DocDancer: Authoring Ultra-responsive Documents with Layout Generation

Yuexi Chen*
University of Maryland
College Park, USA
ychen151@umd.edu

Zhicheng Liu
University of Maryland
College Park, USA
leozcliu@umd.edu

Christopher Tensmeyer
Adobe Research
Maryland, USA
tensmeye@adobe.com

Niklas Elmqvist
University of Maryland
College Park, USA
elm@umd.edu

Vlad I. Morariu
Adobe Research
Maryland, USA
morariu@adobe.com

***Abstract*—Responsive design enhances user experience by adapting layout and content to different display factors. However, existing tools for authoring responsive design primarily adapt to screen width only, and they either rely on predefined templates or require significant manual effort. To expedite responsive design creation, we introduce an authoring tool called *DocDancer*. *DocDancer* supports creating *ultra-responsive* documents where both layout and content adapt to multiple factors (screen width, font properties, customer segments, etc), and provides layout suggestions based on user-provided content and popular responsive patterns. A comparative user study with 16 participants shows that authoring responsive documents in *DocDancer* takes significantly less time and effort than a commercial tool.**

***Index Terms*—responsive design; authoring tools; user interface; hypermedia creation**

I. Introduction

Responsive document design adapts page layout based on screen width to enhance the readability and aesthetics [1]. Here we use *document* to encompass web pages and document-like forms comprising multiple text and visual elements [2]. However, solely adapting documents to screen width is insufficient for optimizing user experience. Additional contextual factors (e.g., font size, line spacing, internet bandwidth [3]–[7]) should also be integrated into responsive design, e.g., low-bandwidth users may require a basic version of websites with fewer images [8], [9], and elderly users might benefit from a special menu layout [10]. To distinguish from the responsive design that only adapts to screen width, we use *ultra-responsive* to describe document behaviors that respond to any contextual factor changes. Existing authoring tools [11], [12] offer limited support for creating ultra-responsive documents. To facilitate efficient exploration of the ultra-responsive design space, we propose a novel authoring tool called *DocDancer* to examine responsive layout options at the late design stage.

Different from existing tools focusing on static layouts, *DocDancer* assigns responsive patterns to content groups and adapts layouts to responsive factors. *DocDancer* also supports multiple responsive patterns to be assigned to the same group and selectively displays patterns based on *active regions* (user-specified responsive factor value ranges). Designers can preview and edit patterns via in-place layout recommendation carousels and off-canvas sliders. Besides the layout, designers can also define custom responsive factors and create alternative content accordingly[1].

We evaluated *DocDancer* via a within-subject study with 16 users, *DocDancer* outperformed Webflow [11], an industry-leading responsive website builder, in terms of efficiency and cognitive load. In summary, our primary contributions include 1) *DocDancer*, a novel WYSIWYG authoring tool for ultra-responsive documents with popular responsive patterns; 2) Empirical results validate the *DocDancer* workflow.

II. Related Work

Responsive web design. Creating responsive websites is a collaborative effort between designers and developers. Initially, designers develop mockups across multiple artboards using prototyping tools like Figma [12], [13], which are then handed over to developers for implementation. While prototyping tools maintain padding during resizing [14], [30], [31], designers must still create different artboards for structural changes. Translating artboards into functional responsive designs necessitates programming expertise [18], [32]–[37]. Expresso [38] infers CSS code but still requires manual keyframe creation. Website builders like Webflow [11], [16] offer responsive templates through a WYSIWYG interface, but these templates focus on resizing only.

Personalized Display and Content. CSS media queries [18] support responsive factors beyond screen widths, such as resolution and aspect ratio. LaaS [39] and C-RWD [19] optimize web layout through visual saliency. Intelligent recommendation systems are widely used [40], [41]. *DocDancer* enables users to create alternative content and layouts for custom factors, while how to collect user information and communicate with the recommender depends on context-aware web services [42]–[46] and is beyond the scope of our discussion.

