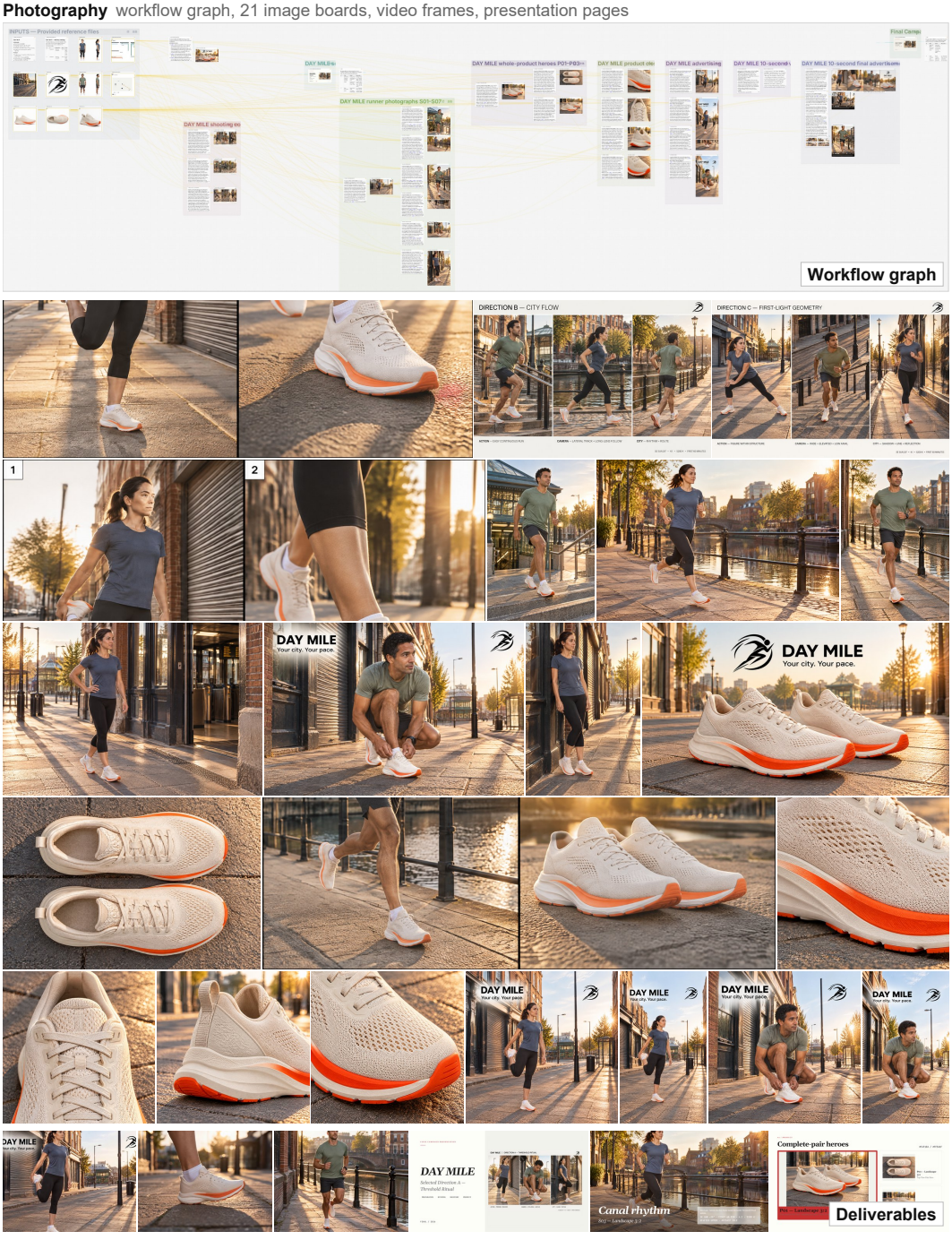


Figure 16: **A 24-artifact photography project (VI),** with the same layout as Figure 11.

