IEDCO Motion

Visual motion tuning system for React and Angular interfaces

EDITION

Revised edition
October 22, 2026

AUTHOR

Rob Vance

PRODUCT

IEDCO Motion for React and Angular (beta)

Motion as a system

Visual tuning for React and Angular interfaces

EDITION	Revised edition • October 22, 2026
AUTHOR	Rob Vance
SOURCE MANUSCRIPT	Based on IEDCO Motion: A visually tuned motion system for React and Angular (draft, October 22, 2026).
PRODUCT	IEDCO Motion for React 1.4.2 and for Angular 1.0.0 (beta)
SOURCES CHECKED	Fourteen references, each tested and read on October 22, 2026 (Appendix B)

> **Attribution note:** IEDCO Labs publishes both IEDCO Motion and this paper, so the paper is not an independent evaluation. Product facts come from the product's own documentation and images. The cited sources support the general design rationale only; none of them tests IEDCO Motion, and the product's performance, authoring-time, and maintenance claims remain hypotheses until measured.

ABSTRACT

Most applications acquire animation one component at a time, and the result is an interface that answers the same timing questions in many different ways. This paper treats interface animation as a design system rather than a collection of effects, using IEDCO Motion, a visually tuned motion system for React and Angular, as the case study. It separates animation mechanisms from the motion language that governs them and draws on research into animation authoring and animated transitions, rendering guidance, framework documentation, and accessibility standards. The resulting approach combines presets with detailed controls, previews every timing in context, organizes decisions into foundations, behaviors, and exceptions, delegates suitable effects to the browser, and applies one reduced-motion policy across the interface. The central argument is that the product's value lies in connecting visual judgment with reusable implementation, and that claims of smaller footprint, faster authoring, or easier maintenance require product-specific measurement. Priorities for later versions and the evidence each claim still needs close the paper.

Keywords: interface animation; motion design systems; visual authoring; React; Angular; Web Animations API; rendering performance; reduced motion; accessibility

1.

Introduction: animation capability without a motion system

Most applications can animate an element long before the application develops a consistent approach to motion. A developer adds a transition to a button. Another developer adds an entrance effect to a panel. A modal receives its own duration and easing curve. Each interaction may look acceptable on its own, yet the application gradually accumulates different answers to the same questions.

Those questions concern how quickly an element should respond, how far an element should move, whether an exit should feel as deliberate as an entrance, and which changes should animate at all. An animation library supplies mechanisms for producing motion. Establishing a coherent motion language remains a separate design and engineering task.

IEDCO Motion addresses that task by giving teams a shared place to define and inspect motion. The system supplies structure, and the person using the system evaluates what the interface actually feels like. The value of the product therefore lies in connecting implementation with judgment.

This paper is a design rationale combined with a research synthesis; it is not an independent evaluation of the product. Section 2 defines the terms used throughout and describes the product. Sections 3 and 4 explain why visual tuning and a shared system matter. Sections 5 and 6 describe the React and Angular integrations. Section 7 sets out the product's competitive position, and Section 8 addresses accessibility and interaction correctness. Sections 9 through 11 present priorities for later versions, the remaining evidence gaps, and the boundaries of the evidence.

2.

Conceptual foundations

2.1

Mechanism and motion language

A mechanism is the means by which an element changes over time: a CSS transition, a keyframe animation, the Web Animations API, or a library function. A motion language is the set of decisions that governs those mechanisms across an interface: which durations and curves apply, how far elements travel, how sequences are ordered, and which changes animate. A team can possess many mechanisms and still lack a motion language. The distinction explains why adding a library does not, by itself, make an interface move consistently.

2.2 Key terms

TERM	MEANING IN THIS PAPER
Motion system	A shared set of motion decisions, applied across an interface from one configuration.
Preset	A named starting configuration that establishes the overall character of the motion.
Foundations	Reusable values such as durations, easing curves, and travel distances.
Behaviors	Interaction patterns such as entrance, exit, feedback, and expansion.
Exceptions	Local adjustments that a particular component requires.
Motion Board	The product's authoring page, where each option sits beside a live preview.
Motion Panel	A development-only panel that adjusts the same settings over a project's own pages.
Reduced motion	The operating-system preference, exposed to CSS as `prefers-reduced-motion`, that asks for less movement.

2.3 Types of evidence

The sources cited in this paper differ in kind, and their weight differs accordingly. Product facts form a separate category, because the product's own documentation describes the product without independent verification.

CATEGORY	SOURCES	HOW THE EVIDENCE IS USED
Research	[1–3]	Findings on animation authoring and animated transitions, within the tasks each study examined. One source is a preprint.
Engineering and framework guidance	[4–7]	Official guidance on rendering and on the React and Angular lifecycles. This guidance does not test IEDCO Motion.
Accessibility standards	[8, 9]	W3C explanations of a success criterion and a sufficient technique. These documents are informative, not normative.
Competitor documentation	[10–13]	Vendor descriptions of available capabilities, not independent comparative research.
Product documentation	[14]	The product's own description, published by IEDCO Labs. These facts have not been independently audited.

2.4 The product

According to its documentation, IEDCO Motion is a skill for Claude that adds one consistent motion system to a React project. Claude installs a small kit, wires the kit into a project's pages one at a time, and checks each change frame by frame on a production build. The system covers loading states, staggered entrances, page and tab exits, list motion after a sort or filter, and side panels that push the page. [14]

Tuning takes place in the Motion Board, where each of the 58 options sits beside a live preview, and a development-only panel adjusts the same settings over the project's own pages. The kit is written into the project as plain files, uses CSS and the browser's animation API rather than a separate animation library, and limits motion to short fades for visitors who have turned on the reduced-motion preference. The Angular edition, currently a beta, uses the same Motion Board and an identical settings export. [14]

---

3. Visual tuning

3.1 Why timing values are difficult to judge in isolation

Animation is difficult to assess through isolated numbers. A duration of 200 milliseconds says little about whether a transition will feel appropriate, because perceived speed also depends on travel distance, easing, the size of the moving element, surrounding activity, and the action that triggered the transition.

A small menu opening beneath a button and a large panel crossing half the viewport illustrate the problem. Giving both the same duration does not necessarily create consistency. The larger movement may feel abrupt, while the smaller movement may feel restrained. Visual tuning makes these relationships observable: a team can compare alternatives, inspect sequencing, and adjust the result in context.

