VENUS: COMPILING AN AESTHETIC STANDARD INTO SUPERVISION FOR WEB CODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Web coding models now meet basic functional requirements, yet still struggle to turn a user's subjective aesthetic wishes, a premium look or visual impact, into concrete design. Improving design ability with high-quality web data is bottlenecked by obtaining professional aesthetic supervision at scale: design-quality criteria are usually implicit in a designer's experience and resist conversion into executable rules, while page-by-page human review is reliable but too costly at the data scale training demands. We present VENUS, a pipeline for constructing web-coding training data driven by professional design knowledge. Instead of converting aesthetic intuition fully into automatic rules, VENUS decomposes it into a multi-dimensional, graded, verifiable design standard that drives a two-stage process of data expansion and supervision construction, so that limited human effort supports building professional aesthetic data at scale. High-quality seed pages are expanded along each dimension and a small amount of human review keeps the good ones; guided by the same standard, these pages are then degraded in a controlled way, each degradation along one design dimension, yielding large-scale aesthetic comparison data without new pairwise human annotation. From this data we distill a local aesthetic evaluation model whose automated review builds a larger high-quality corpus, on which we train a web coding model. On WebDevJudge, a public benchmark of real human preferences over web pages, the VENUS evaluation model attains the highest order consistency in the panel and a strict accuracy within the band of the frontier closed-source judges we compare against, and in a blind study it agrees with independent designers at a rate close to their agreement with one another. An open Qwen-family base trained on the VENUS corpus produces web pages of substantially higher design quality than the same base and competitive with frontier systems on public benchmarks. We will open-source the ten-dimension web design standard and the trained models.

1 INTRODUCTION

Web coding models now turn a natural-language request into a runnable page with basic interaction (Zhang et al., 2025; Lu et al., 2026; Lei et al., 2026), and people increasingly use them for real product and business sites, where a working page is not enough: it should look professionally designed. Yet they describe that wish in feeling words, "a premium look" or "visual impact", without stating the layout, hierarchy, typography, colour, or components that would realize it. The model has to supply those decisions on its own, and it tends to seize the most conspicuous keyword, a dark background for "premium" or saturated colour and oversized type for "impact", while missing how design elements agree with one another. The page runs, but it turns out templated, unfocused, or rough, and rarely reaches a professional standard.

Local keyword matching cannot secure holistic quality because a page's visual appearance is not decided by any single attribute; it is the joint effect of layout, hierarchy, typography, colour, and components. Human-computer interaction research finds that web aesthetics comprises several correlated dimensions (Lavie & Tractinsky, 2004; Moshagen & Thielsch, 2010), that the overall impression a user forms is shaped at once by order, complexity, colour, and expressiveness (Lindgaard et al., 2006; Tuch et al., 2012), and that visual hierarchy itself depends on position, size, colour, and salience acting together (Still, 2018). A model therefore has to learn how these elements agree, not just the individual features.

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.

From the training-data side, a supply of high-quality, design-diverse pages is what supports this kind of learning. Building such a corpus means organizing multi-dimensional aesthetic requirements into an operational standard that gives sample construction and quality screening one basis, one that fixes no single page style but states the coherence and completeness a rendered result can be checked against. On that basis, corpus construction faces two coupled problems: candidate supply, how to keep obtaining pages with high quality potential, and quality screening, how to identify the good pages reliably from a large candidate set.

These two problems are each only partly solved. For candidate supply, commercial tools (v0, Lovable, Builder.io, Replit) constrain the structure and style of generated pages with design assets, and datasets such as WebSight (Laurençon et al., 2024), WebCode2M (Gui et al., 2025), and Web2Code (Yun et al., 2024) enlarge and clean web corpora; but following a specification is not the same as high aesthetic quality, so both still need screening before they become high-quality data. For quality screening, automatic signals built from controlled perturbations, as in UIClip (Wu et al., 2024a), or from principle violations, as in UIPrincipleRL (Mehta et al., 2026), have coverage bounded by their preset defect types; expert evaluation through the UICrit (Duan et al., 2024), DesignSense (Gopal et al., 2026), and TASTE (Zhu et al., 2026) datasets gives direct professional supervision but does not extend cheaply to a large set; and frontier vision-language judges (Chen et al., 2024; Xiong et al., 2025; Li et al., 2025a) scale but carry call cost, external dependency, and unstable judgements. Each has part of what is needed, but none meets a professional scale, a low annotation cost, and screening at volume at once, so candidate supply and quality screening have not yet formed one complete, scalable data pipeline.

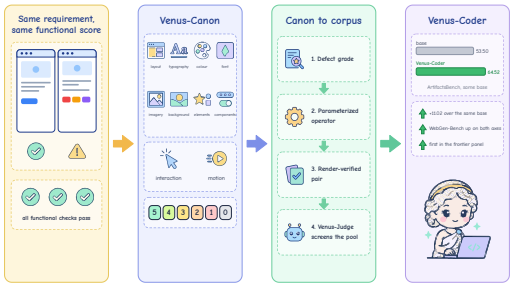

Figure 1: VENUS in one view. Two pages generated from one instruction pass the same functional checks yet differ in everything a designer would notice (left). VENUS-CANON makes that difference nameable as ten design dimensions on six anchored grades; its defect grades compile into page operators that produce labelled pairs without annotation; those pairs train VENUS-JUDGE, the filter that turns a candidate pool into a corpus; and fine-tuning a public base on that corpus yields VENUS-CODER, which lifts ArtifactsBench from 53.50 to 64.52 over the same base and improves both axes of WebGen-Bench (right).

We present VENUS, a framework for building web aesthetic data with a single design standard as its connective tissue, converting a limited amount of expert review into scalable quality supervision. From strong design cases we build seed data pairing a natural-language instruction with a page implementation, which professional designers review to establish a quality reference; guided by the same standard we then apply controlled degradation to produce comparison data carrying a relative preference and a problem attribution, and train an aesthetic evaluation model, VENUS-JUDGE, on it to screen candidate pages at scale (Figure 1). VENUS-JUDGE evaluates the static design dimensions a screenshot can verify, while interaction and motion are supplied by a separate page check. The corpus is then used to train a web coding model, VENUS-CODER; because VENUS-CODER and its base differ only in the corpus, any gain is attributable to what VENUS selected. Our contributions are four. **(i) A multi-dimensional, graded, verifiable web design standard.** VENUS-CANON turns abstract aesthetic requirements into operational criteria that give seed review, degradation construction, and candidate screening one basis, so quality supervision stays consistent across stages. **(ii) A two-stage data method that expands expert-reviewed seeds into comparison supervision.** A small amount of human review establishes a high-quality reference, and standard-guided controlled degradation automatically generates comparison data with preference relations and problem attribution. **(iii) An aesthetic evaluation model for large-scale web screening.** Trained on that data, a

7B judge screens pages at scale and reduces the reliance of large-scale screening on page-by-page human review and frontier-model calls, using a dual-order decision to curb position bias. **(iv) Evidence that aesthetic data construction helps web generation.** Under the same base and training setup, the controlled comparison with VENUS-CODER shows the constructed corpus improves both the functional and the visual design quality of generated pages.

2 RELATED WORK

Front-end quality evaluation. As web generation moved from static code to runnable, interactive pages, evaluation widened from visual reconstruction to function, appearance, and interaction. Design2Code (Si et al., 2025) and WebSight (Laurençon et al., 2024) framed screenshot-to-code as visual reconstruction, Web2Code (Yun et al., 2024) and WebCode2M (Gui et al., 2025) scaled the data behind it, and WebUIBench (Lin et al., 2025) and DesignBench (Xiao et al., 2025) broadened the axes on which front-end output is scored. ArtifactsBench (Zhang et al., 2025) scores a temporally spaced screenshot sequence against a per-task checklist and WebGen-Bench (Lu et al., 2026) drives the deployed site with a navigation agent and reports a separate appearance score; others reach whole builds (He et al., 2026), the development lifecycle (Lei et al., 2026), end-to-end browser tests (Xu et al., 2025), and design capability (Jung et al., 2025). On the design-quality side, UIClip (Wu et al., 2024a) and UICrit (Duan et al., 2024) learn quality and critique from annotated screenshots, AesEval-Bench (An et al., 2026) and NIMA (Talebi & Milanfar, 2018) assess aesthetics along fixed dimensions, recent work detects principle violations (Mehta et al., 2026) or repairs interactive pages from browser state (Wu et al., 2026), and DesignSense (Gopal et al., 2026) and TASTE (Zhu et al., 2026) supply graphic-layout preference and designer multi-dimensional ratings. These lines grow front-end quality from "does it run" to "is it well designed", but mostly serve measurement or feedback; none turns one standard into seed review, comparison supervision, and corpus screening at once, which VENUS-CANON does.