Fashion workflow graph, 18 image boards, video frames, presentation pages



Figure 17: **A second 24-artifact fashion project (VII),** with the same layout as Figure 11.

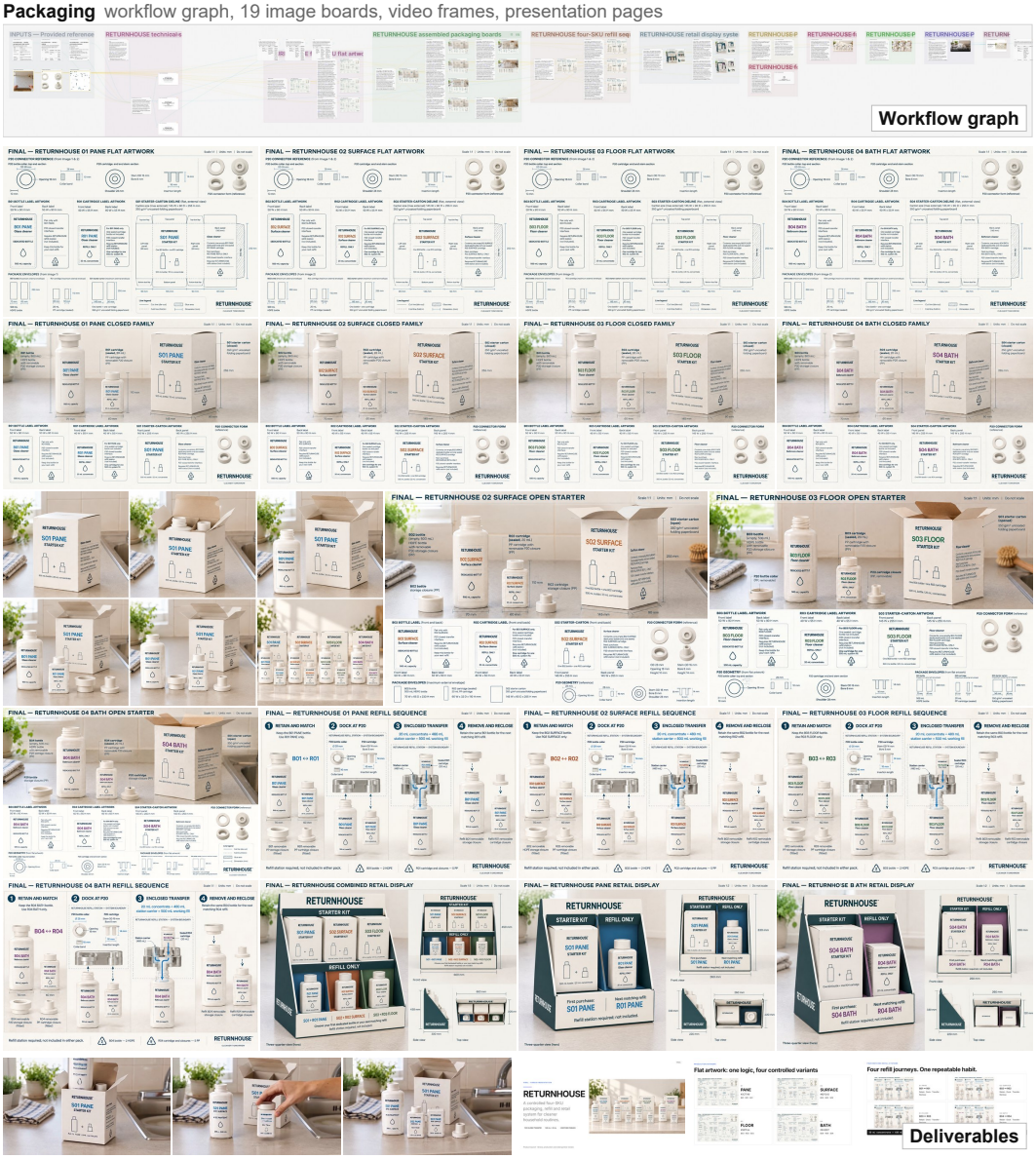


Figure 18: **A 24-artifact packaging project (VIII),** with the same layout as Figure 11.