3.2 Research on animation authoring

Research on animation-authoring tools supports combining several ways of working. In an ideation study of three authoring paradigms (keyframe animation, procedural animation, and presets and templates), Thompson and colleagues found an overall preference for keyframe animation, but also found that preferences depended on the transition and the authoring task; the authors therefore recommended that authoring tools combine all three paradigms. [2] The study concerned animated data graphics, so applying the finding to product interfaces is a design inference rather than a tested outcome.

Research on AI-assisted animation points in the same direction. Keyframer combines natural-language prompts with inline previews of the rendered animation and editors for direct adjustment, and a study with 13 novices and experts examined how participants refined animations through successive prompts and edits. [1] The study supports pairing generated animation code with rendered previews and direct editing. The study does not test IEDCO Motion or show that the product saves authoring time.

3.3

The Motion Board

The Motion Board and the Motion Panel implement this authoring model. Presets named Snappy, Default, and Calm provide starting points, so that a project does not begin with an empty configuration. In the product's own images, each preset scales a global speed and sets an entrance travel distance (Figure 1). [14]

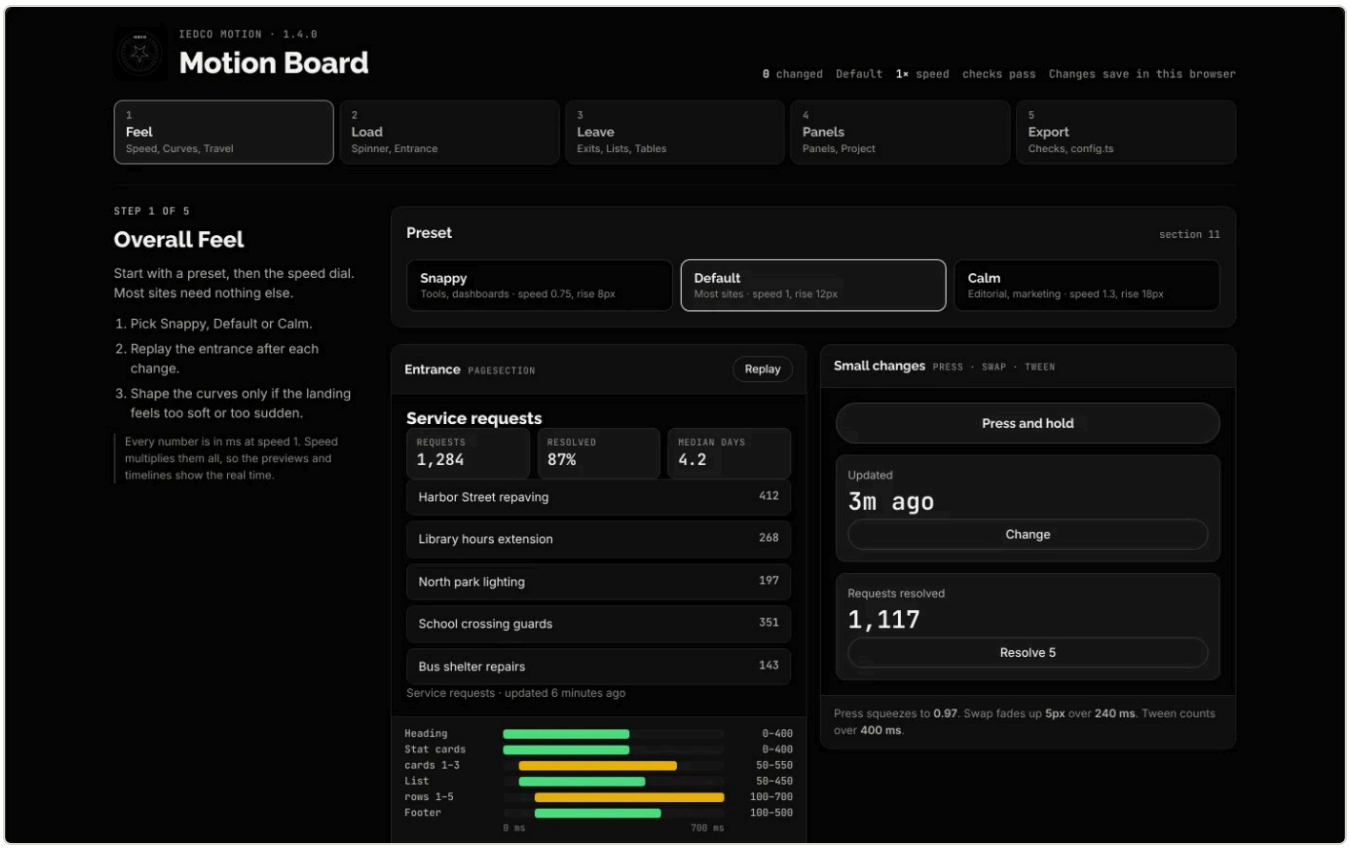


FIGURE 1 The Motion Board's first step, Overall Feel. The Snappy, Default, and Calm presets set a global speed and an entrance travel distance, and each change replays beside a live preview. Product image, IEDCO Motion for React [14]. ADDED IN THIS EDITION

A timeline preview exposes delays, overlap, and ordering that are easy to overlook when settings are scattered across components. The Load step, for example, draws the loading spinner, its fade, and each staggered card entrance on a single time axis (Figure 2).