Model judges and their failure modes. Using a strong model to judge another model's output is standard practice since MT-Bench (Zheng et al., 2023), JudgeLM (Zhu et al., 2025), and Prometheus (Kim et al., 2024), and MLLM-as-a-Judge (Chen et al., 2024) and LLaVA-Critic (Xiong et al., 2025) extend it to multimodal inputs. The failure modes are well documented: judgements shift with candidate order and can flip when the two pages are swapped (Wang et al., 2024; Shi et al., 2025), they favour verbose (Saito et al., 2023) or self-generated (Panickssery et al., 2024) outputs, and reward benchmarks measure these effects at scale (Lambert et al., 2025; Li et al., 2025a); a parallel line replaces scalar preference with rubrics (Hashemi et al., 2024; Gunjal et al., 2026) and checklists (Lee et al., 2025). In the web setting, WebDevJudge (Li et al., 2026a) shows that current judges still trail experts and that more elaborate interactive judging does not necessarily beat direct comparison. VENUS-JUDGE targets large-scale corpus screening: it takes two rendered screenshots, uses both A/B and B/A orders in training and inference, and reports the design dimension behind each difference, a local filter rather than a general judging paradigm.

Web code generation. Early visual-to-code work generates an implementation from an interface image: pix2code (Beltramelli, 2018) produces a constrained DSL from a screenshot, Design2Code (Si et al., 2025) benchmarks HTML/CSS from high-fidelity screenshots, and Sketch2Code (Li et al., 2025b) extends the input to low-fidelity sketches. Text-to-interface work generates code from a requirement: UICoder (Wu et al., 2024b) filters and self-trains SwiftUI generation, WebGen-Bench (Lu et al., 2026) trains WebGen-LM from filtered agent trajectories, AesCoder (Xiao et al., 2026) adds aesthetic reward, and ReLook (Li et al., 2026b) and UI2Code^N (Yang et al., 2025) fold rendering, diagnosis, and revision into a generate-inspect-improve loop; data-side work such as WebSight (Laurençon et al., 2024), Web2Code (Yun et al., 2024), and WebCode2M (Gui et al., 2025) enlarges training resources. Several improve training data through quality filtering or aesthetic supervision; VENUS instead makes a professional aesthetic standard the filtering mechanism, training a local judge from one standard, screening a large pool, and testing the corpus by same-base fine-tuning with a data ablation.

3 METHOD

A designer experiences page quality as a whole, not as a sum of independent scores. VENUS decouples that judgement into dimensions only so that each becomes checkable and reproducible; both the standard and the judge are built to respect how the dimensions interact, since a change in layout shifts the perceived typography and a change in colour shifts the perceived hierarchy. VENUS turns a limited amount of professional design judgement into stable supervision over a large candidate pool through one design standard that runs the length of the pipeline. The standard organizes web design quality into graded, verifiable criteria. Designers screen pages recovered from strong design cases into a small seed set. The same standard then drives controlled degradation of those seeds into comparison data, which trains a local judge, VENUS-JUDGE. VENUS-JUDGE screens a large candidate pool into a corpus, and that corpus fine-tunes a web coding model, VENUS-CODER. The sections below follow this order.

3.1 VENUS-CANON: AN EXECUTABLE AESTHETIC STANDARD

VENUS-CANON is the foundation the rest of VENUS is built on: ten visual dimensions, eight static and two dynamic, each resolved into six grades, written so that two reviewers, or a reviewer and a model, interpret the same page in a more consistent way. The grades do not promise identical scores; they reduce the interpretation gap that free-form aesthetic judgement leaves open, which is what later lets the standard be applied at scale.

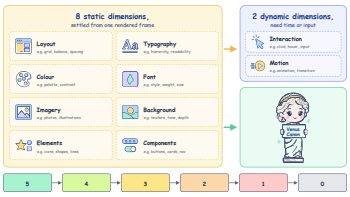

Figure 2: The ten dimensions of VENUS-CANON. Eight static dimensions are settled from a single rendered frame; two dynamic ones, interaction and motion, by operating the page over time. Every dimension runs the same six grades, from 5 (flawless and distinctive) down to 0 (not implemented).

How the standard was written. The dimensions come out of professional critique. Working with professional interface designers over several rounds of reviewing generated pages, we collected the judgements that recurred across genres and rewrote each as a property of the rendered page that a second reviewer could confirm, dropping the judgements that resisted such a rewriting. Decoupling into dimensions is a device for checkability, not a claim that the dimensions are independent: the grade descriptions cross-reference one another (a colour verdict refers to the layout regions it applies to), and both seed screening and the judge see the whole page, not one dimension in isolation. The result covers the static side with eight dimensions, from *layout* and *typography* to *elements* and *components*, and the dynamic side with *interaction* and *motion* (Figure 2); a screenshot settles the eight static dimensions, and operating the page settles the two dynamic ones, so the two halves are screened by different instruments.

Six grades, each a check. Each dimension runs from 5 down to 0 with fixed semantics: 5 is flawless and distinctive, 4 has at most negligible defects, 3 a visible minor defect, 2 a major one, 1 present but unusable, and 0 the dimension absent from the render. What makes the standard usable at scale is that every grade names a property of the artifact: a static grade is settled by inspecting the rendered frame, and a dynamic grade by operating the page and comparing its states. For a page on a regular grid, for instance, a high layout grade is read against concrete check items, the consistency of spacing between blocks, the alignment among columns, and the coordination of content margins, the executable checks a well-composed layout is expected to pass. Where a dimension needs vocabulary, the standard supplies it, defining the colour roles by area and salience so that a single decisive accent becomes a question about regions of the screenshot. Appendix A reproduces all ten dimensions at all six grades.

Anchored grades, not score bands. The prevailing alternative writes aesthetics as score bands, where 90 to 100 means outstanding, which tells the reviewer how good the page should feel but leaves the criterion to them; an anchored grade supplies the criterion itself, which is what lets the standard be executed by a machine. VENUS-CANON also separates 0, the dimension absent from the render, from 1, present but unusable, so a page that never loaded and one with text on a button are different failures on one scale. Not every dimension is in play for every page: a dimension is scored against what the page's design calls for, so a static business page is not marked down on motion for an effect it never sought, and grade 0 records a dimension the page needed but failed to render, not one outside its scope. Every grade is thus *grounded*, naming a practice designers recognize; *checkable* from the render; and, for the defect grades, *reproducible*, since the defect can be introduced by an edit. The first two properties let designers screen seeds consistently; the third underlies the comparison supervision below.

3.2 HIGH-QUALITY SEED CONSTRUCTION

Supervision enters VENUS through pages that designers have actually endorsed, so human judgement is spent where it is cheapest and most reliable, at the small seed stage (Figure 3).

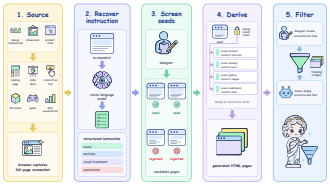

Figure 3: The data pipeline. Reference pages are sourced across six genres and captured in a browser; a vision-language model recovers a structured instruction from each capture; designers screen the generated candidates into seeds against VENUS-CANON; semantic derivation expands each seed into a family of instructions; and the resulting pool is screened by VENUS-JUDGE.

Sourcing. The reference points practitioners consult, design communities, curated showcases and template galleries, and the public sites of products known for their front-end craft, hold up exemplary pages that cluster into six genres: landing pages, slide decks, interactive tools, 3D scenes, games, and data visualizations. Each genre loads a different part of the standard, so the six together exercise every dimension of VENUS-CANON and keep the corpus from collapsing onto a single template. Each sourced page is rendered in a headless browser at a fixed viewport and captured as a full-page screenshot, the representation every later stage works from.

Instruction recovery. A vision-language model reads each capture and emits a structured instruction with four parts: the scene and purpose of the page, its sections and their order, the visual treatment of each section, and the constraints a reimplementation must respect. This is where the colloquial-to-specification gap is bridged: the recovered instruction re-expresses what a user would say in feeling words as the concrete visual attributes VENUS-CANON scores, so "premium" or "high impact" resolve into stated choices of layout, typography, colour, and components. We invert the rendering rather than transcribe the source: a transcript from HTML leaks the answer into the question, while a recovery stays in the language users write in and VENUS-CANON judges in.

Seed screening. Designers apply VENUS-CANON to the pages generated from the recovered instructions and keep the ones that clear it, the first of the two points where human judgement enters the pipeline: a candidate becomes a seed only when no static dimension falls below the minor-defect grade, so the survivors are pages a designer is willing to hold up as references. Their instructions become templates known to have produced a well-composed page at least once, the property that derivation relies on.