UI/UX workflow graph, 21 image boards, video frames, presentation pages



Figure 19: **A 24-artifact UI/UX project (IX),** with the same layout as Figure 11; it includes three direction boards.

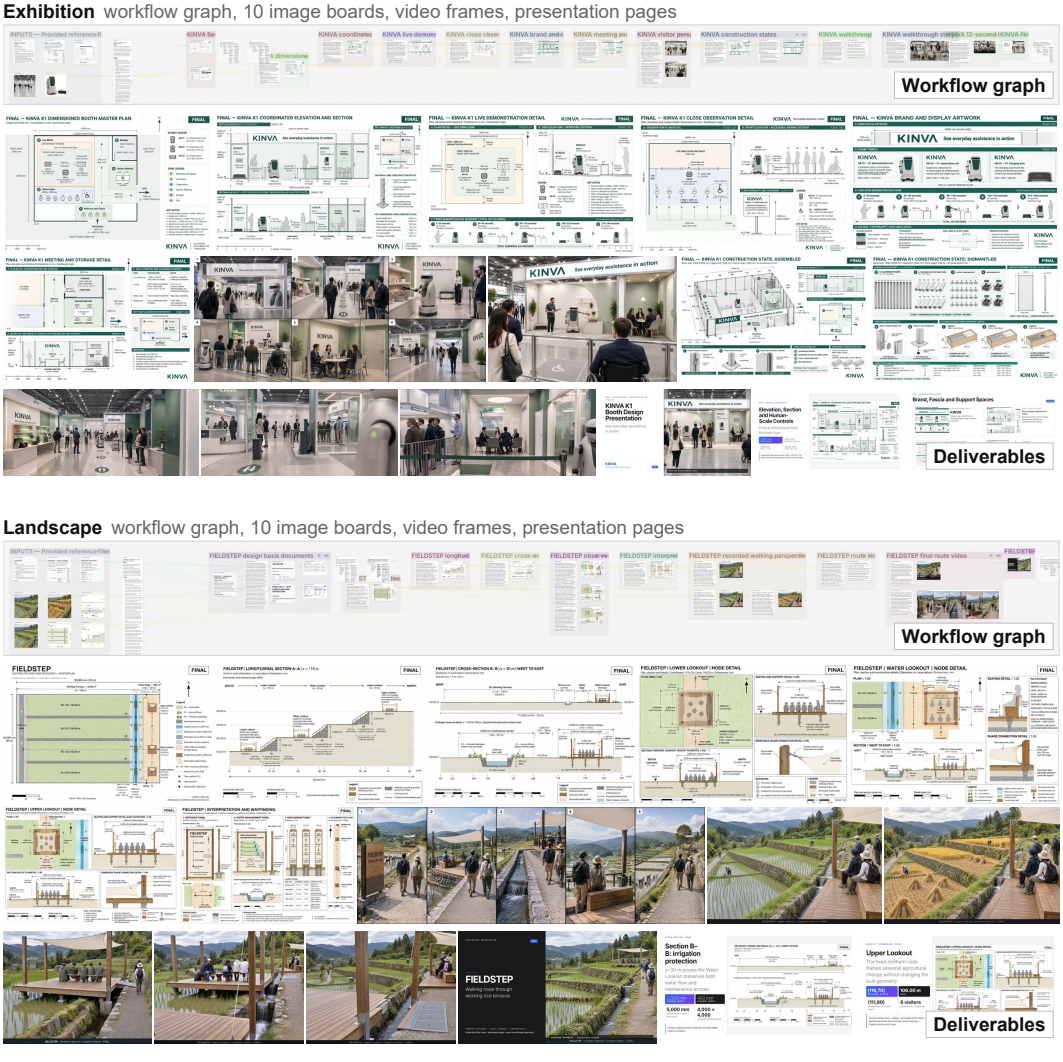
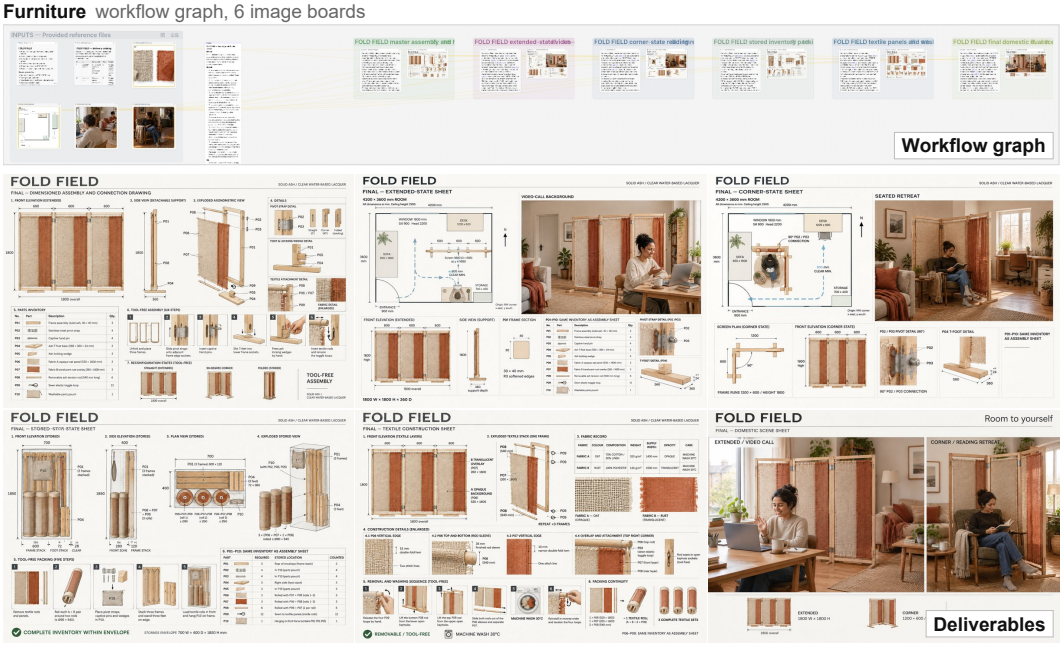