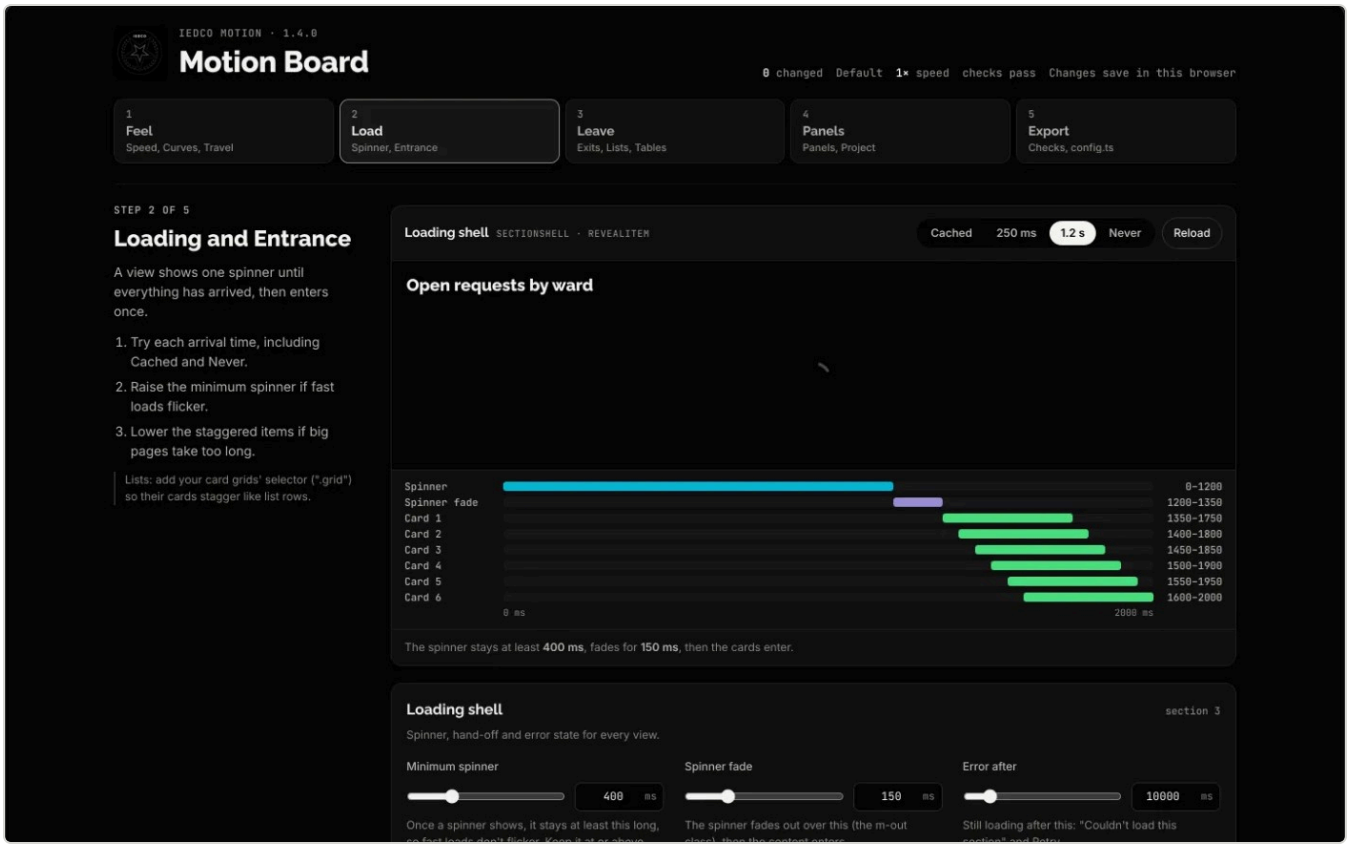


FIGURE 2 The Load step draws the spinner, its fade, and each staggered card entrance on one timeline in milliseconds, so delays and overlaps can be read directly. Product image, IEDCO Motion for React [14]. ADDED IN THIS EDITION

3.4 Evidence status

The expected advantage of visual tuning is a shorter path from an intended interaction to a reviewed result. Whether that advantage translates into measurable time savings has not been tested. Structured authoring tasks that measure completion time, errors, and revisions would be required to establish it (Section 10).

---

4. Motion as a shared system

4.1 Relationships between interactions

A reusable motion system expresses relationships between interactions. The system can establish that immediate feedback begins promptly, that large transitions receive different treatment from small state changes, and that repeated items enter in a controlled sequence. Those decisions can then guide many components.

This arrangement creates a practical maintenance benefit. Changing the character of an application becomes a deliberate adjustment to the system rather than a search through unrelated styles and component files. The size of that benefit has not been measured (Section 10).

4.2 Foundations, behaviors, and exceptions

The strongest configuration model separates three concerns. Foundations hold the reusable values; behaviors define the interaction patterns built from those values; exceptions record the local adjustments that particular components require. This structure allows consistency without forcing every interaction to move identically.

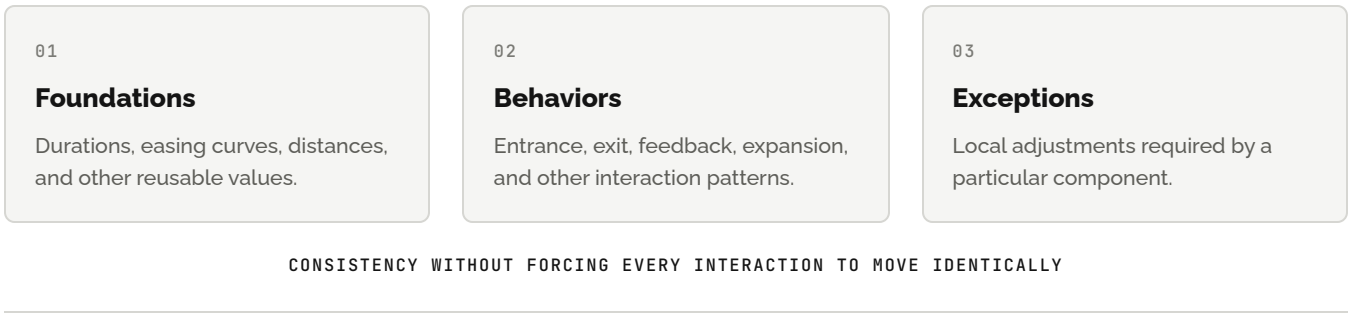


FIGURE 3 The three concerns of the configuration model described in Section 4.2, drawn from the text. ADDED IN THIS EDITION

4.3 Motion that communicates change

A shared system is most useful when its motion carries meaning. Heer and Robertson ran two controlled experiments on animated transitions between statistical graphics and found that animated transitions can significantly improve graphical perception. [3] The finding supports using motion to communicate how one state becomes another. The finding does not establish that animation improves usability in general, that decorative motion offers the same benefit, or that IEDCO Motion improves task performance.

5. Integration with React

5.1 Division of responsibility

React manages application state and component composition, and browser animation mechanisms can handle many of the visual transitions between those states. For suitable effects, CSS transitions, keyframes, or the Web Animations API perform the animation after React establishes the relevant state or element. Continuous animation does not inherently require a React state update for every frame.

5.2 Lifecycle and interruption

The integration still requires careful lifecycle handling. An exit animation requires coordination between the visual departure of an element and the removal of that element. Interrupted animations need defined behavior, and subscriptions and animation handles need cleanup. React's documentation makes cleanup part of a correct integration with any external system, and in development React remounts each component once immediately after the initial mount so that missing cleanup becomes

visible. [7] An animation system must therefore account for setup, dependency changes, and unmounting, so that repeated mounting or interrupted interactions do not leave obsolete effects running.