3.3 CONTROLLED DEGRADATION AND COMPARISON SUPERVISION

A defect grade written concretely enough to be checked is also concrete enough to be produced: "one or two gaps are visibly larger or smaller than the rest" is both a verdict and an edit. We realize the defect grades of every static dimension of VENUS-CANON as parameterized HTML and CSS transformations, the grade's severity controlling the strength of the edit, with several available where a visual defect has more than one mechanism (Figure A1, Appendix A). Because these static dimensions interact, an edit is applied only when a render check confirms it moves the intended dimension without silently degrading another; an operator that also breaks an untargeted dimension, for instance a spacing edit that forces text to overflow its container, is rejected, so the pair isolates the dimension it is meant to teach.

Applying such an operator to a page that has cleared designer review yields a counterfactual that is worse along the intended dimension by a known amount: the original wins, the degraded copy loses, and the responsible dimension is the one the operator touched, so the pair carries a preference, a severity, and a dimension label without a new round of pairwise annotation. The good side of each pair is still human-selected, at the seed stage; what the construction removes is the cost of labelling every comparison by hand. Two constraints keep the pairs honest: any pair whose edit produces no measurable change in the render is discarded, and every pair is shown in both orders during training so that the judge learns the pages rather than their positions. This construction yields the comparison set on which VENUS-JUDGE is trained; a small set of de-noised natural preferences from public arena data (Chiang et al., 2024), disjoint from every test item, is added so the judge also sees the unstructured differences real pages exhibit.

Controlled degradation is distinct from the semantic derivation that builds the generator's corpus. Semantic derivation takes a reviewed seed and varies the factors a user would vary, the scene, the content, and surface constraints such as palette or copy, while holding fixed the factors VENUS-CANON scores, the sectioning discipline, the hierarchy, and the component grammar; the design intent is carried over, the surface is not, so the derived family stays within the design envelope of a page a designer endorsed while spanning many concrete requirements. The derived instructions generate a pool many times larger than the seed set, but a sound instruction does not guarantee a sound page, since the same instruction yields a clear hierarchy on one draw and a flat one on the next. Selecting the well-designed pages out of that pool is where corpus quality is decided.

3.4 THE VENUS-JUDGE MODEL

VENUS-JUDGE is **Qwen2.5-VL-7B-Instruct** fine-tuned on the comparison set, so the whole filter runs on local weights, cheap enough to score the whole pool on a single GPU. It takes two rendered screenshots and returns a verdict with the responsible design dimension, a label the operator that built each pair already carries. The aim is narrow: a local filter that compares professional design quality stably.

3.5 CANDIDATE SCREENING AND CORPUS CONSTRUCTION

Screening is the one stage whose cost grows with the pool rather than with the corpus, and it is where the corpus is made or lost. The candidate pool is produced by prompting frontier web generation systems with the derived instructions; we study whether standard-driven screening converts into a generation gain, so we do not tie the pipeline to any specific candidate generator, and VENUS-JUDGE then selects the corpus from that pool. We screen with the local VENUS-JUDGE rather than a frontier VLM because scoring every candidate in the pool with a frontier model is prohibitively costly, while VENUS-JUDGE matches those models on human preferences at a fraction of the cost.

Screening against a reference. VENUS-JUDGE returns a pairwise verdict, so we screen against a fixed reference rather than in a tournament: every derived page is compared with the seed it was derived from, in both orders, and kept only when VENUS-JUDGE rates the seed as the weaker page in both. A derived page thus enters the corpus only when it improves on a reference already known to clear VENUS-CANON, not merely when it outranks other generated pages, and each page costs one order-consistent comparison.

Settling the dynamic dimensions. VENUS-JUDGE settles the eight static dimensions from the screenshot. Interaction and motion cannot be read from a still frame, so we check the two dynamic dimensions by operating the page, following WebVoyager (He et al., 2024); only pages that also pass this dynamic check enter the corpus. Across the candidate pool, this screening keeps 68.8% of the candidates as corpus.

3.6 THE VENUS-CODER GENERATOR

VENUS-CODER is the open base Qwen3.6-35B-A3B fine-tuned on the corpus that clears VENUS-JUDGE. The corpus is the only supervision added: architecture, scale, and decoding are held to the base, so a difference between VENUS-CODER and Qwen3.6-35B-A3B in generated pages is attributable to the data the standard-driven screening selected. The experiments below test this controlled setup.

4 EXPERIMENTS

Our experiments ask two things. First, does VENUS-JUDGE, trained only on constructed degradation pairs, judge web aesthetics the way a professional designer does? Second, does a corpus screened by that judge make a web coding model generate better-designed pages? We look at the judge first, where agreement with designers is the primary evidence and a public benchmark an external check, then the generator, where a controlled comparison against its own base isolates the corpus effect and public code benchmarks give a reference, and close with an ablation separating screening from data volume.

4.1 EXPERIMENTAL SETUP

VENUS-JUDGE. The primary check on VENUS-JUDGE is human: three independent senior designers each compare operator-generated pairs in a blind study, and we report how often the judge agrees with them. As an external, standardized check we also run WebDevJudge (Li et al., 2026a), a public benchmark whose pairs carry real human preferences over distinct, independently sourced web pages, unlike our constructed operator pairs, and report strict accuracy and order consistency (how often a judge picks the same page after the two candidates are swapped) against frontier and open-weight judges under one protocol.

VENUS-CODER. For the generator, the effect we want to isolate is that of the screened corpus, so the central comparison is between VENUS-CODER and its own base Qwen3.6-35B-A3B, which share architecture, scale, decoding, and training recipe and differ only in the corpus. Public code benchmarks provide a reference alongside this comparison. ArtifactsBench (Zhang et al., 2025) scores a screenshot sequence of each artifact against a per-task checklist over 1,825 tasks; WebGen-Bench (Lu et al., 2026) drives 101 deployed sites with a navigation agent over 647 tasks for functional accuracy and scores appearance from screenshots. Every system is generated and judged under one harness with the same prompts, rendering stack, and judge (Gemini 3.1 Pro), following ArtifactsBench's official single-judge protocol. Because this shared judge applies the standard more strictly than each benchmark's own leaderboard, absolute scores are not comparable to published numbers; the tables compare within-table ordering under identical conditions, a controlled cross-system comparison (Appendix F).

4.2 VENUS-JUDGE EVALUATION

We first ask whether VENUS-JUDGE agrees with professional designers on which of two pages is better designed, the judgement it will make on every page of the pool.

Agreement with professional designers. In the blind study, each of the three senior designers judged the same operator-generated pairs without seeing how they were built. VENUS-JUDGE chose the same page as each designer on 77.2% of the pairs they judged decisively, 75.4 to 78.3% across the three, and matched the designer majority on 80.6%, rising to about 86% on the pairs where the judge is order-consistent (Appendix E). The three designers agree with one another on 85.7%

of shared decisive pairs, so VENUS-JUDGE sits close to, though still below, the level at which the designers agree among themselves, close enough for a local 7B model to screen the pool in place of page-by-page designer review.

External check on WebDevJudge. Table 1 further tests VENUS-JUDGE's agreement with human judgement on WebDevJudge, a public benchmark of web-page preferences.

Table 1: Natural preference agreement on WebDevJudge. Strict requires the verdict correct in both presentation orders; Consistency requires the same page to win after the candidates are swapped. ✓ open weights, ✗ closed.

System	W.	Strict	Consist.
GPT-5.5	✗	**59.9**	91.5
Claude Sonnet 4.6	✗	55.6	85.0
VENUS-JUDGE (7B)	✓	55.6±1.0	**93.2**±1.8
Claude Opus 4.8	✗	53.9	84.6
Gemini 3.1 Pro	✗	53.5	82.0
Qwen3-VL-32B	✓	48.9	74.6
Qwen2.5-VL-72B	✓	48.7	74.8
Qwen3-VL-8B	✓	47.7	77.2
untrained backbone	✓	40.8	71.2

VENUS-JUDGE reaches 55.6 strict accuracy, inside the band occupied by the frontier systems and level with Claude Sonnet 4.6, while the untrained backbone scores 40.8 under the same protocol, so the agreement comes from the compiled supervision, not the backbone. Order consistency matters most for a filter, since a judge that flips under a swap rejects pages for positional reasons and leaves no trace; VENUS-JUDGE reaches 93.2, the highest in the panel, from the mirrored presentation during training. Each verdict also names the responsible design dimension, a cue for reviewing the screening. On a single H100 a single-order verdict takes about 0.5 s; judging both orders, measured throughput is about 3,200 page-pairs per GPU-hour, enough to screen candidates at scale.

