



Assessing the User Interaction Cost of Keyboard Navigation in Web Applications

Christina Z. Chaniotaki, Paul T. Chiou, William G.J. Halfond

**International Conference on Software Maintenance and Evolution
(ICSME) 2026**

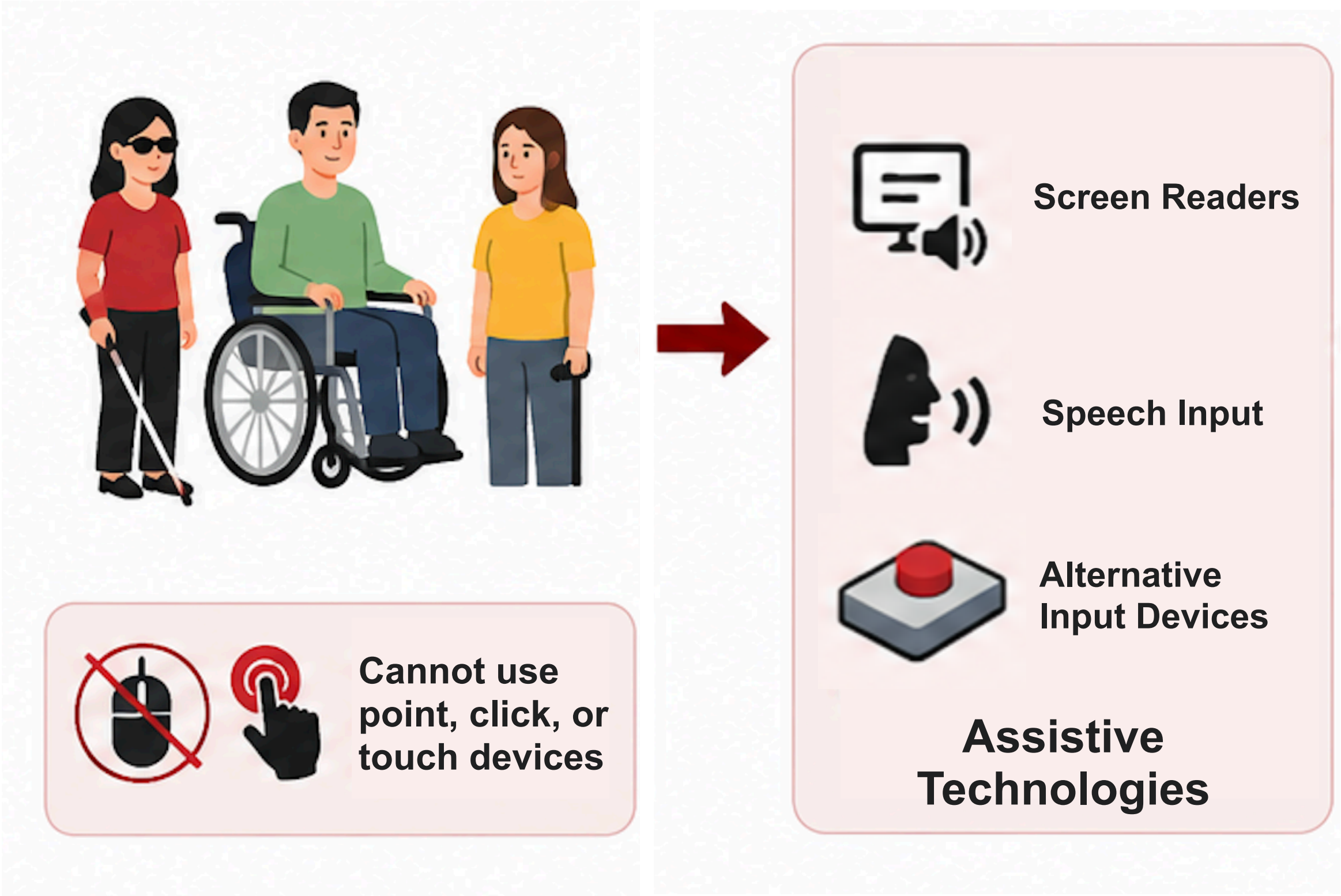


16% of the
Global Population
live with a **disability**

That's **1** in **6** of us



**Cannot use
point, click, or
touch devices**





  Cannot use
point, click, or
touch devices



Screen Readers



Speech Input



Alternative
Input Devices

**Assistive
Technologies**



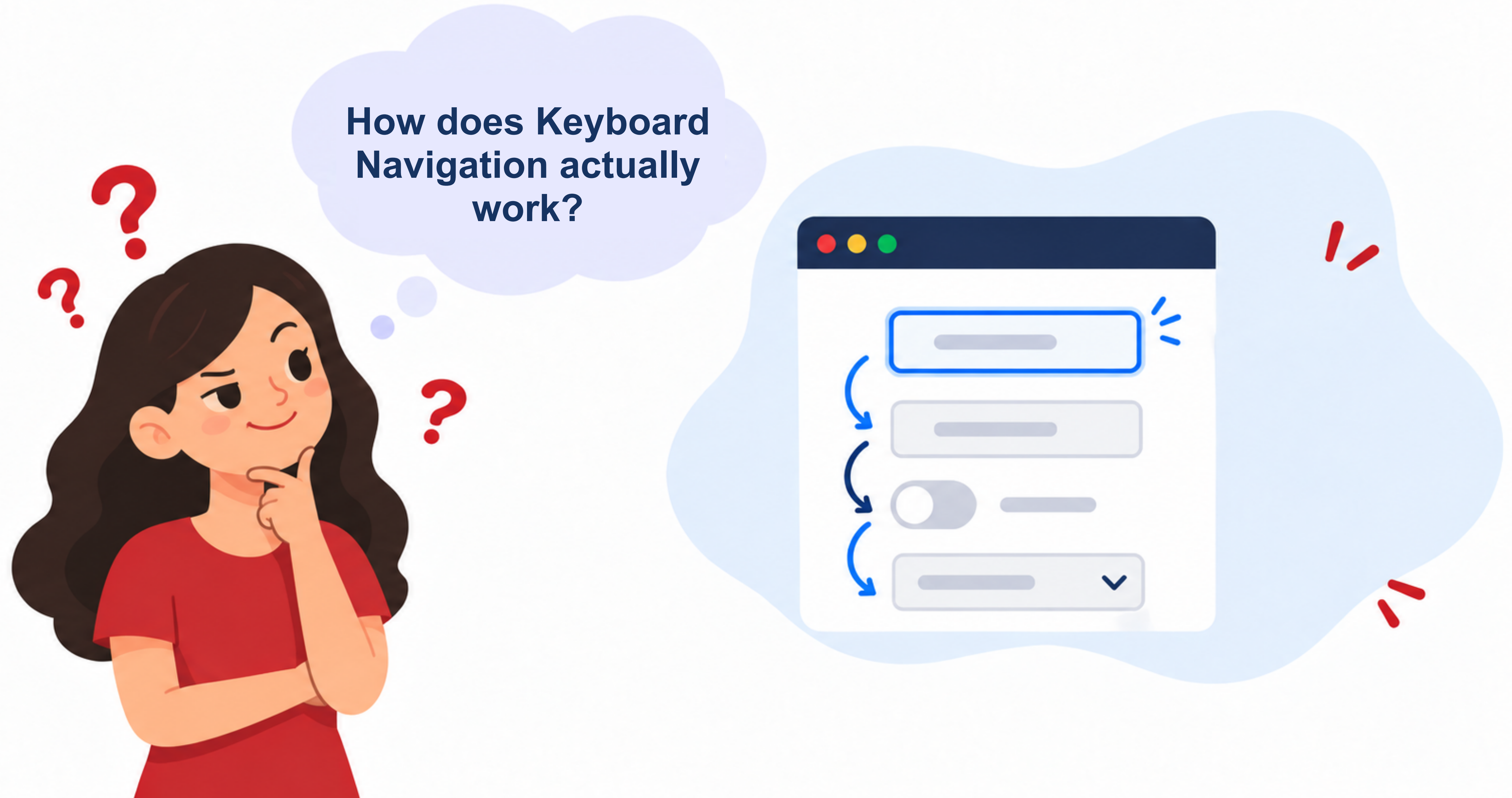
Most Assistive
Technologies simulate
keyboard navigation