Layout generation. Several grid-based document templates adapt to screen widths [22], [23], and pre-defined widget templates generate UIs across devices [20], [21]. *Constraint-based* iOS and Android developers can specify widget locations via mathematical equations [47], [48]. Constraint editors [25], [26], [49]–[58] enable users to specify UI widget constraints. *Content-based* authoring tools recommend graphic

*Work done during an internship at Adobe Research

[1] *DocDancer* video demo: https://youtu.be/883cJyM-aH8

Method/Tool	**Screen Width**	**Personalized Factors**	**Recommender**	**WYSIWYG**
Prototyping tools (e.g., Figma) [12]–[14]	*	×	×	✓
Website builders (e.g., Webflow) [11], [15]–[17]	✓	×	×	✓
CSS Media Queries (Personalized display) [18]	✓	✓	×	×
C-RWD (Personalized content) [19]	✓	✓	*	×
Template-based (e.g., Damask) [20]–[24]	✓	×	×	✓
Constraint-based (e.g., Decor) [25], [26]	✓	×	*	*
Content-based (e.g., DesignScape) [27]–[29]	*	×	✓	✓
DocDancer	✓	✓	✓	✓

TABLE I: Literature survey. Comparison between *DocDancer* and related work in terms of the screen width, personalized factors, layout recommendations, and WYSIWYG (✓: full support, *: partial support, ×: no support).

design layouts based on design heuristics [27]–[29]. Deep neural networks trained on large-scale graphics datasets can also synthesize layouts [59]–[62].

We contrast *DocDancer* with related work in Table I.

III. Usage Scenario

We present a use case involving Emma, a designer working for a travel agency, to demonstrate the benefits of *DocDancer*.

A. Design Requirements

Emma is tasked with creating an email newsletter for a marketing campaign to promote a trip to Japan. Here are the key requirements: **Mobile-friendliness:** past surveys show that half of the customers read emails on mobile devices. **Accessibility:** layouts should be visually pleasing in large font sizes, as the elderly contribute significantly to the customer base. **Custom Content:** different coupons for new and existing customers. Therefore, Emma needs to create 12 variations of the newsletter: 2 customer types (new/existing) × 2 font sizes (normal/large) × 3 screen widths (phone/tablet/desktop). Fig. 1 shows 8 of the finished versions. We now compare Emma's authoring experience in a traditional tool and *DocDancer*.

B. Traditional Workflow

Emma uses Webflow [11], a tool with resizable templates. Emma selects a template, inserts her content, and manually adjusts the tablet and phone versions. Subsequently, she duplicates the file, replaces coupon content, and modifies font sizes for different customer segments. It is tedious and error-prone since Emma must manually apply changes to multiple files, risking inconsistencies among different versions.

C. DocDancer Workflow

With *DocDancer*, Emma can create a single document that seamlessly adapts to various layout and content scenarios. Figure 2 shows the UI of *DocDancer*. First, Emma uploads assets to the asset panel (Fig. 2.1). Then she drags images from the asset panel to the canvas (Fig. 2.4) and creates a group of three images (Fig. 2.5). This action triggers *responsive pattern* recommendations, which defines a group's dynamic layout behavior across multiple responsive factors. We curated common responsive patterns in popular responsive websites [64]–[66] to build a minimal viable responsive pattern library, e.g., grid, tabs, carousels, etc.)

Mobile-friendliness: The second recommendation (Fig. 3.1) uses the *grid* pattern (Fig. 3.2). Emma keeps previewing recommendations by clicking ❯ and finds a *carousel* pattern (Fig. 3.3). She adds the *carousel* pattern to the group. Then, she sets the group's pattern to *grid* when the screen width is larger than 900px and *carousel* when it's smaller than 900px (Fig. 3.4). The dark and light blue on sliders indicate active and inactive regions for a pattern.

Accessibility: Next, Emma works on the content for Japanese restaurants. She creates two groups, each consisting of an image and a short text description. She wants the layout of the groups to respond to font size; after previewing recommendations, she decides to apply a *grid* pattern for normal font size (Fig. 4.1), and a *tab* pattern for large font size (Fig. 4.2). She adds both and sets the group's pattern to be *grid* when the font size is 100% to 120%, and *tab* when the font size is larger than 120% (Fig. 4.3).