4.3 VENUS-CODER EVALUATION

We now ask whether the corpus VENUS-JUDGE selects makes the generator produce better-designed pages. The evidence is the controlled comparison between VENUS-CODER and its base Qwen3.6-35B-A3B; the frontier panel and the public benchmarks put that gain in context.

Table 2: Two public benchmarks under one harness, prompts, and judge; each column is internally comparable but not comparable to published leaderboards (§4.1). **Bold** best in a column, underline second; ✓ open weights, ✗ closed.

		ArtifactsBench	WebGen-Bench	
System	W.	Score	Func.	Appear.
VENUS-CODER	✓	**64.52**	57.0	4.26
DeepSeek-V4-Pro	✓	63.91	52.8	4.15
GLM-5.2	✗	62.60	44.7	3.29
GPT-5.5	✗	62.55	54.6	4.22
Gemini 3.6 Flash	✗	60.26	56.9	**4.33**
Claude Opus 4.8	✗	56.88	**57.1**	4.15
Qwen3.6-35B-A3B	✓	53.50	46.2	3.54
AesCoder	✓	38.87	27.9	3.13

The VENUS corpus lifts the base on both benchmarks. Against the original base Qwen3.6-35B-A3B, VENUS-CODER raises ArtifactsBench from 53.50 to 64.52, a gain of 11.02 points, and on WebGen-Bench functional accuracy from 46.2 to 57.0 and appearance from 3.54 to 4.26, so post-training on the VENUS corpus improves the functional and the visual quality of generated pages together; how much comes from quality screening and how much from added data is examined by the corpus ablation next. A blind study confirms the gain on VENUS-CODER's own pages: three senior designers prefer VENUS-CODER to the base on 77.6% of the paired pages by majority vote (Appendix E).

Comparison with frontier models. Under our unified protocol, VENUS-CODER's ArtifactsBench score of 64.52 is the highest among the compared models, above DeepSeek-V4-Pro's 63.91, and on WebGen-Bench it ranks second on both axes, at 57.0 and 4.26, so an open model fine-tuned on the VENUS corpus is competitive with frontier systems on the evaluated tasks.

4.4 CORPUS ABLATION

To confirm the gain is not explained by added training data alone, we hold the base and recipe fixed and vary only which pages of the same pool become the corpus: a size-matched random subset, the full unfiltered pool, and the VENUS-JUDGE-screened corpus, isolating quality screening from volume (Table 3).

Table 3: Corpus ablation: one base, recipe, and pool, rows differing only in which pages become the corpus. The random subset fixes VENUS-CODER's corpus size and drops the criterion. Harness and judge as in Table 2.

	ArtifactsBench	WebGen-Bench	
Training corpus	**Score**	**Func.**	**Appear.**
none (the base Qwen3.6-35B-A3B)	53.50	46.2	3.54
random subset of the pool, size-matched to ours	59.22	34.6	3.07
the whole pool, unfiltered	61.99	49.8	3.70
VENUS-JUDGE-filtered *(= VENUS-CODER)*	**64.52**	**57.0**	**4.26**

Both unscreened corpora clear the base on ArtifactsBench by 5.72 and 8.49 points, so the pool carries some supervision by itself, but neither reaches VENUS-CODER: at a fixed size screening adds 5.30 points over a random subset, and even the full pool stays 2.53 below it. On WebGen-Bench the random subset falls below the untrained base on both axes, so volume alone does not explain the gain, screening against the standard does. Training for appearance alone also falls short, with AesCoder trailing every column at 38.87, 27.9, and 3.13, though confounded by scale.

4.5 SUBGROUP AND QUALITATIVE ANALYSIS

The gain differs across deliverable types (Appendix D): the non-HTML group rises most, from 47.06 to 58.90, so the design gain extends beyond the single-page HTML that dominates training. Qualitative cases locate it in composition and hierarchy (Appendix C): on the WebGen software-solutions site the base renders its sections at near-uniform visual weight, whereas VENUS-CODER separates the heading levels, unifies card spacing, and concentrates the accent colour on the primary action.

5 DISCUSSION AND LIMITATIONS

VENUS-CANON scores how a page looks, not whether it works, so the gain on functional accuracy is a by-product, and the pipeline applies only a pass/fail check on interaction and motion, leaving its reading of the dynamic experience limited. Comparing VENUS-CANON against another equally strong filter is left open, since scoring the whole pool with a frontier model is impractical at scale, and the designer agreement rests on three senior designers rather than a larger panel.

6 CONCLUSION

Generated web pages learn to work before they learn to look designed, and the gap is a missing definition more than missing capacity. VENUS supplies that definition as VENUS-CANON, an executable design standard, and turns it into scalable supervision: designer-screened seeds and standard-guided controlled degradation compile into a comparison set that trains VENUS-JUDGE, a local 7B filter whose screened corpus lifts a public 35B base on both ArtifactsBench and WebGen-Bench under one controlled comparison. Because the corpus is the only supervision that changes, the gain is attributable to what the standard selects, not to added scale. We will open-source the standard and the trained models, and leave a fuller treatment of interaction and motion to future work.

AI USE STATEMENT

We used an AI assistant, specifically a large language model, solely to refine the grammar, phrasing, and structure of this manuscript. It was not used to generate any of the core scientific content, such as the methodology, experiments, or conclusions presented herein.

REFERENCES

Arctanx An, Shizhao Sun, Danqing Huang, Mingxi Cheng, Yan Gao, Ji Li, Yu Qiao, and Jiang Bian. Can vision language models assess graphic design aesthetics? a benchmark, evaluation, and dataset perspective. *arXiv preprint arXiv:2603.01083*, 2026.

Tony Beltramelli. pix2code: Generating code from a graphical user interface screenshot. In *Proceedings of the ACM SIGCHI symposium on engineering interactive computing systems*, pp. 1–6, 2018.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. *arXiv preprint arXiv:2402.04788*, 2024.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

Peitong Duan, Chin-Yi Cheng, Gang Li, Bjoern Hartmann, and Yang Li. Uicrit: Enhancing automated design evaluation with a ui critique dataset. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–17, 2024.

Varun Gopal, Rishabh Jain, Aradhya Mathur, Nikitha SR, Sohan Patnaik, Sudhir Yarram, Mayur Hemani, Balaji Krishnamurthy, and Mausoom Sarkar. Designsense: A human preference dataset and reward modeling framework for graphic layout generation. *arXiv preprint arXiv:2602.23438*, 2026.

Yi Gui, Zhen Li, Yao Wan, Yemin Shi, Hongyu Zhang, Bohua Chen, Yi Su, Dongping Chen, Siyuan Wu, Xing Zhou, et al. Webcode2m: A real-world dataset for code generation from webpage designs. In *Proceedings of the ACM on Web Conference 2025*, pp. 1834–1845, 2025.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *International Conference on Learning Representations*, volume 2026, pp. 127924–127945, 2026.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. Llm-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 13806–13834, 2024.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890, 2024.

Zehai He, Wenyi Hong, Zhen Yang, Ziyang Pan, Mingdao Liu, Xiaotao Gu, and Jie Tang. Vision2web: A hierarchical benchmark for visual website development with agent verification. *arXiv preprint arXiv:2603.26648*, 2026.

Sam Jung, Agustin Garcinuno, and Spencer Mateega. Ui-bench: A benchmark for evaluating design capabilities of ai text-to-app tools. *arXiv preprint arXiv:2508.20410*, 2025.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353, 2024.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, 2025.

Hugo Laurençon, Léo Tronchon, and Victor Sanh. Unlocking the conversion of web screenshots into html code with the websight dataset. *arXiv preprint arXiv:2403.09029*, 2024.

Talia Lavie and Noam Tractinsky. Assessing dimensions of perceived visual aesthetics of web sites. *International journal of human-computer studies*, 60(3):269–298, 2004.

Yukyung Lee, Joonghoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. Checkeval: A reliable llm-as-a-judge framework for evaluating text generation using checklists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 15771–15798, 2025.

Xinping Lei, Xinyu Che, Junqi Xiong, Chenchen Zhang, Yukai Huang, Chenyu Zhou, Haoyang Huang, Minghao Liu, Letian Zhu, Hongyi Ye, et al. Webcompass: Towards multimodal web coding evaluation for code language models. *arXiv preprint arXiv:2604.18224*, 2026.