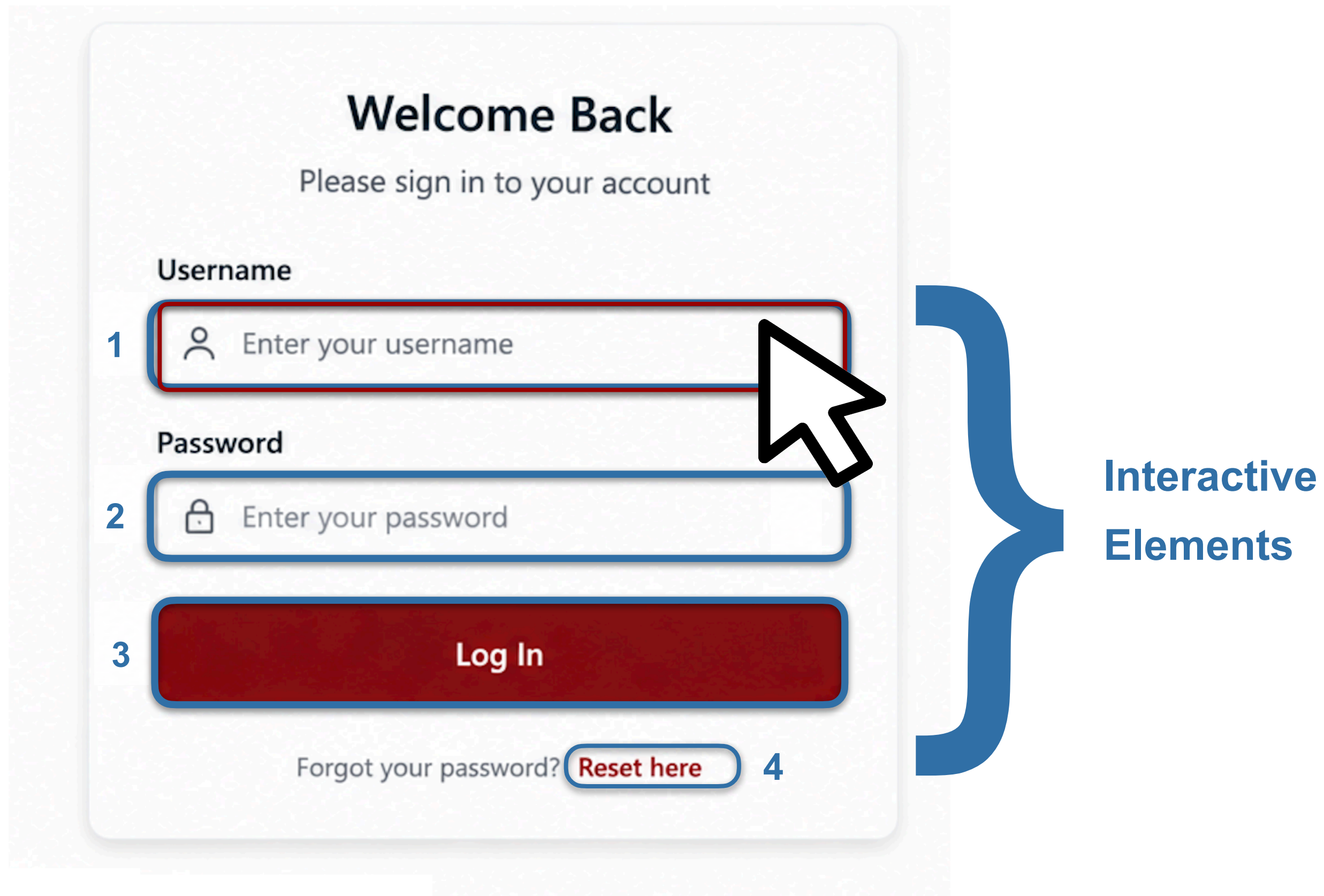
Navigate web applications
efficiently and **independently**



The keyboard is the **foundation of digital accessibility** for many
Assistive Technologies used by **people with disabilities**



Keyboard Navigation Example



Welcome Back
Please sign in to your account

Username

1

Password

2

3

Forgot your password? [Reset here](#) 4

Interactive Elements



Keyboard Navigation Example

Welcome Back
Please sign in to your account

Username

Password

Log In

Forgot your password? [Reset here](#)

Tab

 **Standard keyboard navigation actions**

Tab **Move forward**



Keyboard Navigation Example

Welcome Back
Please sign in to your account

Username

Password

Log In

Forgot your password? [Reset here](#)

Shift + **Tab**

 **Standard keyboard navigation actions**

Tab **Move forward**

Shift + **Tab** **Move backward**



Keyboard Navigation Example

Welcome Back

Please sign in to your account

Username

 Enter your username

Password

 Enter your password

Log In

Forgot your password? [Reset here](#)

Enter



Standard keyboard navigation actions

Tab	Move forward
Shift + Tab	Move backward
Enter / Space	Activate an element



Keyboard Navigation Example

Welcome Back

Please sign in to your account

Username

 Enter your username

Password

 Enter your password

Log In

Forgot your password? [Reset here](#)