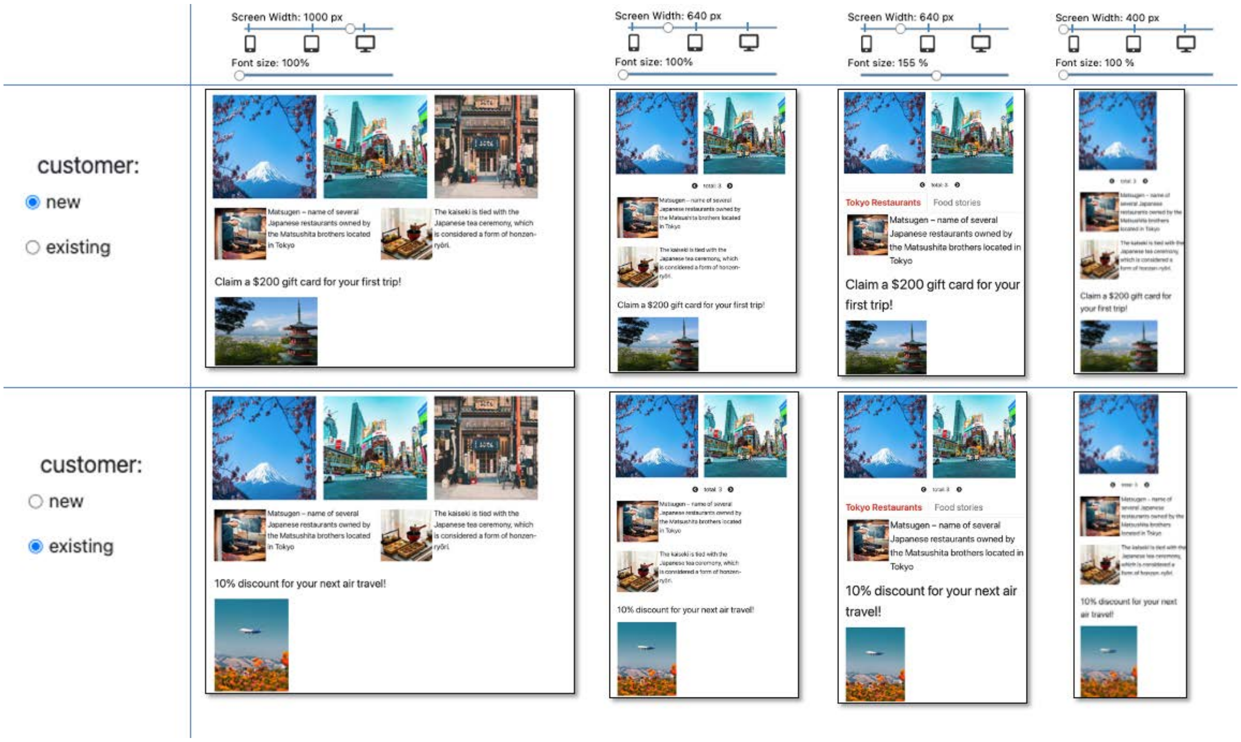
Custom content: To present different coupons to different customers, first, Emma creates a coupon for new customers, then she clicks ⧉ *add alternatives*. The alternative panel now displays two *alternatives*, which initially are duplicates, so Emma can create the second alternative by tweaking the first one rather than starting from scratch (Fig. 5.1). Emma then replaces one duplicate with the coupon for existing customers (Fig. 5.2). Next, Emma clicks *create factors* (Fig. 2.7), names the factor *customer*, and creates two options, *new* and *existing*, then she binds *customer=new* to *Alternative-1* and *customer=existing* to *Alternative-2* in the panel (Fig. 5.4).

Emma then inspects all versions by manipulating the responsive factor values in the *Canvas preview* panel (Fig. 2.6) and examines how the layout and content respond on the canvas. Some combinations are shown in Fig. 1. For the entire canvas, *DocDancer* reflows different groups, which prevents them from overlapping. To revise the dynamic behavior of any content, Emma can select the content on the canvas or on the group panel (Fig 2.9) and make edits. When finished, Emma exports a JSON file, which contains all the information needed to generate JS/HTML/CSS files. She could also load it back to *DocDancer* for further edits.

IV. User Study & Evaluations

We compared *DocDancer* with Webflow [11], the commercial WYSIWYG website builder with the most responsive design features. We aim to answer two questions:

- R1: Does *DocDancer* help users create responsive documents more efficiently than Webflow?



Matsugen – name of several Japanese restaurants owned by the Matsushita brothers located in Tokyo

The kaiseki is tied with the Japanese tea ceremony, which is considered a form of honzen-ryōri.

Claim a $200 gift card for your first trip!

10% discount for your next air travel!

Fig. 1: Previews of an ultra-responsive document showing eight different versions of a marketing newsletter with adaptive layouts and contents designed for different screen widths, font sizes, and customer types.