Chunyang Li, Yilun Zheng, Xinting Huang, Tianqing Fang, Jiahao Xu, Lihui Chen, Yangqiu Song, and Winston Hu. Webdevjudge: Evaluating (m) llms as critiques for web development quality. In *International Conference on Learning Representations*, volume 2026, pp. 116724–116756, 2026a.

Lei Li, Yuancheng Wei, Zhihui Xie, Xuqing Yang, Yifan Song, Peiyi Wang, Chenxin An, Tianyu Liu, Sujian Li, Bill Yuchen Lin, et al. Vl-rewardbench: A challenging benchmark for vision-language generative reward models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24657–24668. IEEE, 2025a.

Ryan Li, Yanzhe Zhang, and Diyi Yang. Sketch2code: Evaluating vision-language models for interactive web design prototyping. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3921–3955, 2025b.

Yuhang Li, Chenchen Zhang, Ruilin Lv, Ao Liu, Ken Deng, Yuanxing Zhang, Jiaheng Liu, and Bo Zhou. Relook: Vision-grounded rl with a multimodal llm critic for agentic web coding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25471–25485, 2026b.

Zhiyu Lin, Zhengda Zhou, Zhiyuan Zhao, Tianrui Wan, Yilun Ma, Junyu Gao, and Xuelong Li. Webuibench: a comprehensive benchmark for evaluating multimodal large language models in webui-to-code. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 15780–15797, 2025.

Gitte Lindgaard, Gary Fernandes, Cathy Dudek, and Judith Brown. Attention web designers: You have 50 milliseconds to make a good first impression! *Behaviour & information technology*, 25 (2):115–126, 2006.

Zimu Lu, Yunqiao Yang, Houxing Ren, Haotian Hou, Han Xiao, Ke Wang, Weikang Shi, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Webgen-bench: Evaluating llms on generating interactive and functional websites from scratch. *Advances in Neural Information Processing Systems*, 38, 2026.

Nishi Mehta, Swathi Alse, Himani Kumawat, Yue Yu, and Pratik Jayarao. Learning to detect ui principle violations via reinforcement learning. *arXiv preprint arXiv:2607.20690*, 2026.

Morten Moshagen and Meinald T Thielsch. Facets of visual aesthetics. *International journal of human-computer studies*, 68(10):689–709, 2010.

Arjun Panickssery, Samuel R Bowman, and Shi Feng. Llm evaluators recognize and favor their own generations. *Advances in Neural Information Processing Systems*, 37:68772–68802, 2024.

Keita Saito, Akifumi Wachi, Koki Wataoka, and Youhei Akimoto. Verbosity bias in preference labeling by large language models. *arXiv preprint arXiv:2310.10076*, 2023.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in llm-as-a-judge. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 292–314, 2025.

Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. Design2code: Benchmarking multimodal code generation for automated front-end engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3956–3974, 2025.

Jeremiah D Still. Web page visual hierarchy: Examining faraday's guidelines for entry points. *Computers in Human Behavior*, 84:352–359, 2018.

Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE transactions on image processing*, 27(8):3998–4011, 2018.

Alexandre N Tuch, Eva E Presslaber, Markus Stöcklin, Klaus Opwis, and Javier A Bargas-Avila. The role of visual complexity and prototypicality regarding first impression of websites: Working towards understanding aesthetic judgments. *International journal of human-computer studies*, 70 (11):794–811, 2012.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, et al. Large language models are not fair evaluators. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9440–9450, 2024.

Jason Wu, Yi-Hao Peng, Xin Yue Amanda Li, Amanda Swearngin, Jeffrey P Bigham, and Jeffrey Nichols. Uiclip: a data-driven model for assessing user interface design. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–16, 2024a.

Jason Wu, Eldon Schoop, Alan Leung, Titus Barik, Jeffrey P Bigham, and Jeffrey Nichols. Uicoder: Finetuning large language models to generate user interface code through automated feedback. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7511–7525, 2024b.

Jiajun Wu, Jian Yang, Tuney Zheng, Wei Zhang, Haowen Wang, Yihang Lou, and Xianglong Liu. Htmlcure: Turning browser experience into state guided repair for interactive html. *arXiv preprint arXiv:2605.26807*, 2026.

Bang Xiao, Lingjie Jiang, Shaohan Huang, Tengchao Lv, Yupan Huang, Lei Cui, Furu Wei, et al. Code aesthetics with agentic reward feedback. In *International Conference on Learning Representations*, volume 2026, pp. 49269–49305, 2026.

Jingyu Xiao, Ming Wang, Man Ho Lam, Yuxuan Wan, Junliang Liu, Yintong Huo, and Michael R Lyu. Designbench: A comprehensive benchmark for mllm-based front-end code generation. *arXiv preprint arXiv:2506.06251*, 2025.

Tianyi Xiong, Xiyao Wang, Dong Guo, Qinghao Ye, Haoqi Fan, Quanquan Gu, Heng Huang, and Chunyuan Li. Lllava-critic: Learning to evaluate multimodal models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13618–13628. IEEE, 2025.

Kai Xu, YiWei Mao, XinYi Guan, and ZiLong Feng. Web-bench: A llm code benchmark based on web standards and frameworks. *arXiv preprint arXiv:2505.07473*, 2025.

Zhen Yang, Wenyi Hong, Mingde Xu, Xinyue Fan, Weihan Wang, Jiale Cheng, Xiaotao Gu, and Jie Tang. Ui2codeˆ n: Ui-to-code generation as interactive visual optimization. *arXiv preprint arXiv:2511.08195*, 2025.

Sukmin Yun, Haokun Lin, Rusiru Thushara, Mohammad Q Bhat, Yongxin Wang, Zutao Jiang, Mingkai Deng, Jinhong Wang, Tianhua Tao, Junbo Li, et al. Web2code: A large-scale webpage-to-code dataset and evaluation framework for multimodal llms. *Advances in neural information processing systems*, 37:112134–112157, 2024.

Chenchen Zhang, Yuhang Li, Can Xu, Jiaheng Liu, Ao Liu, Changzhi Zhou, Ken Deng, Dengpeng Wu, Guanhua Huang, Kejiao Li, et al. Artifactsbench: Bridging the visual-interactive gap in llm code generation evaluation. *arXiv preprint arXiv:2507.04952*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Haonan Zhu, Elad Hirsch, Alexandria Minetti, Allison Nulty, and Purvanshi Mehta. Taste: A designer-annotated multi-dimensional preference dataset for ai-generated graphic design. *arXiv preprint arXiv:2605.20731*, 2026.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. Judgelm: Fine-tuned large language models are scalable judges. In *International Conference on Learning Representations*, volume 2025, pp. 51257–51296, 2025.

APPENDIX

A THE FULL VENUS-CANON

All ten dimensions at all six grades. Grade 5 is flawless and distinctive, 4 is negligible or no defect, 3 is a visible minor defect, 2 is a major defect, 1 is present but unusable, and 0 is not implemented. Semicolons separate the individual checks that make up a grade.

Layout

5: Vertically adjacent sections are spaced identically or in a clearly regular rhythm; columns in a row share a bottom alignment; the content area keeps equal margins on both sides and does not run to the edge.
4: Major sections are well aligned and spacing is broadly even, with no conspicuous large or small gaps.
3: Most of the page is orderly, but one or two gaps are visibly larger or smaller than the rest.
2: The whole page uses a single layout treatment and is highly repetitive; content is noticeably skewed to one side; the layout is too plain, with large empty margins; the layout is severely crowded; spacing is seriously inconsistent in several places.
1: Elements physically overlap and occlude each other, with text on a button or a navigation bar covering content; several layout schemes coexist and conflict; content is clipped by its container; a slide-deck request produces a long-scroll layout.
0: The browser displays no layout at all.

Typography

5: At least three visible size levels for display headings, subheadings, and body; headings carry no alignment-breaking punctuation; the display heading is at least twice the body size; important phrases contrast in weight; lines are clearly separated.
4: Headings and body text are clearly distinguished in size; bold marks emphasis; leading is normal and not crowded.
3: The heading is visibly larger than the body but not by enough, and the focus is not apparent within three seconds; or the page is uniform in weight; paragraph wrapping breaks in illogical places.
2: Size, weight, and leading do not separate the levels; headings and body are nearly the same size; items at the same level differ in size; alignment schemes are mixed; or the page is a solid block of text.
1: Large amounts of text are too small to read; or text colour is too close to the background for the core content to be legible; text fails to render correctly.
0: There is no content and therefore no typography.