Standard keyboard navigation actions

Tab

Move forward

Shift

+

Tab

Move backward

Enter / Space

Activate an element

←

↑

↓

→

Scroll / move

Esc

Close modals or menus

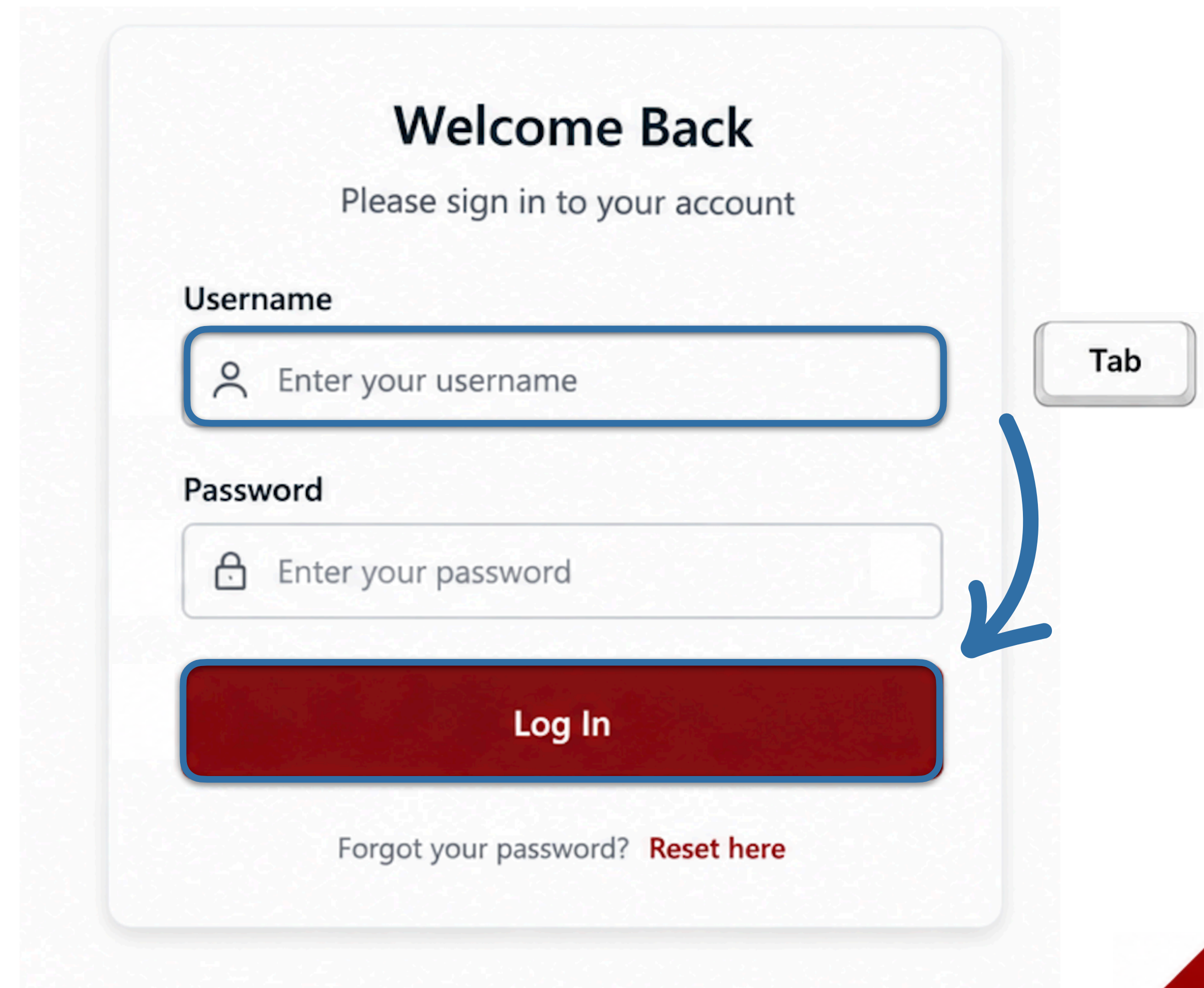


The Web Remains Unfriendly for Keyboard-Based Users

48%
of popular sites
lack full keyboard access



Focus Problem



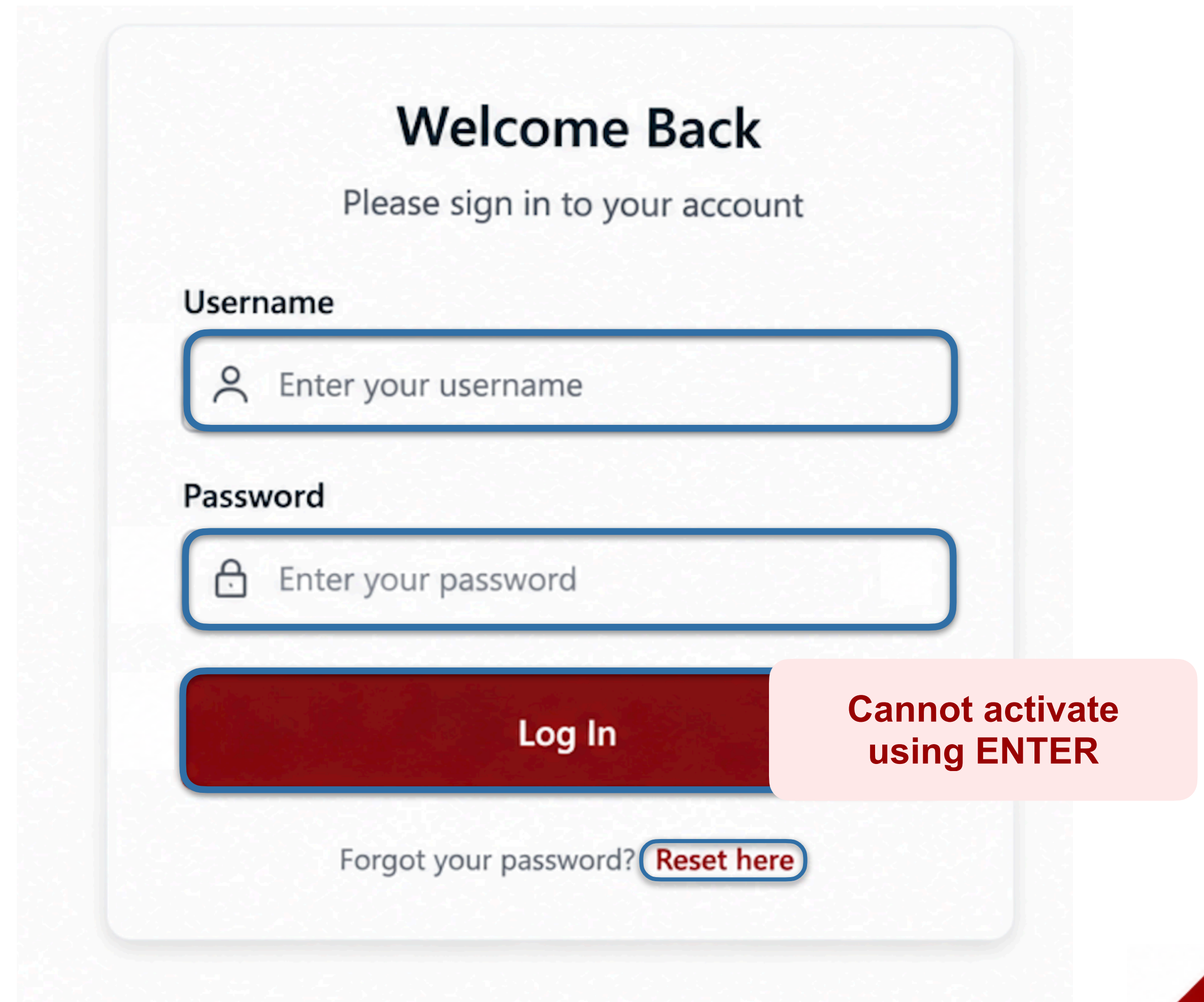
The screenshot shows a login form titled "Welcome Back" with the instruction "Please sign in to your account". It contains three main input areas: a "Username" field with a user icon and placeholder text "Enter your username", a "Password" field with a lock icon and placeholder text "Enter your password", and a "Log In" button. A "Tab" button is located to the right of the input fields. A blue arrow points from the "Tab" button to the "Log In" button, indicating a focus issue where the focus is not correctly transferred to the next element.