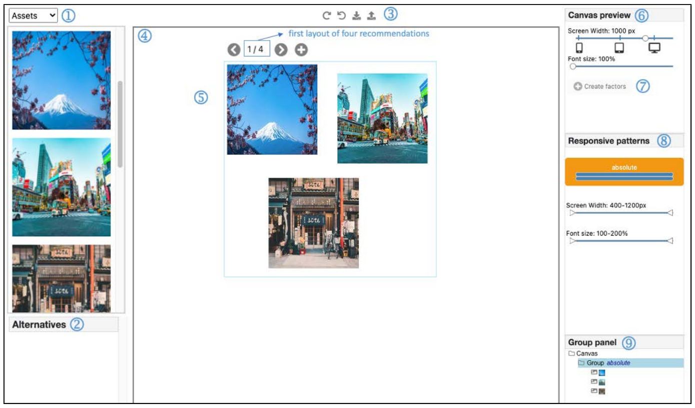


Fig. 2: Overview of the *DocDancer* user interface. (1) Asset panel. (2) Alternative panel (will be populated when alternative content is created). (3) Toolbar. (4) Canvas. (5) A group of three images on canvas. (6) Built-in preview sliders for screen width and font size. (7) Button for creating custom responsive factors. (8) Responsive pattern panel displays responsive pattern names and active regions. (9) Group panel displays content hierarchy and responsive pattern information. (Images are public domain from Unsplash [63].)

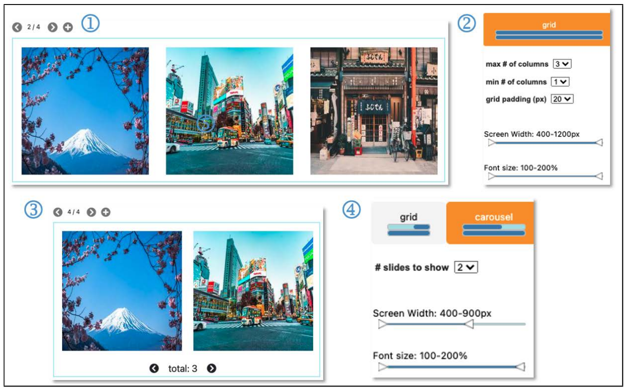
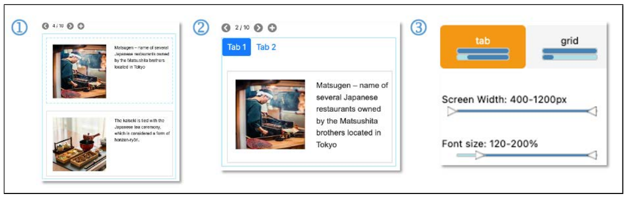


Fig. 3: Responsive pattern recommendations for a group of three images. (1) Recommended *grid* pattern. (2) Adjustable parameters for the grid. (3) Recommended *carousel* pattern. (4) The group applies responsive patterns based on screen width: screen width = 900-1200px: grid; 400-900px: carousel.

- R2: How do users perceive the recommendation-based and content-first workflow in *DocDancer*?

We recruited 16 participants (8 females, 8 males), including 6 professional designers and 10 students in design-related majors, with varying years of experience. All participants were familiar with responsive design, with 12 regularly creating responsive designs and 4 with prior experience with Webflow.

We conducted a within-participant study to compare *DocDancer* with Webflow, randomly and evenly dividing participants into two groups (A and B) based on experience. All user studies were carried out via Zoom and recorded. Participants watched tutorial videos of both tools and were then asked to reproduce a given responsive design using both



Matsugen – name of several Japanese restaurants owned by the Matsushita brothers located in Tokyo

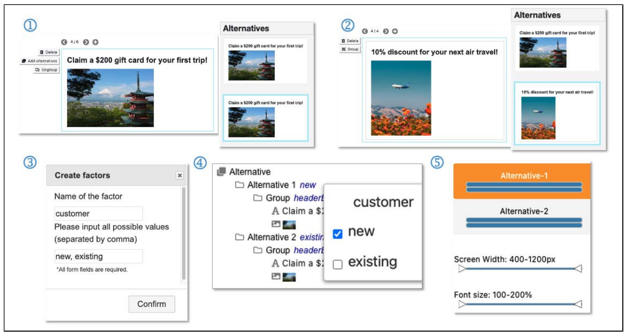
The kaiseki is tied with the Japanese tea ceremony, which is considered a form of honzen-ryōri.