Colour

5: At most two primary colours; the secondary colour is well defined; a single accent appears only at the decisive position; all elements of the same type are identically coloured; no colour sits outside the primary system.
4: Primary and secondary colours are clear; at most three or four colours in total; the tone is harmonious; there is no clashing pair.
3: There is an evident primary colour, but one or two colours do not belong to its system; or one or two of the primary, secondary, and accent roles are missing without making the page ugly; or the page reads as undesigned for want of colour.
2: Missing roles make the palette ugly; more than five colours that do not harmonize; elements at the same level are coloured inconsistently; there are clashing pairs.
1: Text and background are nearly the same colour, so core content cannot be read.
0: A browser fallback page is displayed and there is no palette.

Role definitions. The primary colour is the largest and most frequent and sets the overall impression. The secondary colour supports it on card backgrounds and section fills, occupying less area than the primary and more than the accent. The accent appears only at the most decisive positions, small in area and highest in salience. Black, white, and grey cannot serve as an accent.

Font

5: One or two families, not the system default; mixed CJK and Latin text is balanced in weight; the family has design character and matches the scene; strong display type is drawn as SVG or image.
4: One or two non-default families; no evident conflict with the page; rendering is correct with no garbled glyphs; the treatment is somewhat plain.
3: A system default family, safe but without character; or the family does not conflict with the scene but

shows no deliberate choice; there is no type treatment beyond colour.
2: Three or more visibly different families on one page; or the family conflicts with the scene, as when a serious financial page uses a cartoon handwriting face.
1: Font loading fails and produces garbled or tofu glyphs; or the page falls back entirely to the browser default, against the design intent.
0: There is no text.

Imagery

5: Sharp with no compression noise and no broken sources; all images share one visual style; every image relates directly to the subject, with no filler.
4: Sharp and undistorted with no broken sources; content relates indirectly to the subject; styles are broadly consistent.
3: Sharpness is mediocre with mild compression; one or two images differ in style; or some images feel weakly related to the subject.
2: Images are stretched or squashed; or there are watermarks or visible artifacts; or styles differ sharply, mixing photography with cartoon illustration.
1: Images fail to load and show a broken-image icon; or unreplaced placeholders remain, such as `picsum.photos` or `placehold.it`.
0: There are no images.

Background

5: Adjacent sections differ visibly in background colour or lightness, so the sectioning is legible; there is an evident decorative treatment; gradients transition smoothly with no banding; foreground content appears to lift off the background.
4: Sections are distinguished, for instance alternating white and light grey; gradients are smooth; the whole is clean.
3: One background colour throughout, with no visual sectioning from top to bottom, but nothing that causes a problem.
2: Adjacent sections clash in colour; or gradients show hard bands; or the background is so busy that content sinks into it.
1: Background and text are nearly the same colour, so content disappears.
0: The page has crashed and there is no background.

Elements

5: Icon stroke weights are identical across the page, from one library; icons in the same layout are identical in size; brand logos match the real logos; decorative shapes stay within the primary colour system; SVG artwork is of high quality.
4: Icons are stylistically consistent and from one library; sizes are broadly equal; real brand logos carry their distinguishing features; SVG artwork is ordinary.
3: Icons are mostly consistent, but one or two differ visibly in stroke weight or style; or one icon repeats throughout; brand logos are replaced by text; SVG artwork is crude.
2: Icons come from several libraries with differing weights and styles; or emoji are used in place of designed icons; icons contain drawing errors.
1: All icons fail to load and show blanks or question-mark boxes.
0: No decorative elements are used.

Components

5: All primary buttons share corner radius, height, colour, and shadow; there is a distinctive styling treatment; primary and secondary buttons are visually separated, filled versus outlined; cards in a row share a height and an internal padding.
4: Components of the same class are consistent with no outliers; the styling is plain; primary and secondary actions are distinguished; card structure is clear and uncrowded.
3: Components of the same class are broadly consistent, but one or two details differ, such as one button with a visibly larger radius; or buttons are flat rectangles with no detail, but consistent.
2: Components of the same class differ in size, colour, or radius across the page; or an unstyled native browser input appears.
1: A primary button or the navigation bar fails to render; or a key component is visually broken.
0: There are no components.

Interaction

5: Every clickable element shows at least two simultaneous visual changes on hover; pressing gives evident tactile feedback through darkening or slight scaling; form fields change border colour on focus; the cursor becomes a pointer on hover.
4: Primary buttons and navigation links change colour on hover; form fields highlight their border on focus; the cursor becomes a pointer.
3: Some elements respond but coverage is incomplete, as when primary buttons have hover states and cards and links do not; or the cursor does not become a pointer.
2: The page has almost no hover feedback; clicking produces no visual confirmation.
1: Clicking breaks the layout; or primary buttons do not respond at all.
0: There is no interaction.

Motion

5: Animation is complete, varied, and holds attention; elements within one section enter at staggered times, producing a sequential reveal; elements settle with no jitter; a single animation lasts between 0.3 and 0.8 seconds.
4: Animation is complete but basic; major sections have entrance animations; scroll-triggered elements appear in order; motion is smooth with no stutter.
3: Animation exists, but all elements in a section appear at once with no stagger, and the repertoire is limited to a fade; or single animations run longer than one second.
2: Animation is exaggerated and impedes reading; or animations run longer than two seconds; the implementation is incomplete.
1: Animation makes elements vanish or flicker, leaving content missing once it completes; or it causes severe stutter.
0: There is no motion.

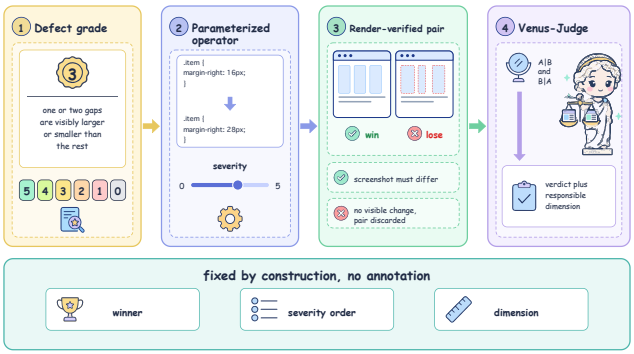

Figure A1: Comparison pairs are constructed, not newly annotated (§3.3). Each defect grade of each static dimension becomes a parameterized operator; applying it to a designer-reviewed page yields a counterfactual worse along one intended dimension by a known amount. A render check discards pairs whose edit is invisible or whose side effects touch another dimension, so the winner, the severity order, and the responsible dimension follow from the construction.

B SCORING PROMPTS

Seed screening

Designers screen generated candidates against VENUS-CANON (§3.2). The instrument is the standard itself, presented one dimension at a time with the six anchors visible. The reviewer records a grade per dimension and a keep or reject decision, and a candidate becomes a seed only if no static dimension falls below the minor-defect grade.

VENUS-JUDGE inference prompt

```
You are given two screenshots of web pages, A and B, generated for
the same requirement.  Judge which page has the worse visual design
quality, and name the single design dimension that best explains the
```

```
difference.
Dimensions: layout, typography, colour, font, imagery, background,
elements, components.
Answer with JSON only:
{"worse": "A" | "B", "dimension": "<one of the eight>"}
```

Dynamic observation instruction

```
Operate the page as a user would. For each action, record the visual
state immediately before and immediately after. Cover: hover on
every clickable element; focus on every form field; press on every
primary button; a full scroll from top to bottom. Report, per action,
whether a visual change occurred and what changed.
```

C PAGE COMPARISONS

Figure A2 shows the same WebGen-Bench tasks answered by Qwen3.6-35B-A3B and by VENUS-CODER, with the full instruction of each task reproduced below it. Each tile is the capture that the appearance judge scored, and the printed grade is that judge's score for the page. We show three cases; the full set of captures is released with the evaluation harness.