These requirements explain why a guided system can offer more value than a folder of CSS effects. The useful work includes deciding where animation belongs, integrating that animation with the component, and checking that the interaction remains correct when users act quickly. The product's Leave step illustrates one such decision: exits run at a fraction of the matching entrance, so that departing content clears quickly (Figure 4). [14]

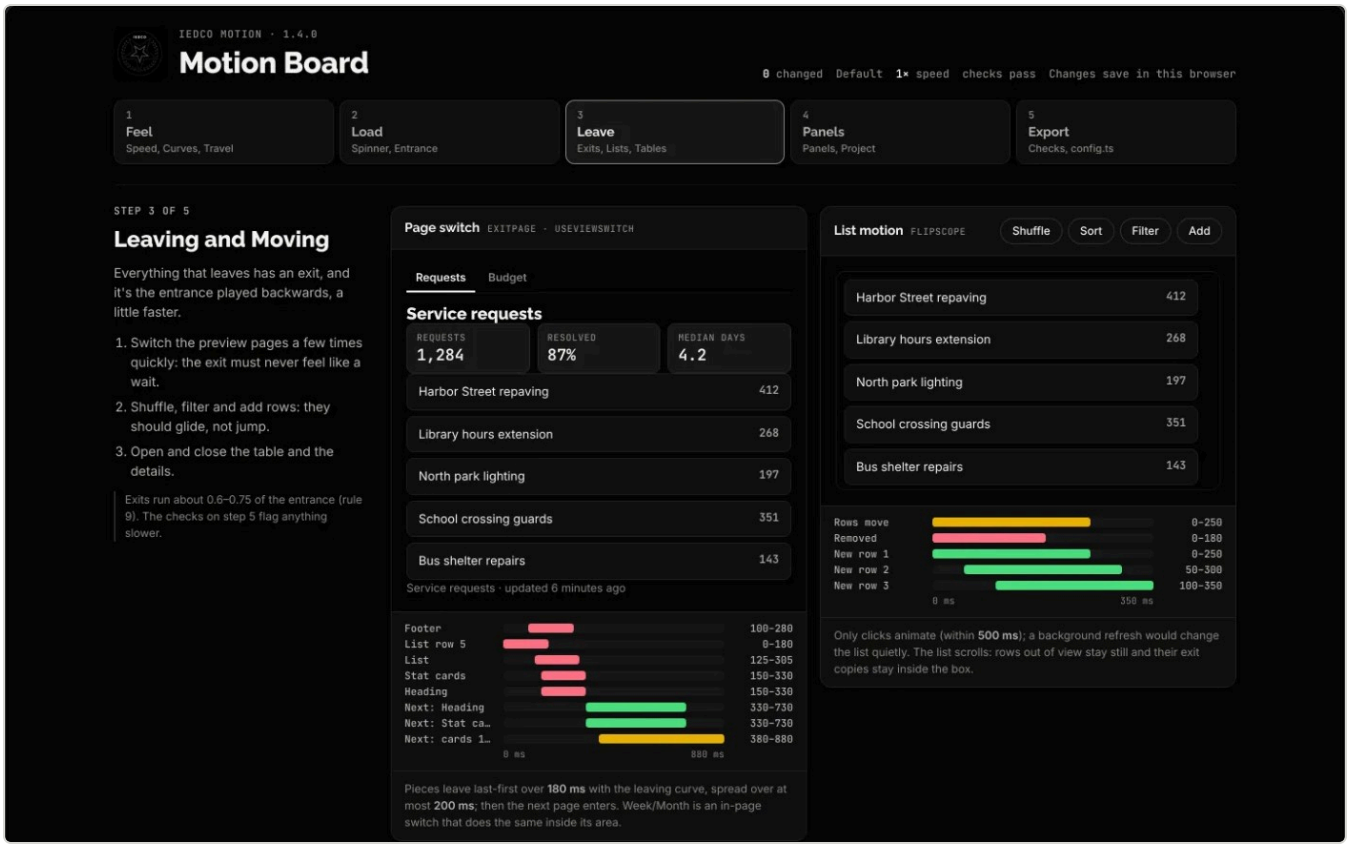


FIGURE 4 The Leave step pairs page exits with list motion after a shuffle, sort, or filter. The board notes that exits run at about 0.6 to 0.75 of the matching entrance. Product image, IEDCO Motion for React [14].
ADDED IN THIS EDITION

5.3 The rendering path

The rendering path also matters. Google's guidance recommends animating `transform` and `opacity` where possible and, before animating any other property, determining that property's effect on the rendering pipeline, because properties that trigger layout or paint add work for the browser. [4] Using native mechanisms alone does not guarantee good performance, and specialist libraries can use the same efficient rendering paths. Profiling remains necessary to verify the behavior of the finished interface.

6. Integration with Angular

6.1 A framework-specific integration

Angular requires its own integration rather than a translation of React component syntax. The motion language can remain consistent across editions, while Angular manages state, templates, dependency injection, and lifecycle through its own conventions.

6.2 Angular's native-oriented direction

Angular's documentation reinforces a native-oriented approach. The `@angular/animations` package is deprecated as of version 20.2, the release that also introduced `animate.enter` and `animate.leave` for entrance and exit animation with CSS. [5, 6] The migration guide adds that removing the legacy package can significantly reduce the size of an application's JavaScript bundle. [5] That statement concerns removal of Angular's own legacy package and is not a comparison between IEDCO Motion and other animation libraries. A native-oriented motion system is therefore compatible with the framework's current direction, although each release of the product still needs an explicit supported-version policy.

6.3 Parity as equivalent behavior

Cross-framework consistency means equivalent behavior and design intent; it does not require identical implementation. A panel should communicate the same relationship in React and Angular, even when the code responsible for mounting, animating, and removing the panel differs.

The product documentation reports that every feature of the Angular beta passes the product's test suite on Angular 21 and 22, including server rendering and zoneless applications, and that the motion matches the React edition frame for frame in comparison tests. The same documentation states that the Angular edition has not yet been verified in a production application. [14]

7. Competitive position

7.1 Against package-based libraries

The clearest advantage of IEDCO Motion is the combination of visual control, reusable decisions, and a delivery model that does not require a separate animation package. The advantage matters most for teams whose needs center on ordinary product interactions: panels, menus, dialogs, entrances, exits, and feedback. For those interactions, a broad animation dependency may provide more capability than the interface requires.