Matsugen – name of several Japanese restaurants owned by the Matsushita brothers located in Tokyo

Screen Width: 400-1200px

Font size: 120-200%

Fig. 4: Responsive pattern recommendations applied to a group of two groups of image and text: (1) Recommended *grid* pattern. (2) Recommended *tab* pattern. (3) The group applies one of these patterns depending on font size: font size is 100%-120%: *grid*, 120%-200%: *tab*.



Claim a $200 gift card for your first trip!

Claim a $200 gift card for your first trip!

Claim a $200 gift card for your first trip!

10% discount for your next air travel!

Claim a $200 gift card for your first trip!

10% discount for your next air travel!

Create factors

Name of the factor

customer

Please input all possible values (separated by comma)

new, existing

*All form fields are required.

customer

☑ new

☐ existing

Screen Width: 400-1200px

Font size: 100-200%

Fig. 5: Creating alternatives in *DocDancer*. (1) Create content for new customers and add alternatives. (2) Create content for existing customers. (3) Input names and options of responsive factors. (4) Bind different options to different content: for *customer* = *new*, show the gift card offer; for *customer* = *existing*, show the discount offer. (5) Both alternatives are active on all font sizes and screen widths, so which alternative is shown depends only on the *customer* variable.

tools. Following the study, participants completed a survey and engaged in a semi-structured interview. Each session lasted approximately 1.5 hours, and participants were compensated with Amazon gift cards ($15/hour). All participants completed both tasks with minimal guidance.

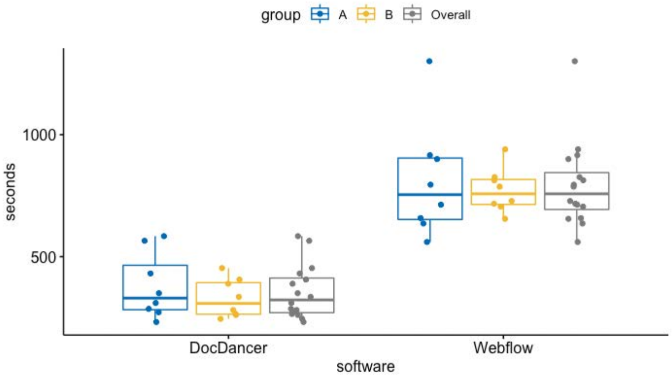


Fig. 6: Task completion time (seconds). *DocDancer* has much less completion time than Webflow.

R1: Does *DocDancer* help users create responsive documents more efficiently than Webflow?

We recorded the task completion time for each participant (Fig. 6). On average, it takes 354.0 seconds (SD = 109.4) to complete the task in *DocDancer* and 790.6 seconds (SD = 172.8) in Webflow. We conducted a paired two-sample Wilcox test (i.e., Mann-Whitney test) and p = 3.052e-05, indicating *DocDancer* being significantly faster than Webflow.

We further compared the task completion time for group A (t_A) and group B (t_B). In the *DocDancer* task, on average, t_A = 378.8 seconds (SD = 134.5), t_B = 329.4 (SD = 78.4). The time difference is insignificant (p=0.50, Wilcox rank sum test). In the Webflow task, t_A = 809.9 seconds (SD = 234.8), t_B = 771.4 (SD = 89.6). The time difference is also not significant (p = 0.88, Wilcox rank sum test), suggesting that the task order did not significantly affect completion time.

In the post-study survey, participants rated *DocDancer* as more efficient than Webflow: *DocDancer* received 3 "Strongly agree", 10 "Agree", and 3 "Neutral", while Webflow received 3 "Strongly agree", 7 "Agree", 5 "Neutral", and 1 "Disagree". Neither tool receives "Strongly disagree." Participants also rated *DocDancer* as less mentally demanding: for *DocDancer*, 7 participants rated "Very low", 3 "Low", 4 "Medium" and 2 "High"; For Webflow, no participants rated "Very low", 6 rated "Low", 4 "Neutral", 5 "High" and 1 "Very high".

R2: How do users perceive the recommendation-based and content-first workflow in *DocDancer*?