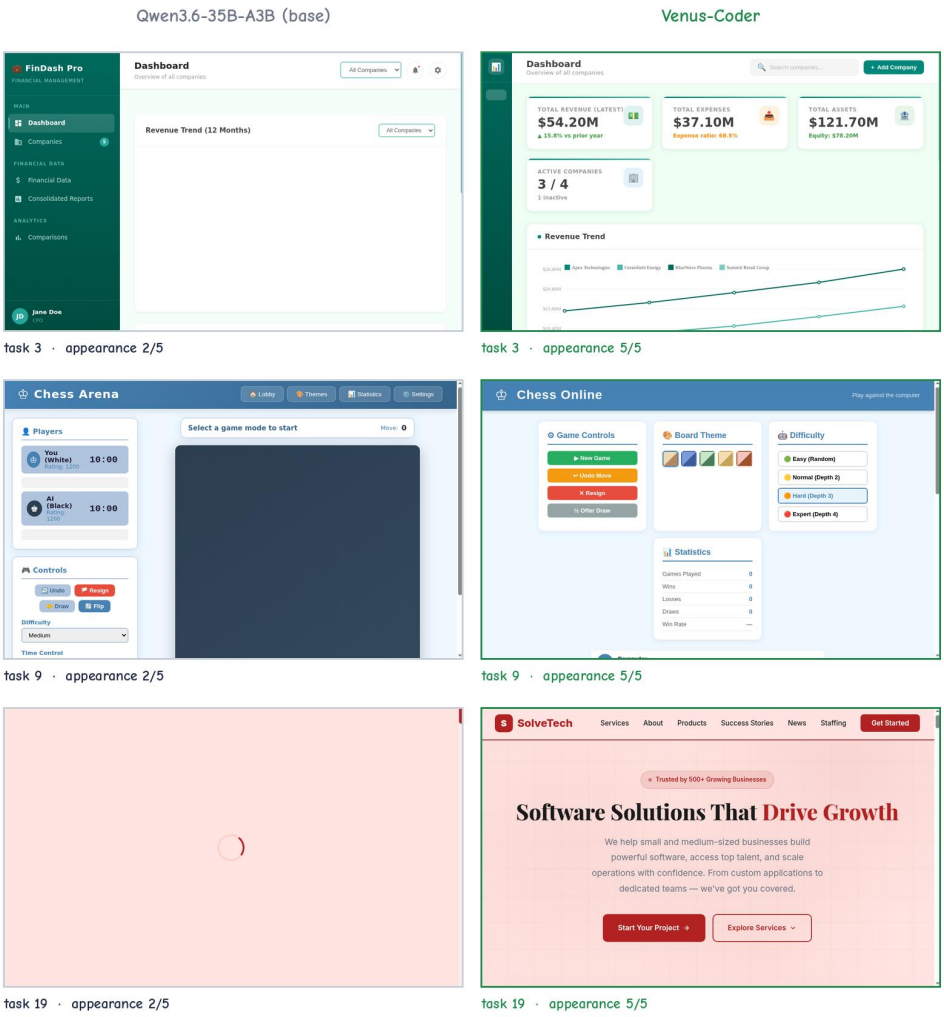

Figure A2: The same WebGen-Bench task answered by Qwen3.6-35B-A3B (left) and by VENUS-CODER (right). Tiles are the captures the appearance judge scored; the printed grade is the judge's score for that page. The instructions are given below.

WebGen-Bench task 3

Please implement a multi-company dashboard for managing and displaying financial data from multiple companies. The dashboard should be able to collect and display financial information from each company, provide consolidated reports, and support cross-company comparisons and reporting. Users should be able to browse financial data from each company, view consolidated reports, and perform financial management and reporting. Apply mint cream as the background; style all components with teal.

WebGen-Bench task 9

Please implement a chess web application where users can play chess online. The application should have functionalities for creating new games, joining existing games, moving pieces, undoing moves, and resigning. Users should be able to click buttons to choose from different board themes and difficulty levels, and view game history and statistics. Style the layout with alice blue in the background and steel blue for the interface.

WebGen-Bench task 19

Please create a software solutions website to provide information to prospective customers. The website should have content that introduces software solutions and staffing business, including product and service descriptions, industry news and trends, customer cases and success stories, etc. The website should be able to attract the attention of small to medium-sized businesses, and the content should be easy to understand and engaging. Users should be able to browse the website, search for relevant information or product descriptions, read articles and news, and contact the website owner for more information. Configure all backgrounds with misty rose and all components with firebrick.

D ARTIFACTSBENCH DELIVERABLE TYPES

ArtifactsBench is scored as a single web-artifact benchmark, but its 1,825 tasks do not all target a web page. Labelling each task by the deliverable its prompt asks for gives two groups: 1,691 tasks (92.7%) whose target is a self-contained page that a browser renders, HTML or SVG, and 134 (7.3%) whose target needs a platform runtime or a build step and does not render as a standalone page. Figure A3 shows the split. The non-HTML group is dominated by WeChat mini-programs and quick-apps, with a tail of native or mobile apps and framework projects such as Vue and React. We score the two groups separately under the same judge model and the same last-markup-block extraction rule.

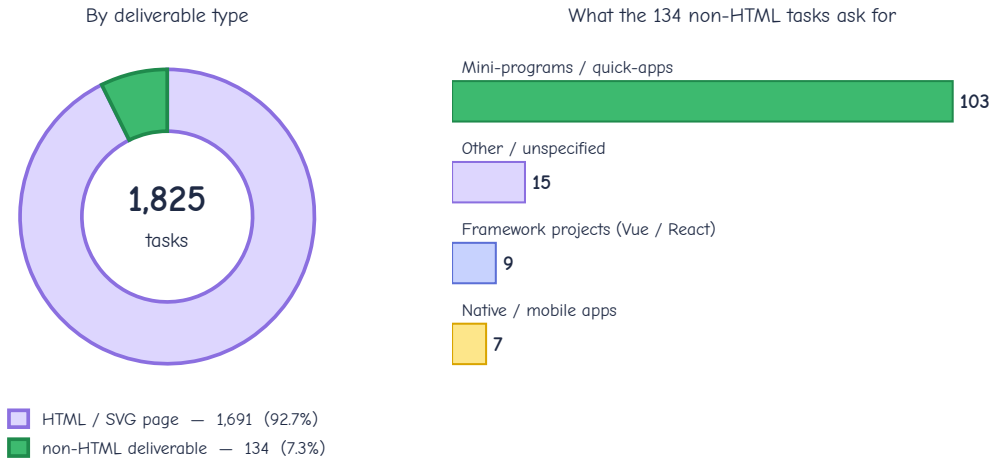

Figure A3: ArtifactsBench by deliverable type. Left: 1,691 of 1,825 tasks target a self-contained web page, 134 do not. Right: what those 134 non-HTML tasks ask for.

Non-HTML task prompts (verbatim)

Create a WeChat mini-program for playing Tetris.

Write a media player frontend using Flutter.

Generate an SPFx + Vue project.

Help me develop a React app that is compatible with both Android and iOS.

Table A1 gives the per-group scores. VENUS-CODER ranks first on the HTML group and ties GPT-5.5 for first on the non-HTML group, and it improves over the base it was trained from on both, by 10.96 on the HTML group and 11.84 on the non-HTML group. The corpus is almost entirely single-page HTML, so the gain on mini-program and framework tasks is transfer rather than memorised templates: the model carries front-end layout and visual judgement across deliverable types instead of reproducing pages it has seen.

Table A1: ArtifactsBench by deliverable type (§D): mean score on the HTML/SVG group (1,691 tasks) and the non-HTML group (134), same judge and extraction as Table 2. **Best** in each column; VENUS-CODER ties GPT-5.5 on the non-HTML group.

System	HTML / SVG (1,691)	non-HTML (134)
VENUS-CODER	**64.97**	**58.90**
DeepSeek-V4-Pro	64.80	52.64
GLM-5.2	63.33	53.46
GPT-5.5	62.84	**58.90**
Gemini 3.6 Flash	61.05	50.29
Claude Opus 4.8	57.18	53.08
Qwen3.6-35B-A3B	54.01	47.06
AesCoder	39.10	35.96

Figure A4 shows renders from both groups, Qwen3.6-35B-A3B against VENUS-CODER, with the judge's per-item score. The pattern is the same in either group: VENUS-CODER produces a fuller, more organised interface where the base leaves a thin or broken page.