Avoiding that dependency can reduce package-management obligations and give a team more direct ownership of the resulting implementation. Generated application code still has a cost, however, so "no animation package required" must not be presented

as “no runtime code” or “zero overhead.”

Any footprint comparison must also use realistic alternatives. Motion for React relies on tree shaking, so that only the functions a project imports are shipped, and offers a `LazyMotion` component that loads animation features on demand; the vendor states that this approach can reduce the initial bundle to as little as 5 kB. [10, 11] Comparing IEDCO Motion with the entire size of a competing package would therefore be misleading. The appropriate comparison is the production code needed to deliver equivalent behavior.

7.2 Against manually authored CSS

A skilled developer can write efficient transitions directly. Against manually authored CSS, the strongest advantage of IEDCO Motion is the authoring system: a coordinated workflow for choosing, previewing, and maintaining the transitions.

7.3 Where specialist libraries fit

For complex gesture handling, automatic layout transitions, or sophisticated choreography, a mature specialist library may remain the more economical engineering choice. Motion for React, for example, can animate an element’s size and position automatically when its layout changes and can animate shared elements across components. [12]

ALTERNATIVE	STRENGTH OF THE ALTERNATIVE	CLAIMED ADVANTAGE OF IEDCO MOTION
Package-based library	Broad capability, including layout and shared-element animation [12]; tree shaking and lazy loading [10, 11].	No separate animation package; a visual authoring workflow. Footprint advantage unmeasured.
Manually authored CSS	No added tooling; efficient when written by a skilled developer.	A coordinated workflow for choosing, previewing, and maintaining transitions.
Angular’s legacy package	Familiar to existing Angular projects.	Alignment with Angular’s native direction [5]; production use unverified.

7.4 Position statement

IEDCO Motion therefore occupies a specific competitive position: a visually adjustable motion system for teams that want consistent interface behavior with limited additional infrastructure.

8. Accessibility and interaction correctness

8.1 Reduced motion

Motion quality includes what happens when animation is reduced or removed. A reduced-motion experience should preserve state changes, feedback, and access to content. Large translations or scaling effects may need to become immediate changes

or quieter alternatives.

This requirement has an established accessibility basis. WCAG 2.2 Success Criterion 2.3.3, Animation from Interactions, a Level AAA criterion, calls for motion animation triggered by interaction to be capable of being disabled unless the animation is essential. [8] W3C lists the CSS `prefers-reduced-motion` media query as a sufficient technique for that criterion. [9] Honoring the preference does not, by itself, establish conformance with WCAG.

The requirement applies across the system. A central preference is more dependable than expecting each component author to remember an independent exception, and the product documentation states that visitors with the preference turned on receive short fades only. [14] Established competitors already provide reduced-motion controls; Motion for React, for example, offers a `reducedMotion` option and a `useReducedMotion` hook. [13] Accessibility is therefore a baseline requirement and an area for implementation quality, not an exclusive differentiator.

8.2 Interaction correctness

Interaction correctness requires similar attention. Repeated clicks, rapid navigation, interrupted exits, and keyboard focus should remain predictable. A transition that delays input or leaves focus on an element that is no longer available is an incomplete implementation, however attractive the transition may be.

---

9. Priorities for later versions

9.1 Publish reproducible performance measurements

The strongest next step is evidence. A benchmark suite should compare equivalent interactions across IEDCO Motion, a minimal native implementation, and relevant libraries. The suite should report compressed production size, main-thread activity, rendering behavior, and performance on representative lower-powered devices. Development controls should be measured separately from production output.

9.2 Make interruption behavior explicit

Each component should define what happens when an animation reverses or receives another command before completion. A menu reopened during its exit should recover cleanly. A panel should not jump to an obsolete starting position. Lifecycle cleanup should prevent callbacks from acting on removed elements.

9.3 Add motion diagnostics

The authoring tools could identify expensive animated properties, excessive delays, conflicting transforms, missing reduced-motion behavior, and development controls accidentally included in production. These checks would connect visual tuning with

implementation quality. The current Motion Board already ends with a Check and Export step that runs rule checks before the settings are exported (Figure 5); the scope of those checks is not evaluated in this paper.