The Web Remains Unfriendly for Keyboard-Based Users

48%
of popular sites
lack full keyboard access



Activation Problem



Welcome Back
Please sign in to your account

Username

Password

Log In

Forgot your password? [Reset here](#)

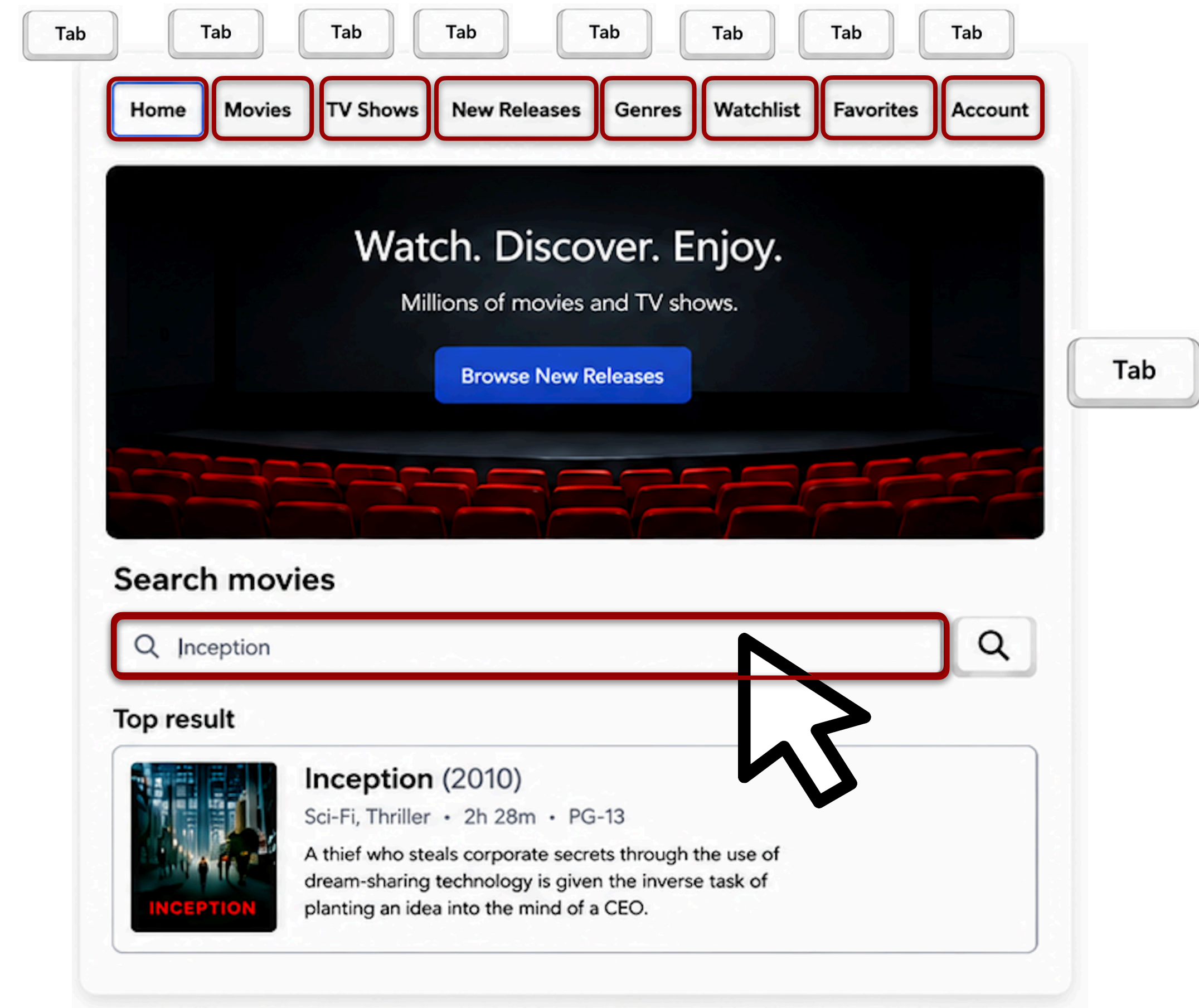
Cannot activate using ENTER

The Web Remains Unfriendly for Keyboard-Based Users

48%
of popular sites
lack full keyboard access



Increased Effort
9 Tabs vs 1 mouse click



Accessibility involves not only whether interaction is **possible**, but also whether it is **practical**

Our Goal

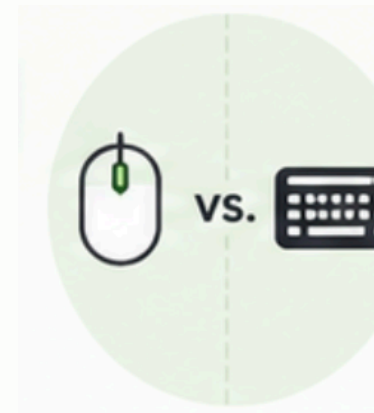
Investigate and understand the **navigation effort** and **discrepancies in UI behavior** between keyboard and mouse navigation

RQ1

What is the accessibility interaction cost for keyboard-based users to **navigate** a web page?

RQ2

What effort is required for keyboard-based versus mouse users to **complete typical use cases** in web apps?

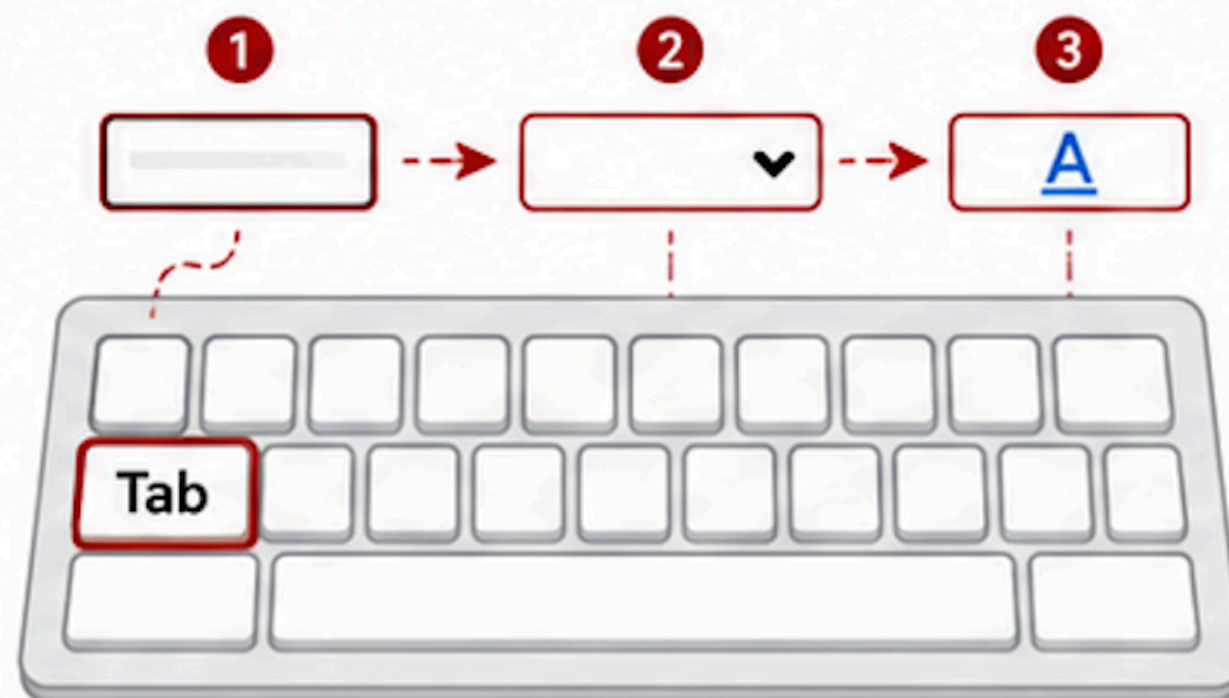
RQ3

What are the **differences in UI behavior** when interacting with a web page via the keyboard versus the mouse?