Ease of learning. Participants found Webflow's complex, with some struggling to remember where to add a grid template or how to modify the number of columns. Conversely, they found *DocDancer* intuitive to use. P12 said, *"if you want to create very simple documents that are responsive [...] people who are not from a design background can still work in DocDancer, but not necessarily [in] Webflow."*

Content-first vs. template-first layout authoring. In *DocDancer*, after users have created content groups, they only need to preview and select the recommended layout, while in Webflow, they need to modify the responsive template for structural changes. P10 said, *"Switching between different patterns is easy, it's also non-destructive, I don't need to delete or recreate but just switch to a different layout. It's also much simpler than Webflow [...] I don't have to think about which layout I want to use beforehand, I can preview all different options before I make the decision, less mentally demanding."*

V. Conclusion & Future Work

We have introduced *DocDancer*, an authoring tool for ultra-responsive documents, significantly streamlining the design workflow via responsive layout recommendations and custom factors. We believe *DocDancer* is only the beginning of authoring tools tailored for responsive design, with numerous areas for future research: (1) **Data-driven recommendations:** with more responsive designs available, machine learning models could replace the current heuristic pattern match. (2) **Global layout optimization**: currently, *DocDancer* recommends layout for each group independently, future work could optimize responsive layout for the entire page.

References

[1] E. Marcotte, "Responsive web design." https://alistapart.com/article/responsive-web-design, 2010.

[2] V. I. Morariu, C. Yuexi, C. A. Tensmeyer, Z. Liu, and L. N. E. Elmqvist, "Responsive document authoring," May 25 2023. US Patent App. 17/535,067.

[3] S. Dutton, "Understanding low bandwidth and high latency." https://developers.google.com/web/fundamentals/performance/poor-connectivity, 2022.

[4] S. Beier, S. Berlow, E. Boucaud, Z. Bylinskii, T. Cai, J. Cohn, K. Crowley, S. L. Day, T. Dingler, J. Dobres, J. Healey, R. Jain, M. Jordan, B. Kerr, Q. Li, D. B. Miller, S. Nobles, A. Papoutsaki, J. Qian, T. Rezvanian, S. Rodrigo, B. D. Sawyer, S. M. Sheppard, B. Stein, R. Treitman, J. Vanek, S. Wallace, and B. Wolfe, "Readability research: An interdisciplinary approach," *CoRR*, vol. abs/2107.09615, 2021.

[5] S. Wallace, R. Treitman, J. Huang, B. D. Sawyer, and Z. Bylinskii, "Accelerating adult readers with typeface: A study of individual preferences and effectiveness," in *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 1–9, ACM, 2020.

[6] S. Wallace, Z. Bylinskii, J. Dobres, B. Kerr, S. Berlow, R. Treitman, N. Kumawat, K. Arpin, D. B. Miller, J. Huang, and B. D. Sawyer, "Towards individuated reading experiences: Different fonts increase reading speed for different individuals," *ACM Transactions on Computer-Human Interaction*, vol. 29, no. 4, pp. 38:1–38:56, 2022.

[7] S. Wallace, Z. Bylinskii, J. Dobres, B. Kerr, S. Berlow, R. Treitman, N. Kumawat, K. Arpin, D. B. Miller, J. Huang, *et al.*, "Towards individuated reading experiences: Different fonts increase reading speed for different individuals," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 29, no. 4, pp. 1–56, 2022.

[8] S. Morris, "Designing for low bandwidth users." https://clarkstonconsulting.com/insights/designing-for-low-bandwidth-users/, 2021.

[9] J. Jackson, "How facebook made mobile site faster for users with limited bandwidth." https://www.computerworld.com/article/2960779/how-facebook-made-mobile-site-faster-for-users-with-limited-bandwidth.html, 2015.

[10] H. L. Antonelli, W. M. Watanabe, and R. P. de Mattos Fortes, "Adapting web menus to mobile devices for elderly interactions," in *Proceedings of the International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-Exclusion*, (New York, NY, USA), pp. 47–54, ACM, 2018.

[11] https://webflow.com.

[12] https://www.figma.com/".

[13] https://www.adobe.com/products/xd.html.

[14] https://www.sketch.com/docs/designing/symbols/resizing-symbols-with-smart-layout/.

[15] https://wix.com.

[16] Squarespace Inc. https://www.squarespace.com/.

[17] https://weebly.com.

[18] https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries.

[19] M. Laine, Y. Zhang, S. Santala, J. P. P. Jokinen, and A. Oulasvirta, "Responsive and personalized web layouts with integer programming," *Proceedings of the ACM on Human-Computer Interaction*, vol. 5, no. EICS, pp. 1–23, 2021.