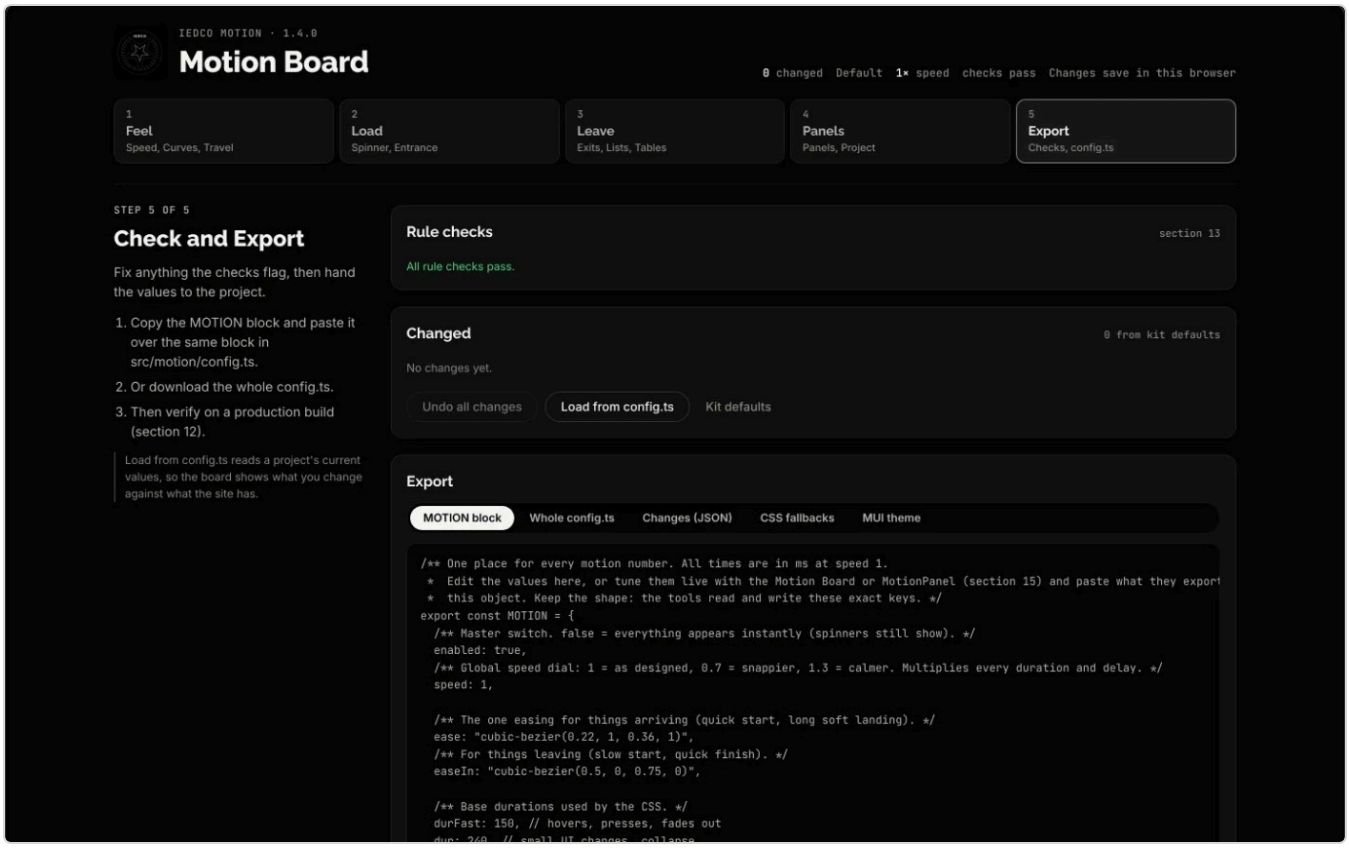


FIGURE 5 The Check and Export step runs rule checks and then exports the settings as one block to paste into the project's configuration. Product image, IEDCO Motion for React [14].
ADDED IN THIS EDITION

9.4 Strengthen framework parity

A shared behavior suite could verify that the React and Angular editions produce equivalent timing relationships, interruption outcomes, and accessibility behavior. Such a suite would make parity testable while allowing each edition to follow its framework's conventions.

9.5 Improve governance and portability

Versioned configuration, documented defaults, and clear local overrides would help larger teams maintain the system. Exportable settings could also make motion decisions easier to review alongside other design-system changes.

10.

Remaining evidence gaps

External research supports the design choices described here, but no cited source establishes that IEDCO Motion is faster, smaller, or easier to use than competing tools. The table lists each product claim and the evidence that would be needed to support it.

PROPOSED CLAIM	EVIDENCE NEEDED
IEDCO Motion ships less code	Equivalent production builds, compressed incremental size, and explicit accounting for generated application code.
Visual tuning saves time	Comparative authoring tasks that measure completion time, errors, and revisions.
Shared settings reduce maintenance effort	Tasks that compare system-wide changes with scattered component settings.
React and Angular behave equivalently	Shared behavior tests covering timing, interruption, lifecycle, and reduced motion, extended to production applications.
IEDCO Motion performs better	Equivalent interactions profiled on representative devices and browsers.

11.

Methodological boundaries

The research cited in Sections 3 and 4 examined animation authoring and animated data graphics, not product interfaces built with IEDCO Motion. Applying those findings to the product is a design inference. The Keyframer study [1] is a preprint, and its participant sample was small.

The engineering, framework, and competitor documentation [4–7, 10–13] describes capabilities and recommendations that may change after October 22, 2026, and should be checked again before any version-specific claim is published. The accessibility documents [8, 9] are informative explanations; the criterion they explain is Level AAA, and meeting it is not the same as conforming to WCAG as a whole.

The product facts [14] come from IEDCO Labs, the publisher of both the product and this paper. The test results reported there have not been independently reproduced, and no IEDCO Motion code was audited for this paper.

12.

Conclusion

The strongest contribution of IEDCO Motion is a workflow that connects visual judgment with reusable implementation. For React and Angular teams focused on everyday interface behavior, that approach can offer a better fit than adopting a broad animation package or maintaining disconnected effects by hand.

The lightweight proposition of the product becomes most convincing when its production output is measured, its lifecycle behavior is tested, and its accessibility rules are enforced consistently. Later versions should deepen that foundation through diagnostics, benchmark evidence, interruption handling, and dependable parity between the two frameworks.

Acknowledgments and contribution statement

Rob Vance is the author of the source manuscript. A research pass dated October 22, 2026 identified the supporting sources and suggested passages used in this edition. IEDCO Labs prepared this edition, which rewrites the manuscript as an impersonal research report; Appendix A records each substantive change. The editorial standards applied in the revision, namely clear pronoun referents, grammatical agreement, a formal introductory register, complete sentences, and precise terminology, follow Dr. Felicia Ledogar's annotations on the companion paper, Human writing voice. Dr. Ledogar did not review this manuscript, and this acknowledgment does not imply that she endorses the paper, the product, or the studies it cites.

References

Grouped by type of evidence. Every link was tested and every page read on October 22, 2026 (Appendix B).

RESEARCH

[1] Tseng, T., Cheng, R., & Nichols, J. (2024; revised 2025). Keyframer: Empowering Animation Design using Large Language Models. arXiv preprint arXiv:2402.06071v2.
https://arxiv.org/abs/2402.06071

[2] Thompson, J., Liu, Z., Li, W., & Stasko, J. (2020). Understanding the Design Space and Authoring Paradigms for Animated Data Graphics. Computer Graphics Forum, 39(3), 207–218 (EuroVis 2020).
https://www.microsoft.com/en-us/research/publication/understanding-the-design-space-and-authoring-paradigms-for-animated-data-graphics/

[3] Heer, J., & Robertson, G. (2007). Animated Transitions in Statistical Data Graphics. IEEE Transactions on Visualization and Computer Graphics, 13(6), 1240–1247 (InfoVis 2007).
https://www.microsoft.com/en-us/research/publication/animated-transitions-in-statistical-data-graphics/

ENGINEERING AND FRAMEWORK GUIDANCE

[4] Basques, K., & Andrew, R. (n.d.). How to create high-performance CSS animations. web.dev, Google. Retrieved October 22, 2026.
https://web.dev/articles/animations-guide

[5] Angular. (n.d.). Migrating away from Angular's Animations package. Angular documentation. Retrieved October 22, 2026.

https://angular.dev/guide/animations/migration

[6] Angular. (n.d.). Animating your applications with animate.enter and animate.leave. Angular documentation. Retrieved October 22, 2026. https://angular.dev/guide/animations