Methodology

Simulate web navigation using crawling tools that capture the different UI states on each webpage

For Keyboard

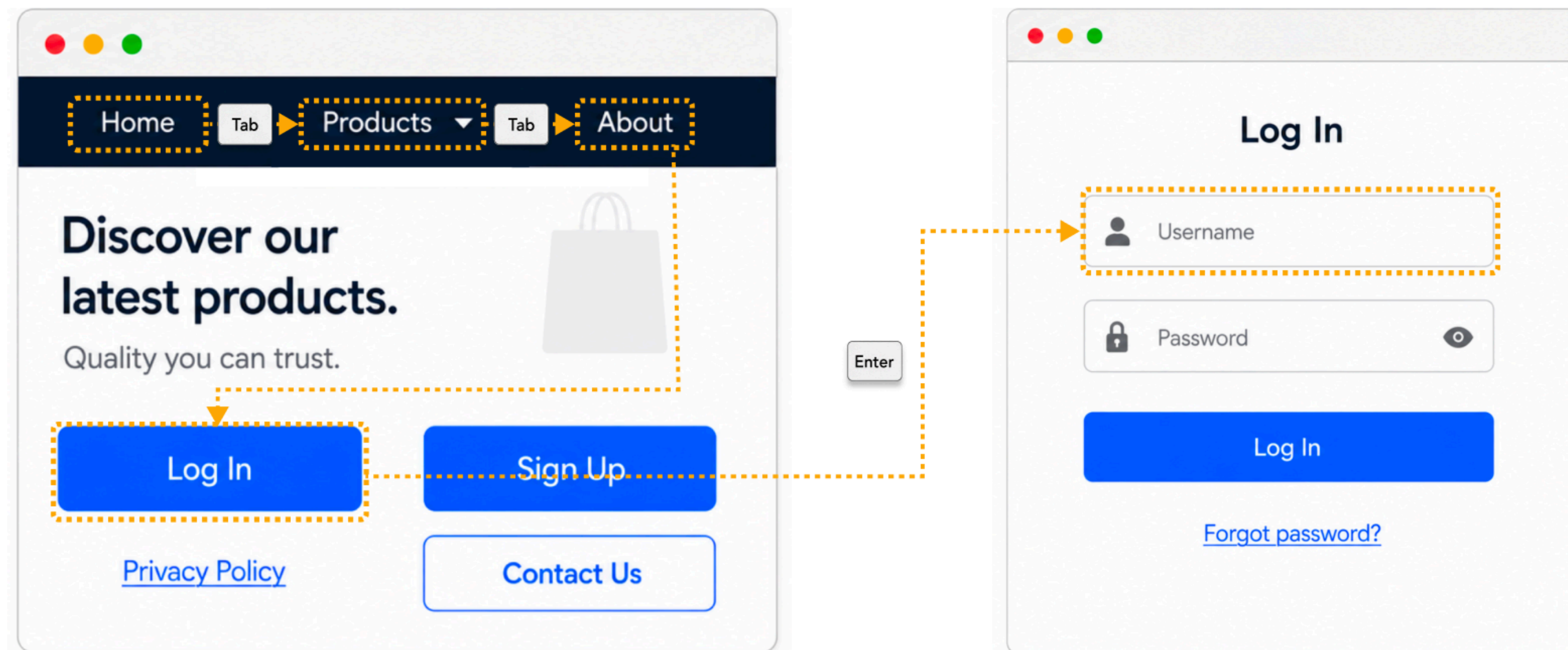


Keyboard-based crawler

We designed a tool (Krawler)

The {K}eyboard Navigation C{rawler} Tool: Krawler

Simulates keyboard navigation on a given web page and generates a Keyboard Experience Representation (KER) model



1

Starts from the given web page and finds all the interactive elements

2

Tries all the standard keyboard actions on each interactive element

3

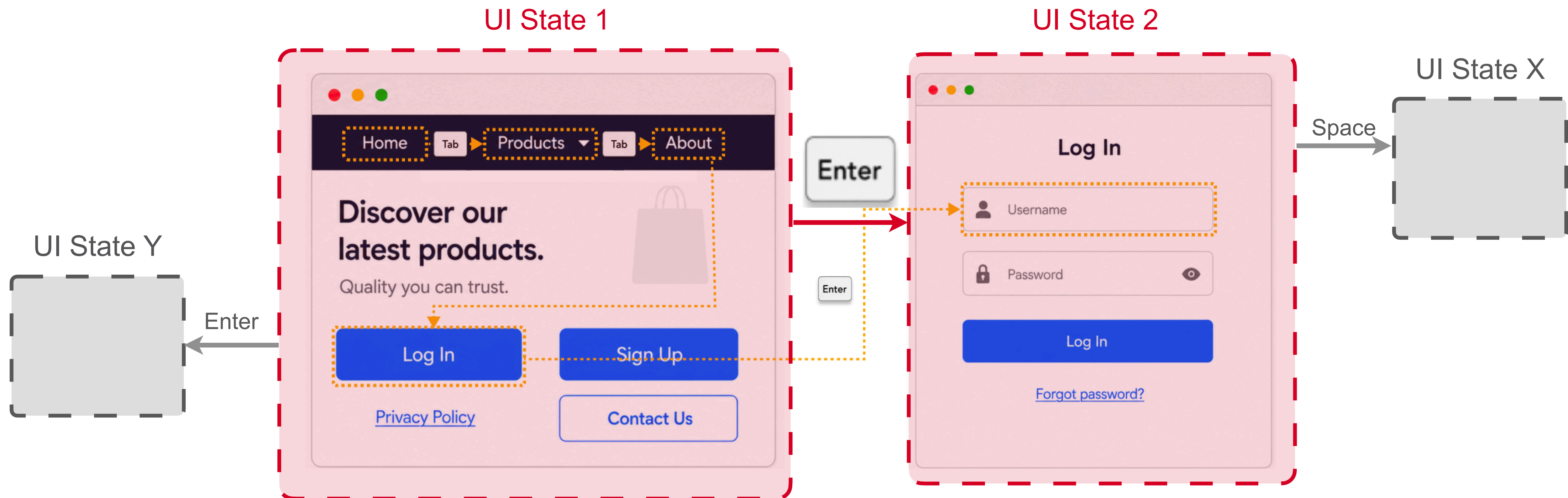
Records the results of each action. If new elements appear, a new UI state is created

4

Repeats until no new states or elements are found.

The {K}eyboard Navigation C{rawler} Tool: Krawler

Simulates keyboard navigation on a given web page and generates a Keyboard Experience Representation (KER) model