[20] J. Lin and J. A. Landay, "Employing patterns and layers for early-stage design and prototyping of cross-device user interfaces," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 1313–1322, ACM, 2008.

[21] J. Nichols, B. A. Myers, and K. Litwack, "Improving automatic interface generation with smart templates," in *Proceedings of the ACM Conference on Intelligent User Interfaces*, (New York, NY, USA), pp. 286–288, ACM, 2004.

[22] E. Schrier, M. Dontcheva, C. Jacobs, G. Wade, and D. Salesin, "Adaptive layout for dynamically aggregated documents," in *Proceedings of the ACM Conference on Intelligent User Interfaces*, (New York, NY, USA), pp. 99–108, ACM, 2008.

[23] C. Jacobs, W. Li, E. Schrier, D. Bargeron, and D. Salesin, "Adaptive grid-based document layout," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 838–847, 2003.

[24] C. Jacobs, W. Li, E. Schrier, D. Bargeron, and D. Salesin, "Adaptive document layout," *Communications of the ACM*, vol. 47, no. 8, pp. 60–66, 2004.

[25] N. Sinha and R. Karim, "Responsive designs in a snap," in *Proceedings of the Joint Meeting on Foundations of Software Engineering*, (New York, NY, USA), pp. 544–554, ACM, 2015.

[26] Y. Jiang, R. Du, C. Lutteroth, and W. Stuerzlinger, "ORC layout: Adaptive GUI layout with OR-constraints," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 413:1–413:12, ACM, 2019.

[27] P. O'Donovan, A. Agarwala, and A. Hertzmann, "Learning layouts for single-page graphic designs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 8, pp. 1200–1213, 2014.

[28] P. O'Donovan, A. Agarwala, and A. Hertzmann, "Designscape: Design with interactive layout suggestions," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 1221–1224, ACM, 2015.

[29] K. Todi, D. Weir, and A. Oulasvirta, "Sketchplore: Sketch and explore with a layout optimiser," in *Proceedings of the ACM Conference on Designing Interactive Systems*, (New York, NY, USA), pp. 543–555, ACM, 2016.

[30] O. Lindberg, "Best practices for designing with responsive resize in adobe xd." https://blog.adobe.com/en/publish/2018/09/28/how-to-design-with-responsive-resize-xd.html, 2018.

[31] https://help.figma.com/hc/en-us/articles/360040451373-Create-dynamic-designs-with-auto-layout.

[32] https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox.

[33] Bootstrap Team, "bootstrap." https://getbootstrap.com/.

[34] Django Software Foundation. https://docs.djangoproject.com/en/3.2/topics/templates/.

[35] https://masonry.desandro.com.

[36] https://jekyllrb.com/.

[37] https://gohugo.io/.

[38] R. Krosnick, S. W. Lee, W. S. Lasecki, and S. Onev, "Expresso: Building responsive interfaces with keyframes," in *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*, (Piscataway, NJ, USA), pp. 39–47, IEEE, 2018.

[39] M. Laine, A. Nakajima, N. Dayama, and A. Oulasvirta, "Layout as a service (laas): A service platform for self-optimizing web layouts," in *International Conference on Web Engineering*, pp. 19–26, Springer, 2020.

[40] D. Das, L. Sahoo, and S. Datta, "A survey on recommendation system," *International Journal of Computer Applications*, vol. 160, no. 7, 2017.

[41] E. Adar, C. Gearig, A. Balasubramanian, and J. Hullman, "Persalog: Personalization of news article content," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 3188–3200, 2017.

[42] Q. Z. Sheng and B. Benatallah, "Contextuml: a uml-based modeling language for model-driven development of context-aware web services," in *International Conference on Mobile Business (ICMB'05)*, pp. 206–212, IEEE, 2005.

[43] S. Ceri, F. Daniel, and M. Matera, "Extending webml for modeling multi-channel context-aware web applications," in *Fourth International Conference on Web Information Systems Engineering Workshops, 2003. Proceedings.*, pp. 225–233, IEEE, 2003.

[44] C. Ahn and Y. Nah, "Design of location-based web service framework for context-aware applications in ubiquitous environments," in *2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, pp. 426–433, IEEE, 2010.