[7] React. (n.d.). Synchronizing with Effects. React documentation. Retrieved October 22, 2026. https://react.dev/learn/synchronizing-with-effects

ACCESSIBILITY STANDARDS

[8] W3C Web Accessibility Initiative. (n.d.). Understanding Success Criterion 2.3.3: Animation from Interactions (WCAG 2.2). Retrieved October 22, 2026. https://www.w3.org/WAI/WCAG22/Understanding/animation-from-interactions.html

[9] W3C Web Accessibility Initiative. (n.d.). Technique C39: Using the CSS prefers-reduced-motion query to prevent motion (WCAG 2.2 Techniques). Retrieved October 22, 2026. https://www.w3.org/WAI/WCAG22/Techniques/css/C39

COMPETITOR DOCUMENTATION

[10] Motion. (n.d.). Reduce bundle size. Motion for React documentation. Retrieved October 22, 2026. https://motion.dev/docs/react-reduce-bundle-size

[11] Motion. (n.d.). LazyMotion. Motion for React documentation. Retrieved October 22, 2026. https://motion.dev/docs/react-lazy-motion

[12] Motion. (n.d.). Layout animation. Motion for React documentation. Retrieved October 22, 2026. https://motion.dev/docs/react-layout-animations

[13] Motion. (n.d.). Accessibility. Motion for React documentation. Retrieved October 22, 2026. https://motion.dev/docs/react-accessibility

PRODUCT DOCUMENTATION

[14] IEDCO Labs. (2026). IEDCO Motion: React 1.4.2 and Angular 1.0.0 (beta). Labs, industrialethic.com. Retrieved October 22, 2026. Product images from the same entry are reproduced in Figures 1, 2, 4, and 5. https://industrialethic.com/#labs

Appendix A. Revision record

This edition rewrites the draft as an impersonal research report. Every claim, qualification, and caveat in the draft is retained, and the draft's product statements are now tied to the product documentation. The table lists each substantive change by draft section.

DRAFT SECTION	THIS EDITION	SUBSTANTIVE CHANGE
Abstract	Title page	Rewritten as a formal abstract with keywords. The attribution note states that the paper is not an independent evaluation.
The problem	1, 2.1	Rhetorical questions restated as a problem definition; mechanism and motion language defined.
None	2.2 to 2.4	Key terms, evidence types, and a product description from the product documentation [14] added.
Why visual tuning works	3	Authoring research [1], [2] added with their limits. Figures 1 and 2 added from the product images. The untested time-saving claim kept as untested.
Why a shared system matters	4	The three concerns drawn as Figure 3. Animated-transition research [3] added with its limits.
How the approach fits React	5	React's cleanup and development-remount guidance [7] cited; Google's rendering guidance [4] cited by name and link. Figure 4 added.
How the approach fits Angular	6	Version 20.2 deprecation and bundle-size statement cited [5], [6] with their scope. Product parity test results [14] added, including the statement that production use is unverified.
Where IEDCO Motion offers an advantage	7	Motion for React capabilities cited [10] to [12]. Comparison table added. "Zero overhead" caution retained.
Accessibility and interaction correctness	8	WCAG 2.3.3 (Level AAA) and Technique C39 cited [8], [9]; competitor reduced-motion controls cited [13].
Priorities for later versions	9	Retained in full. Existing rule checks noted in 9.3 with Figure 5.
None	10, 11	Evidence-gap table and methodological boundaries added from the research handoff.
Conclusion	12	Retained.
None	References	Fourteen references added; the draft named Google, Angular, and Motion guidance without links.

Editorial standards applied

The five annotations below were written by Dr. Felicia Ledogar on the companion paper. Each was applied to this manuscript as a general standard.

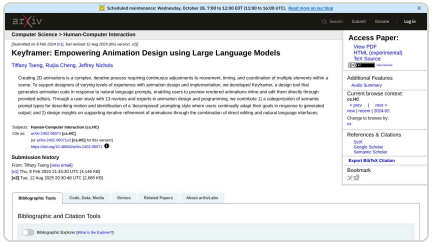
STANDARD	SOURCE ANNOTATION	APPLICATION IN THIS EDITION
Clear referents	“what does its refer to??”	“It,” “its,” and “this” replaced with the noun wherever more than one referent was possible.
Number agreement	“do not use they, it is plural”	Singular subjects such as “a team” and “a developer” are not followed by “they.”
Formal introduction	“intro??? too casual”	The draft’s opening questions restated as a problem definition; headings formalized.
Complete sentences	“last sentence is not a sentence, there is no subject/noun”	Prose fragments expanded into complete sentences. Tables keep short entries.
Precise terminology	“contrastive?? or contrasting??”	Mechanism and motion language distinguished; key terms defined in Section 2.2 and used consistently.

Appendix B. Source verification

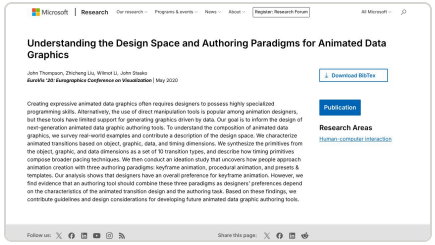
ADDED IN THIS EDITION This appendix is an IEDCO Labs production record. Every link in this edition is live and was tested on October 22, 2026, and each cited page was read to confirm that the page supports the sentence citing it. Two Microsoft Research pages refuse automated clients but open in a real browser, and Crossref confirms both records. The two W3C pages answer automated requests but show a bot check to an automated browser; no bot check was attempted, and those captures were rendered from the fetched pages. Cookie banners were removed from the captures, not clicked.