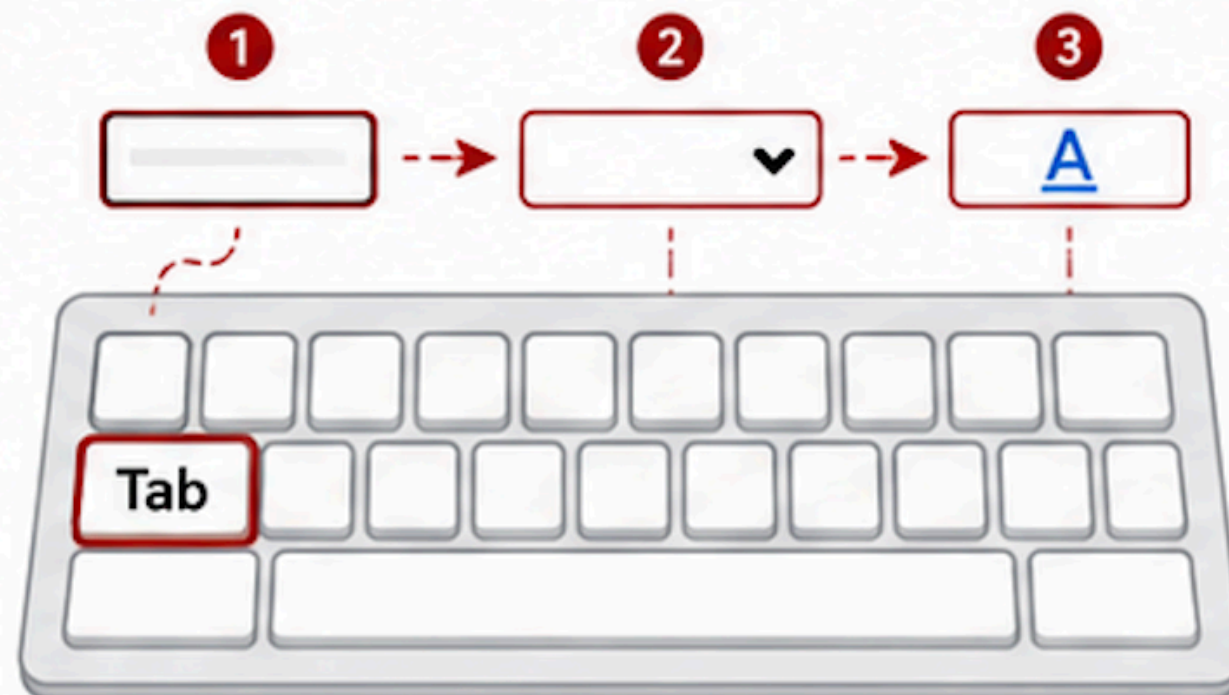


- 1 **Starts from the given web page** and finds all the interactive elements
- 2 **Tries all the standard keyboard actions** on each interactive element
- 3 **Records the results** of each action. If new elements appear, a new UI state is created
- 4 **Repeats** until no new states or elements are found. The result is the **KER**

Methodology

Simulate web navigation using crawling tools that capture the different UI states on each webpage

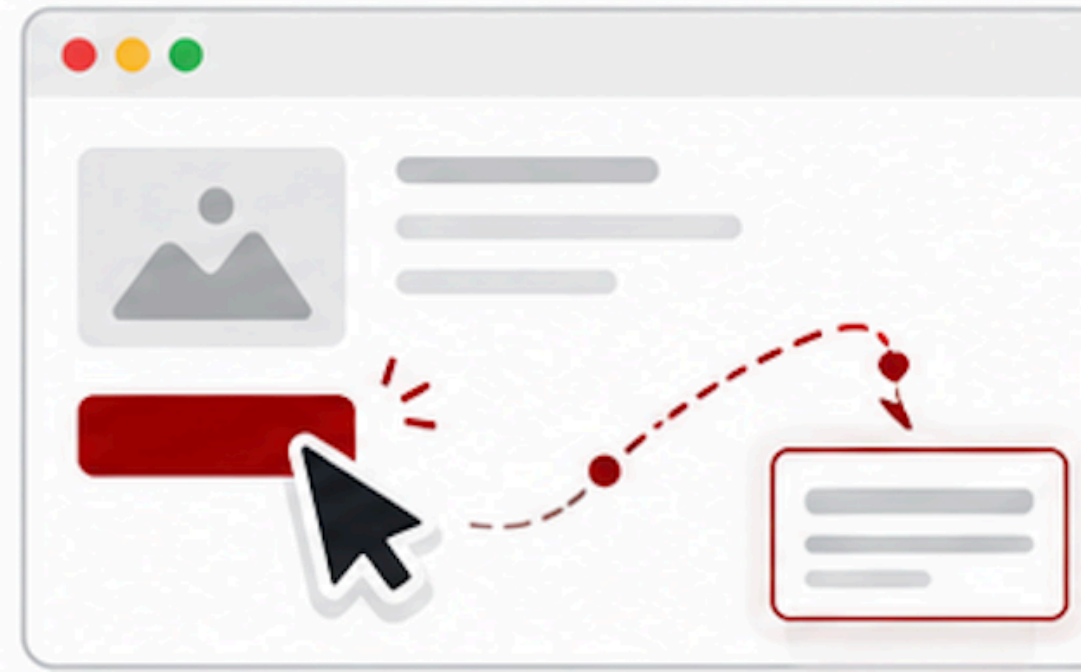
For Keyboard



Keyboard-based crawler

We designed a tool (Krawler)