Figure 20: **Two 14-artifact projects (X).** Exhibition design (top) and landscape design (bottom), with the same layout as Figure 11.

Furniture workflow graph, 6 image boards

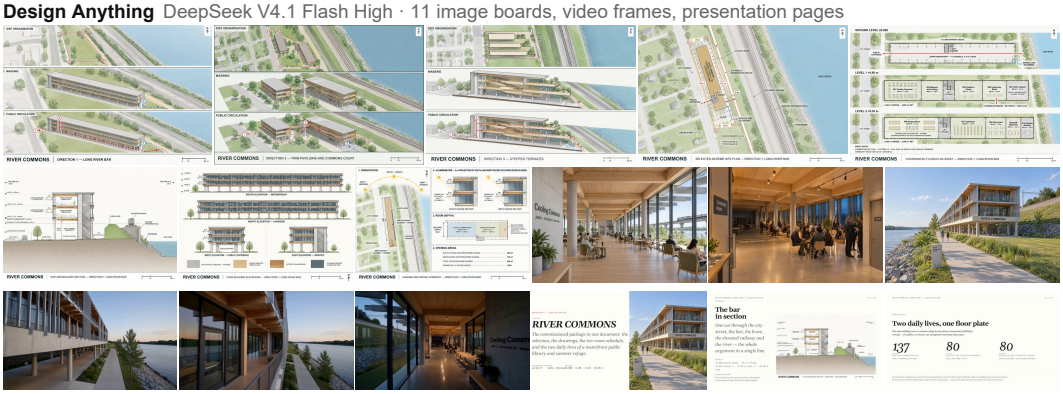


Product workflow graph, 6 image boards

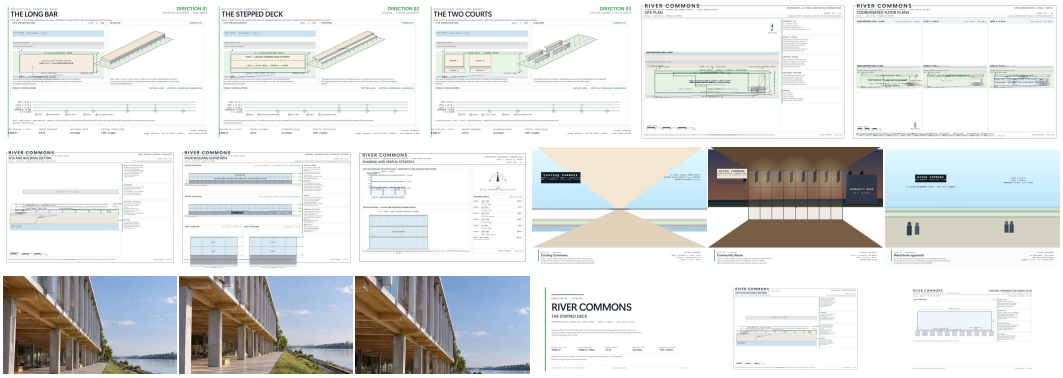


Figure 21: **Two 6-artifact projects (XI).** Furniture (top) and product design (bottom), with the same layout as Figure 11.

Design Anything DeepSeek V4.1 Flash High · 11 image boards, video frames, presentation pages



OpenDesign DeepSeek V4.1 Flash High · 11 image boards, video frames, presentation pages



JarvisHub DeepSeek V4.1 Flash High · 8 image boards

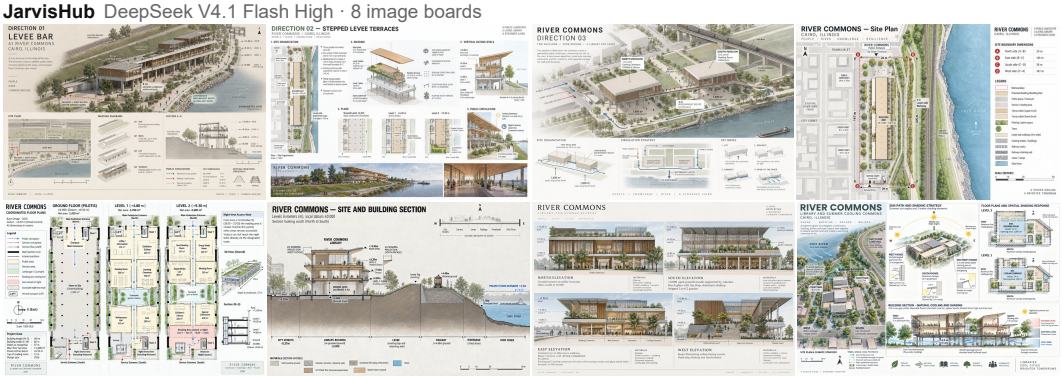


Figure 22: **All outputs of the project in Figure 4: DeepSeek V4.1 Flash High.** Each block shows one system's boards for the 11 image requirements of the brief, in requirement order and uncropped, followed by three video frames and three presentation pages; text documents are not shown.