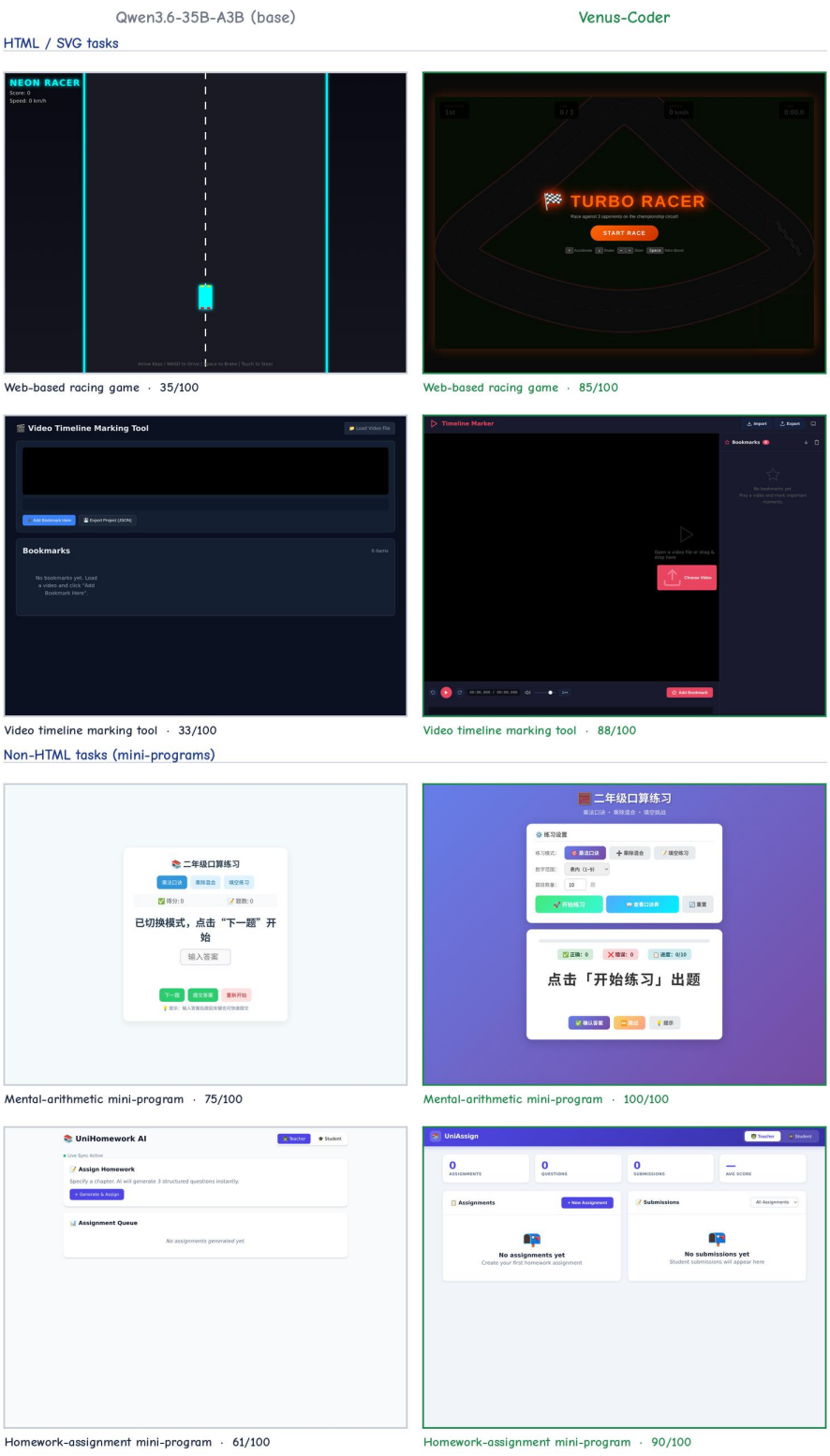

Figure A4: ArtifactsBench renders, Qwen3.6-35B-A3B (left) and VENUS-CODER (right): two HTML tasks and two non-HTML mini-program tasks. The printed grade is the judge's per-item score on a 0–100 scale.

E Alignment with Human Judgement

Two blind studies with senior designers check that VENUS tracks real human judgement at both ends of the pipeline: the filter, and the pages it produces.

The filter. Three independent senior designers each compared 120 operator-generated pairs, balanced over the eight static dimensions and blind to how each pair was built. On the pairs a designer judged decisively, VENUS-JUDGE chose the same page on 75.4–78.3% of them (77.2% on average) and matched the designer majority 80.6% of the time, counting an order-inconsistent verdict as a disagreement; on the pairs where both presentation orders agreed, the rate is about 86%. Agreement follows the designers' certainty and the perceived gap (Figure A5): 80.2% at high confidence against 68.8% at medium, and 80.4% where the gap was marked obvious against 63.0% where it was slight. The three designers agree with one another on 85.7% of shared decisive pairs, slightly above the judge's agreement with them.

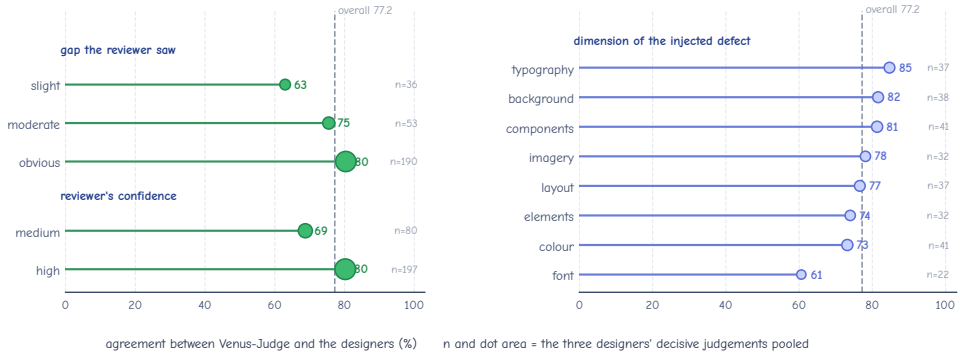

Figure A5: Agreement between VENUS-JUDGE and the three independent senior designers on operator-generated pairs each designer judged decisively (an order-inconsistent verdict counts as disagreement), grouped by the perceived gap, by confidence, and by the dimension of the introduced defect. Each row's count n and dot area pool the three designers' decisive judgements; the dashed line is the overall rate.

The generated pages. Three independent senior designers each compared 50 pages generated by VENUS-CODER against the pages its base Qwen3.6-35B-A3B produced for the same requirement, side by side in a randomized, blinded order; each chose the better-designed page or marked the pair too close to call. On average they preferred VENUS-CODER on 74.8% of the pairs they judged decisively, and by majority vote on 77.6% of the 49 pairs with a majority, against 22.4% for the base; every designer favoured VENUS-CODER on a clear majority of pairs. The corpus thus brings a gain designers see on the generated pages, not only a rise in the judge's score.

F Implementation Details

VENUS-JUDGE. The judge is a Qwen2.5-VL-7B-Instruct backbone fine-tuned with the vision encoder frozen, taking two screenshots per example at a longest side of 896 px. The training set combines the compiled comparisons from §3.3 and the de-noised natural preferences described there; WebDevJudge items are excluded from every training source. The inference prompt is given in Appendix B. Throughput on a single H100 is a median of 0.55 s per comparison, about 3,200 pairs judged in both orders per GPU-hour.

WebDevJudge protocol. Every pair carries a natural human preference. Every judge sees each pair in both presentation orders. *Strict* counts a pair as correct only if the verdict is correct in both orders, and *Consistency* is the share of pairs on which the same page wins in both orders. The reference judges are run through the same prompt and the same image pipeline.

ArtifactsBench. All 1,825 tasks are answered with a single response, rendered in the benchmark's own headless-browser harness, captured as a temporally spaced screenshot sequence, and scored

against the per-task checklist by Gemini 3.1 Pro. The reported score is the benchmark's normalized score scaled to 100.

WebGen-Bench. Each of the 101 site instructions is answered with a single self-contained HTML page, deployed locally, and driven by the benchmark's WebVoyager agent over its 647 test tasks; functional accuracy is the share of tasks the agent completes. Appearance is scored from a full-page screenshot on the benchmark's 1 to 5 scale by the same judge model.

G CORPUS ABLATION

The three fine-tuned rows of Table 3 start from the same Qwen3.6-35B-A3B checkpoint and are trained with the same recipe, the same hyperparameters, and the same number of epochs; the only variable is which pages of one candidate pool become the training corpus. The random subset is a uniform draw from the unfiltered pool with the sample size fixed to the size of the VENUS-JUDGE-filtered corpus, so it isolates the selection criterion from the amount of supervision. The unfiltered row takes the pool in full and is therefore the largest corpus of the four. All four rows are evaluated under the harness, prompts and judge model of §4.1.

Table A2: Corpus ablation, broken out. ArtifactsBench on the full set and on its HTML and non-HTML subsets (Appendix D); WebGen-Bench functional accuracy and appearance on the 1 to 5 scale. Same base, same recipe, same pool for all four rows.

	ArtifactsBench			WebGen-Bench	
Training corpus	**Full**	**HTML**	**non-HTML**	**Func.**	**Appear.**
none (base)	53.50	54.01	47.06	46.2	3.54
random subset	59.22	59.99	49.53	34.6	3.07
whole pool	61.99	62.47	56.00	49.8	3.70
VENUS-JUDGE-filtered	**64.52**	**64.97**	**58.90**	**57.0**	**4.26**

Table A2 breaks the same runs out by ArtifactsBench subset. The ordering of the four corpora is the same on every subset, so the gain from screening is not an artifact of the deliverable mix; the non-HTML subset, where the target needs a platform runtime rather than a browser, moves furthest, from 47.06 at the base to 58.90.