For Mouse



Point-and-click crawlers

Crawljax

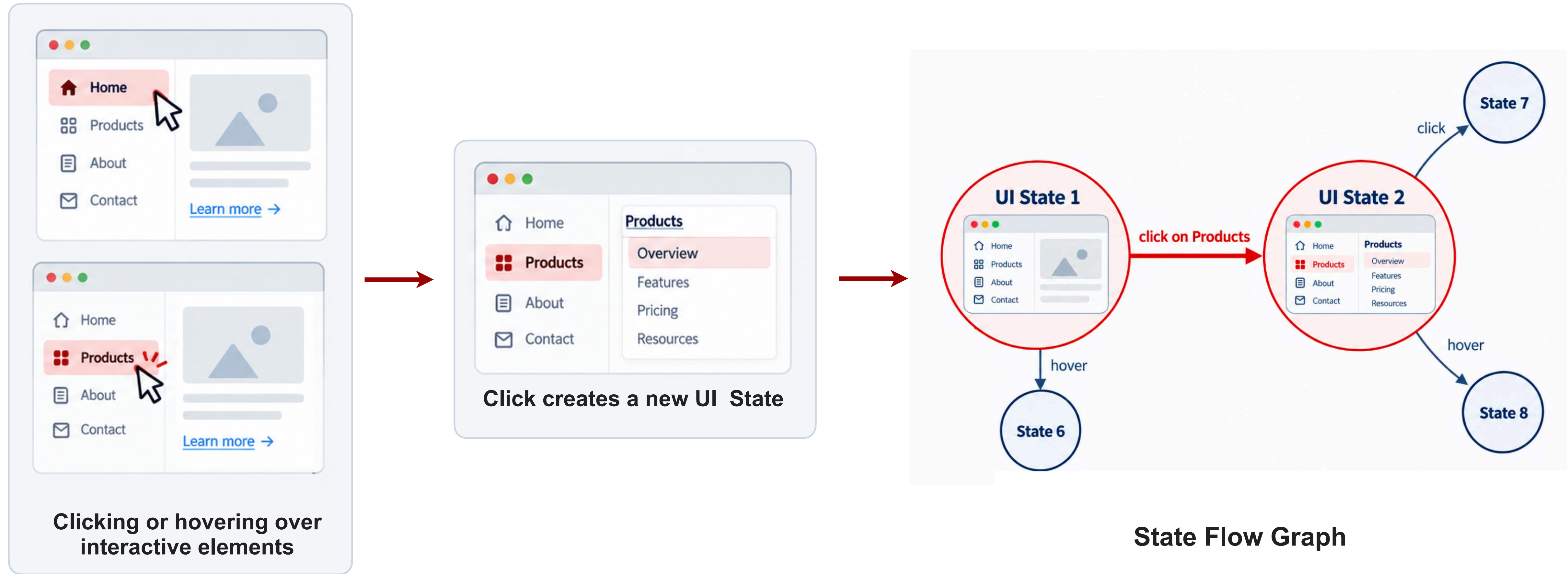
Qualstate

RQ1

RQ2

RQ3

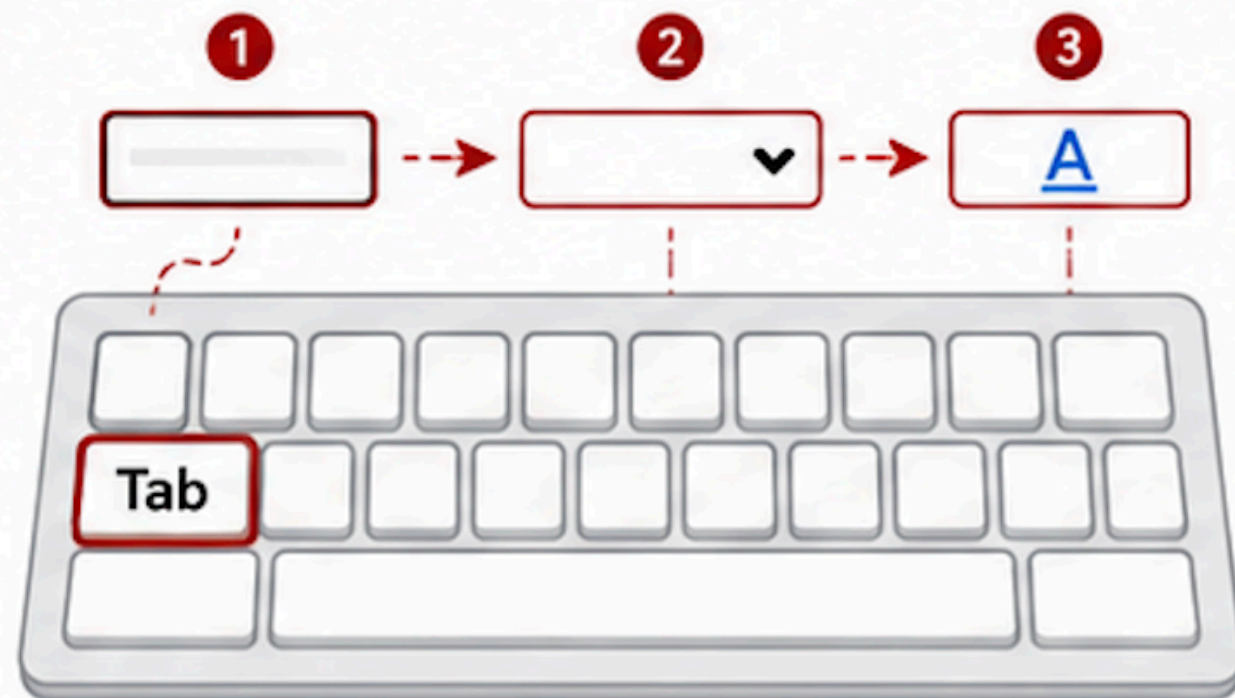
Point-and-Click Crawlers



Methodology

Simulate web navigation using crawling tools that capture the different UI states on each webpage

For Keyboard



Keyboard-based crawler

We designed a tool (Krawler)

For Mouse



Point-and-click crawlers

Crawljax

Qualstate

Testbed



Browsers



Operating systems



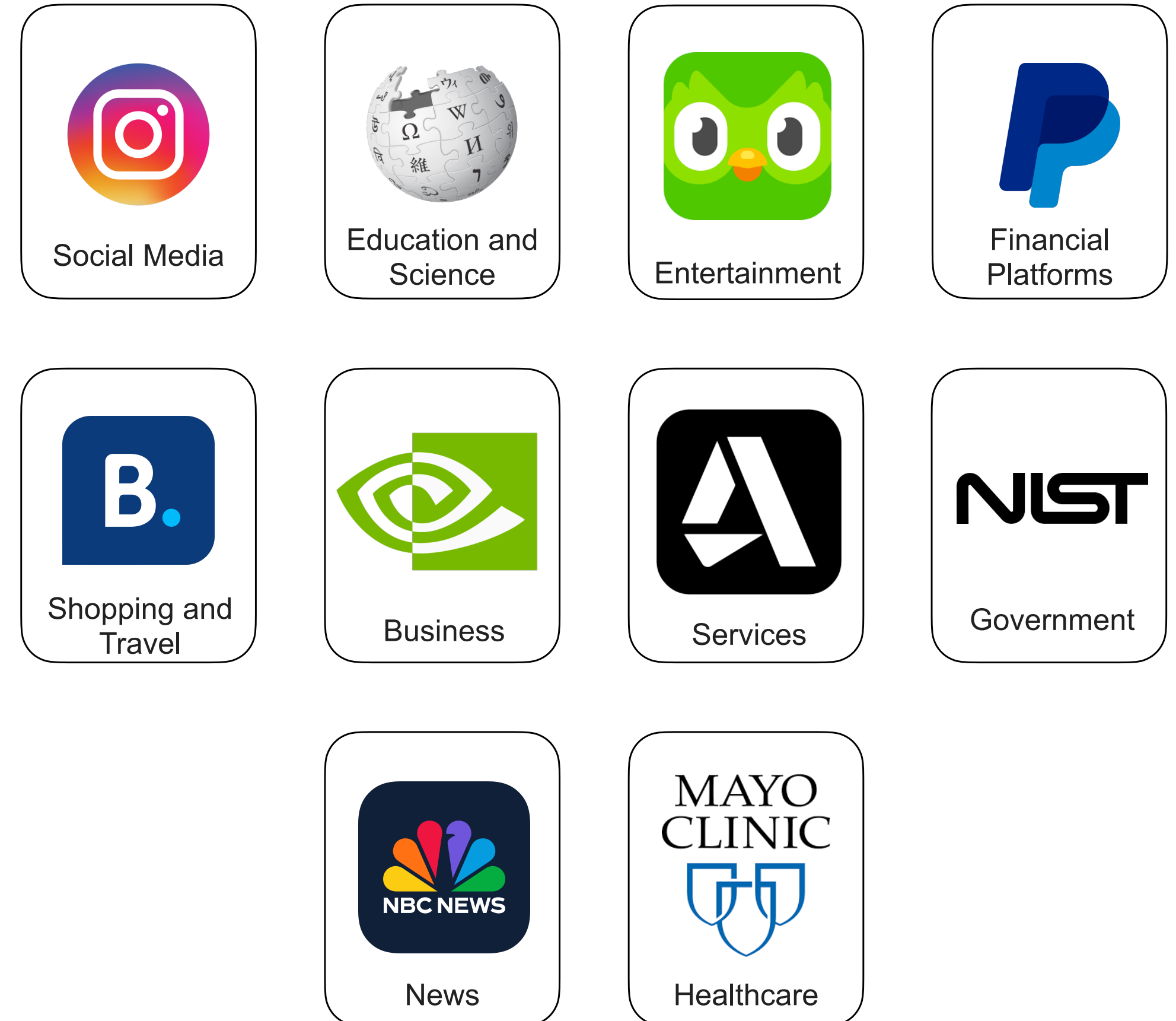
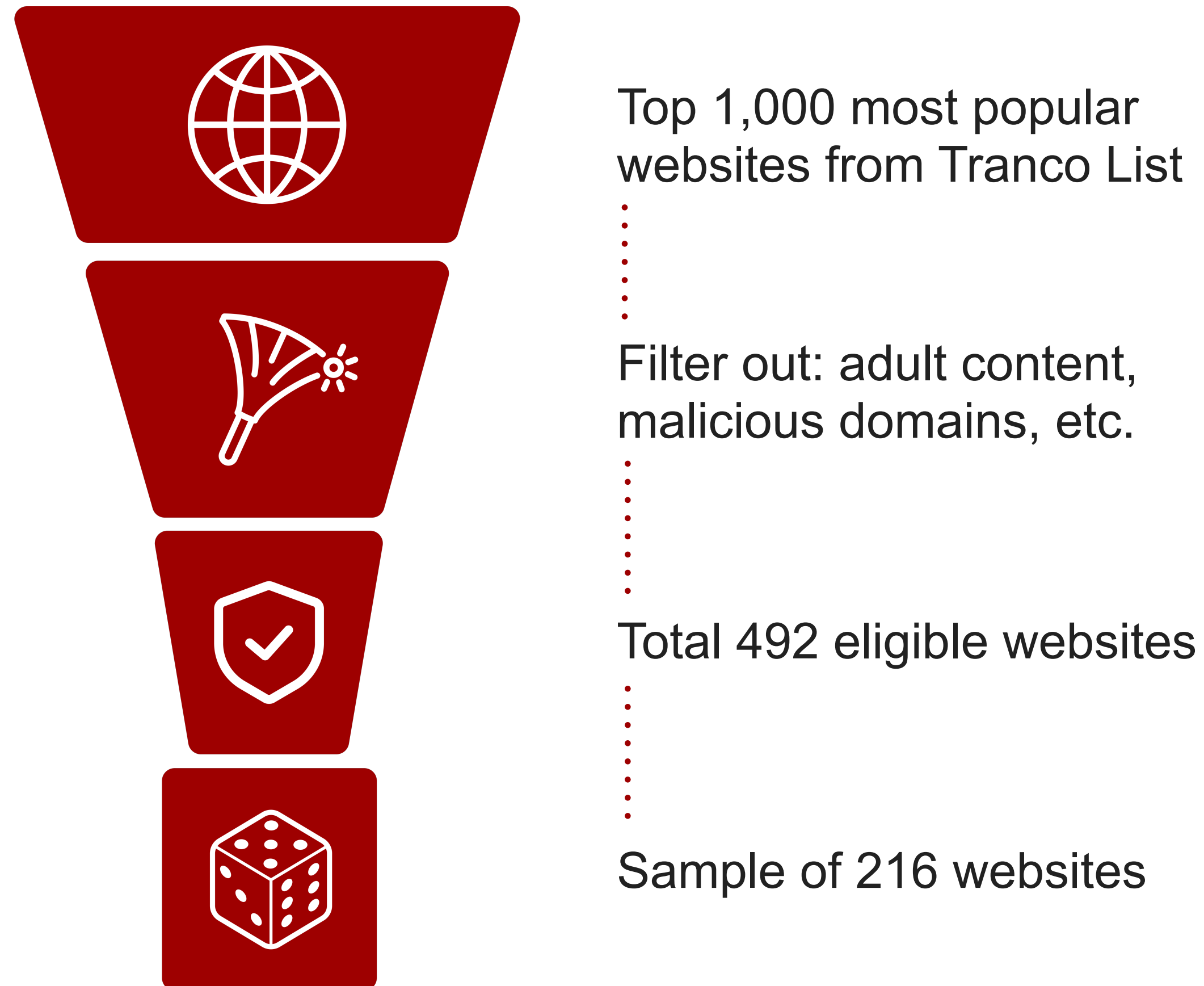
RQ1