[45] B. Salem, J. Alves Lino, and J. Simons, "A framework for responsive environments," in *European Conference on Ambient Intelligence*, pp. 263–277, Springer, 2017.

[46] P. Dourish, "Seeking a foundation for context-aware computing," *Human–Computer Interaction*, vol. 16, no. 2-4, pp. 229–241, 2001.

[47] E. Sadun, *iOS Auto Layout Demystified*. Upper Saddle River, NJ, USA: Addison-Wesley Professional, 2013.

[48] "Build a responsive ui with constraintlayout." https://developer.android.com/training/constraint-layout".

[49] K. Gajos and D. S. Weld, "SUPPLE: automatically generating user interfaces," in *Proceedings of the ACM Conference on Intelligent User Interfaces*, (New York, NY, USA), pp. 93–100, ACM, 2004.

[50] C. Zeidler, C. Lutteroth, W. Sturzlinger, and G. Weber, "The Auckland layout editor: an improved GUI layout specification process," in *Proceedings of the ACM Symposium on User Interface Software and Technology*, (New York, NY, USA), pp. 343–352, ACM, 2013.

[51] C. Zeidler, C. Lutteroth, W. Stuerzlinger, and G. Weber, "Evaluating direct manipulation operations for constraint-based layout," in *Proceedings of the IFIP Conference on Human-Computer Interaction*, vol. 8118 of *Lecture Notes in Computer Science*, (New York, NY, USA), pp. 513–529, Springer, 2013.

[52] C. Zeidler, C. Lutteroth, and G. Weber, "Constraint solving for beautiful user interfaces: how solving strategies support layout aesthetics," in *Proceedings of the International Conference of the NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction*, (New York, NY, USA), pp. 72–79, ACM, 2012.

[53] C. Lutteroth, R. Strandh, and G. Weber, "Domain specific high-level constraints for user interface layout," *Constraints*, vol. 13, no. 3, pp. 307–342, 2008.

[54] L. de Moura and N. Bjørner, "Z3: An efficient SMT solver," in *Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, vol. 4963 of *Lecture Notes in Computer Science*, (Berlin, Heidelberg, Germany), pp. 337–340, Springer, 2008.

[55] Y. Jiang, W. Stuerzlinger, M. Zwicker, and C. Lutteroth, "ORCSolver: An efficient solver for adaptive gui layout with or-constraints," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 483:1–483:14, ACM, 2020.

[56] Y. Jiang, W. Stuerzlinger, and C. Lutteroth, "ReverseORC: Reverse engineering of resizable user interface layouts with or-constraints," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 316:1–316:18, ACM, 2020.

[57] A. Swearngin, C. Wang, A. Oleson, J. Fogarty, and A. J. Ko, "Scout: Rapid exploration of interface layout alternatives through high-level design constraints," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 466:1–466:13, ACM, 2020.

[58] T. Hottelier, R. Bodik, and K. Ryokai, "Programming by manipulation for layout," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York, NY, USA), pp. 231–241, ACM, 2014.

[59] X. Zheng, X. Qiao, Y. Cao, and R. W. Lau, "Content-aware generative modeling of graphic design layouts," *ACM Transactions on Graphics*, vol. 38, no. 4, pp. 133:1–133:15, 2019.

[60] J. Li, J. Yang, A. Hertzmann, J. Zhang, and T. Xu, "LayoutGAN: Generating graphic layouts with wireframe discriminators," *CoRR*, vol. abs/1901.06767, 2019.

[61] J. Li, J. Yang, J. Zhang, C. Liu, C. Wang, and T. Xu, "Attribute-conditioned layout gan for automatic graphic design," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 10, pp. 4039–4048, 2020.

[62] H.-Y. Lee, L. Jiang, I. Essa, P. B. Le, H. Gong, M.-H. Yang, and W. Yang, "Neural design network: Graphic layout generation with constraints," in *Proceedings of the European Conference on Computer Vision*, vol. 12348 of *Lecture Notes in Computer Science*, (New York, NY, USA), pp. 491–506, Springer, 2020.

[63] https://unsplash.com/.

[64] Alexa, "The top 500 sites on the web." https://www.alexa.com/topsites.

[65] Designmodo, "Responsive web design: 50 examples and best practices." https://designmodo.com/responsive-design-examples/, 2021.

[66] J. Kirkwood, "11 powerful examples of responsive web design." https://designmodo.com/responsive-design-examples/, 2018.