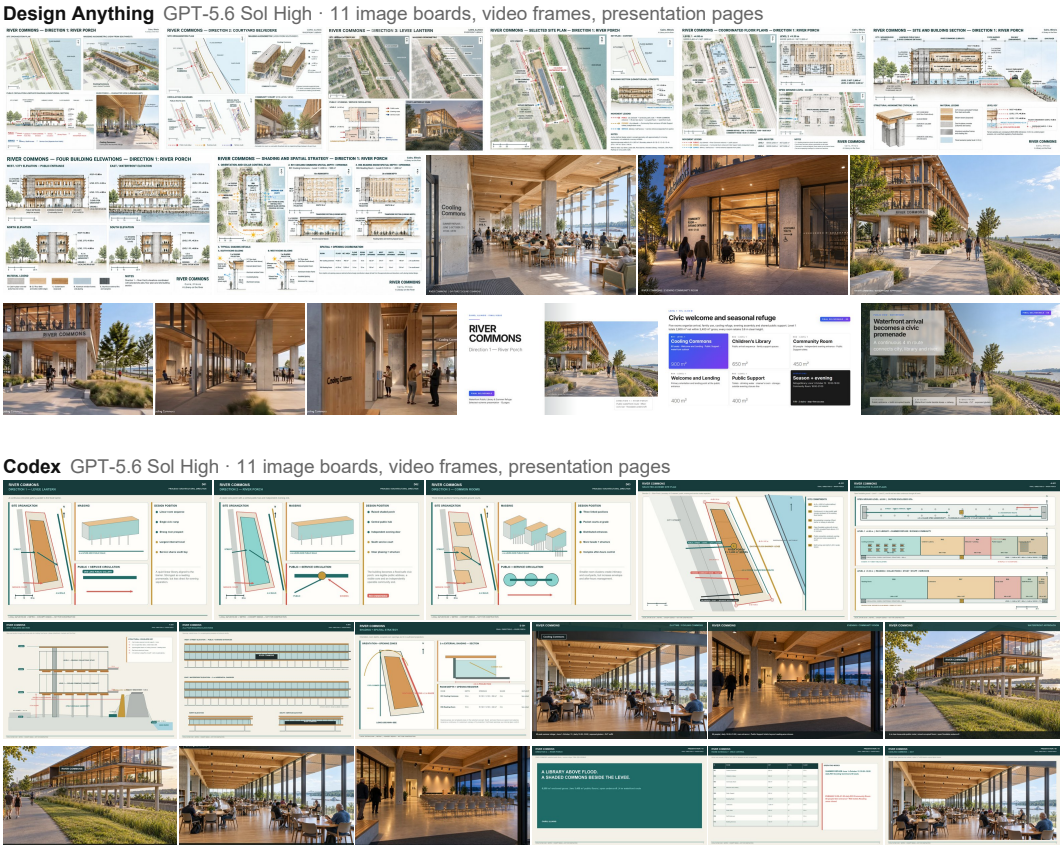


Figure 23: **All outputs of the project in Figure 4: GPT-5.6 Sol High,** with the same layout as Figure 22.

F LIMITATIONS

Each system contributes one final result per task, so differences of a few points between systems may fall within run-to-run variation, which this snapshot does not measure, and the three scale conditions of a family are not independent samples. The judge calibration was performed on the pilot corpus with the same judge prompts; its transfer to the 228-task corpus is assumed, not measured. Two evaluator steps are uncalibrated: category assignment, on which delivery completion depends, and text transcription, on which constraint satisfaction depends. Per-image quality rests on one vision-language judge, checked against a second judge only on a 15-task sample of three systems, and the two judges agree only moderately on individual tasks. The DeepSeek comparison holds the language model and the image generator fixed but not the video generator, and the GPT-5.6 Sol comparison holds only the language model fixed, so cross-system differences reflect the whole product rather than the harness alone. The component studies use small separate task sets and bound the marginal value of each mechanism given the others, not its value in isolation; the planning study's originally registered 20-point decision threshold was withdrawn.