REF.	HOST	RESULT
[1]	arxiv.org	HTTP 200. Title, authors and revision (v2, August 12, 2025) match; abstract reports the 13-participant study.
[2]	microsoft.com	curl receives 403 (automated client); opens with HTTP 200 in a real browser. Title, authors and EuroVis 2020 match; summary reports the keyframe preference and the recommendation to combine paradigms. Crossref: DOI 10.1111/cgf.13974, Computer Graphics Forum 39(3).
[3]	microsoft.com	curl receives 403 (automated client); opens with HTTP 200 in a real browser. Title and authors match; summary reports two controlled experiments. Crossref: DOI 10.1109/TVCG.2007.70539, IEEE TVCG 13(6).
[4]	web.dev	HTTP 200. Title and authors match; the page recommends transform and opacity and checking other properties against the rendering pipeline.
[5]	angular.dev	HTTP 200. The page states the package is deprecated as of v20.2 and that removing it can significantly reduce bundle size.
[6]	angular.dev	HTTP 200. The page documents animate.enter and animate.leave.
[7]	react.dev	HTTP 200. The page covers cleanup and the development remount after initial mount.
[8]	w3.org	HTTP 200 to curl. The page identifies 2.3.3 as Level AAA and gives the criterion text. A real browser receives a bot check, so the capture is rendered from the fetched page.
[9]	w3.org	HTTP 200 to curl. The technique is listed as sufficient for 2.3.3. Captured from the fetched page, as for [8].
[10]	motion.dev	HTTP 200. The page explains tree shaking of Motion for React.
[11]	motion.dev	HTTP 200. The page states LazyMotion can shrink the initial bundle to as low as 5 kB.
[12]	motion.dev	HTTP 200. The page describes automatic layout animation and shared elements (layoutId).

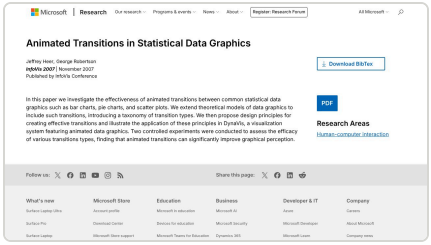
REF.	HOST	RESULT
[13]	`motion.dev`	HTTP 200. The page documents the reducedMotion option and the useReducedMotion hook.
[14]	`industrialethic.com`	HTTP 200. The live Labs page carries the product facts cited (58 options, Angular 21 and 22 tests, not yet verified in production).



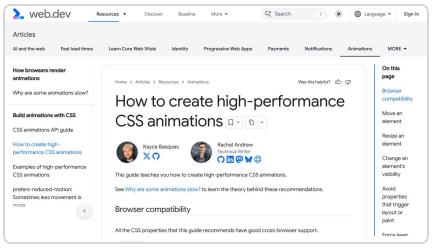
[1] ARXIV.ORG · OCT 22, 2026



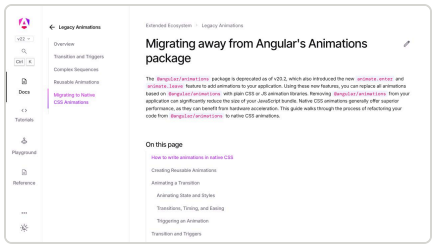
[2] MICROSOFT.COM · OCT 22, 2026



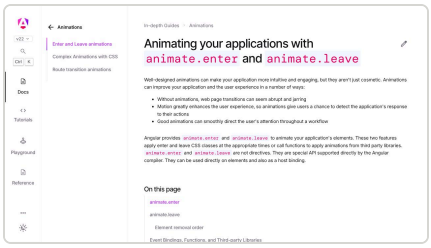
[3] MICROSOFT.COM · OCT 22, 2026



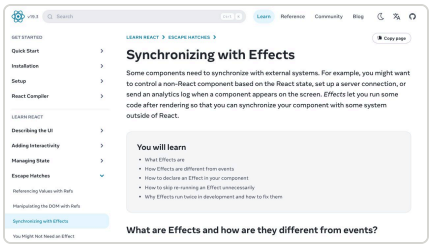
[4] WEB.DEV · OCT 22, 2026



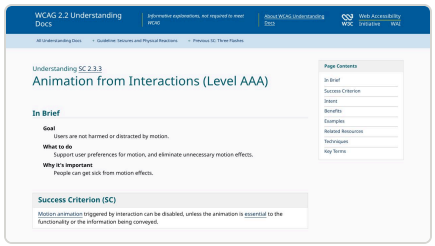
[5] ANGULAR.DEV · OCT 22, 2026



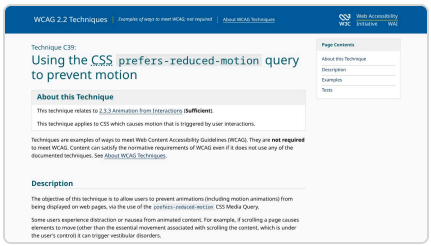
[6] ANGULAR.DEV · OCT 22, 2026



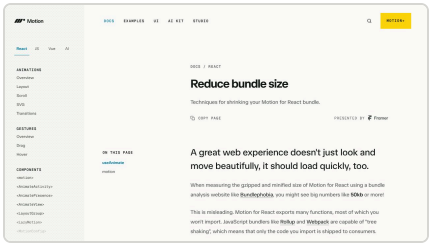
[7] REACT.DEV · OCT 22, 2026



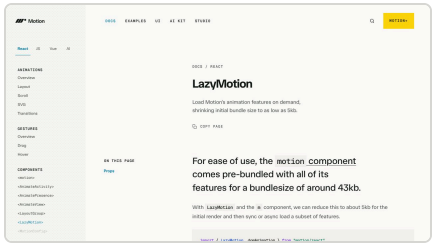
[8] W3.ORG · OCT 22, 2026



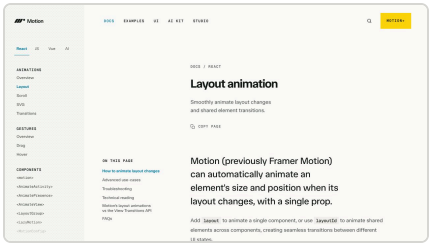
[9] W3.ORG · OCT 22, 2026



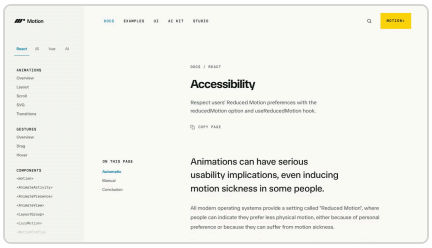
[10] MOTION.DEV · OCT 22, 2026



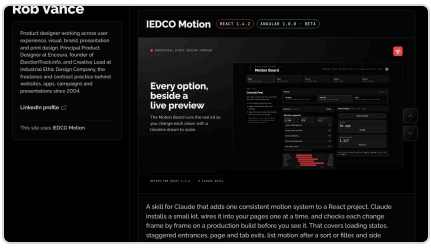
[11] MOTION.DEV · OCT 22, 2026



[12] MOTION.DEV · OCT 22, 2026



[13] MOTION.DEV · OCT 22, 2026



[14] INDUSTRIALETHIC.COM · OCT 22, 2026