RQ2

RQ3

RQ3

Subject Selection



Our Goal

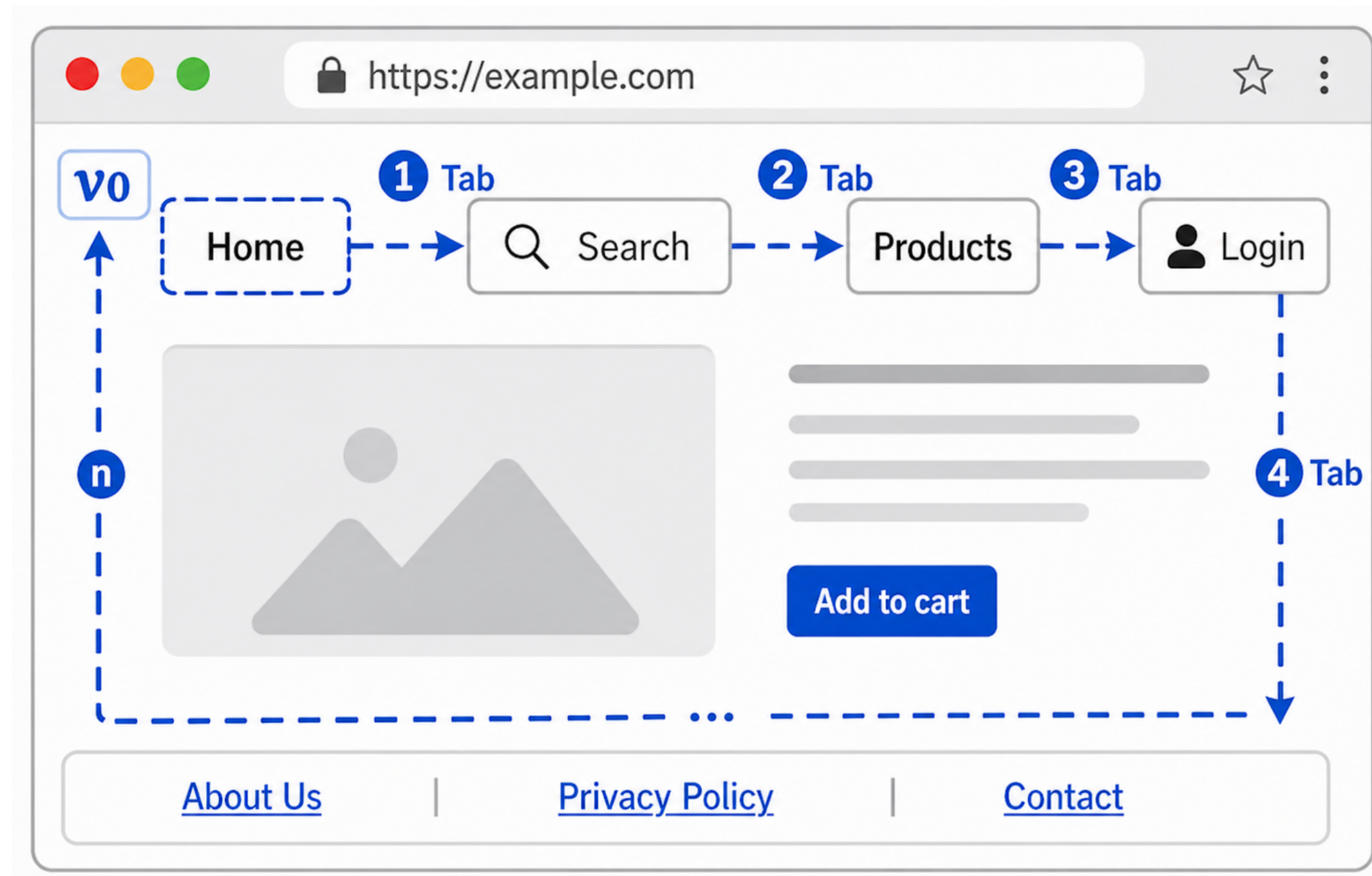
Investigate and understand the **navigation effort** and **discrepancies in UI behavior** between keyboard and mouse navigation

RQ1



What is the accessibility interaction cost for keyboard-based users to **navigate** a web page?

RQ1: Full-page Traversal Effort



Full-page Traversal

Number of keystrokes required to return to the first interactive element $v0 \dashrightarrow \dots \dashrightarrow v0$

RQ1: Results & Discussion

Avg. 125 keystrokes

3.1 minutes

to complete a full-page traversal
for keyboard-based users with a disability

1.5 sec/keystroke * 125 keystrokes = 3.1 min

RQ1: Results & Discussion

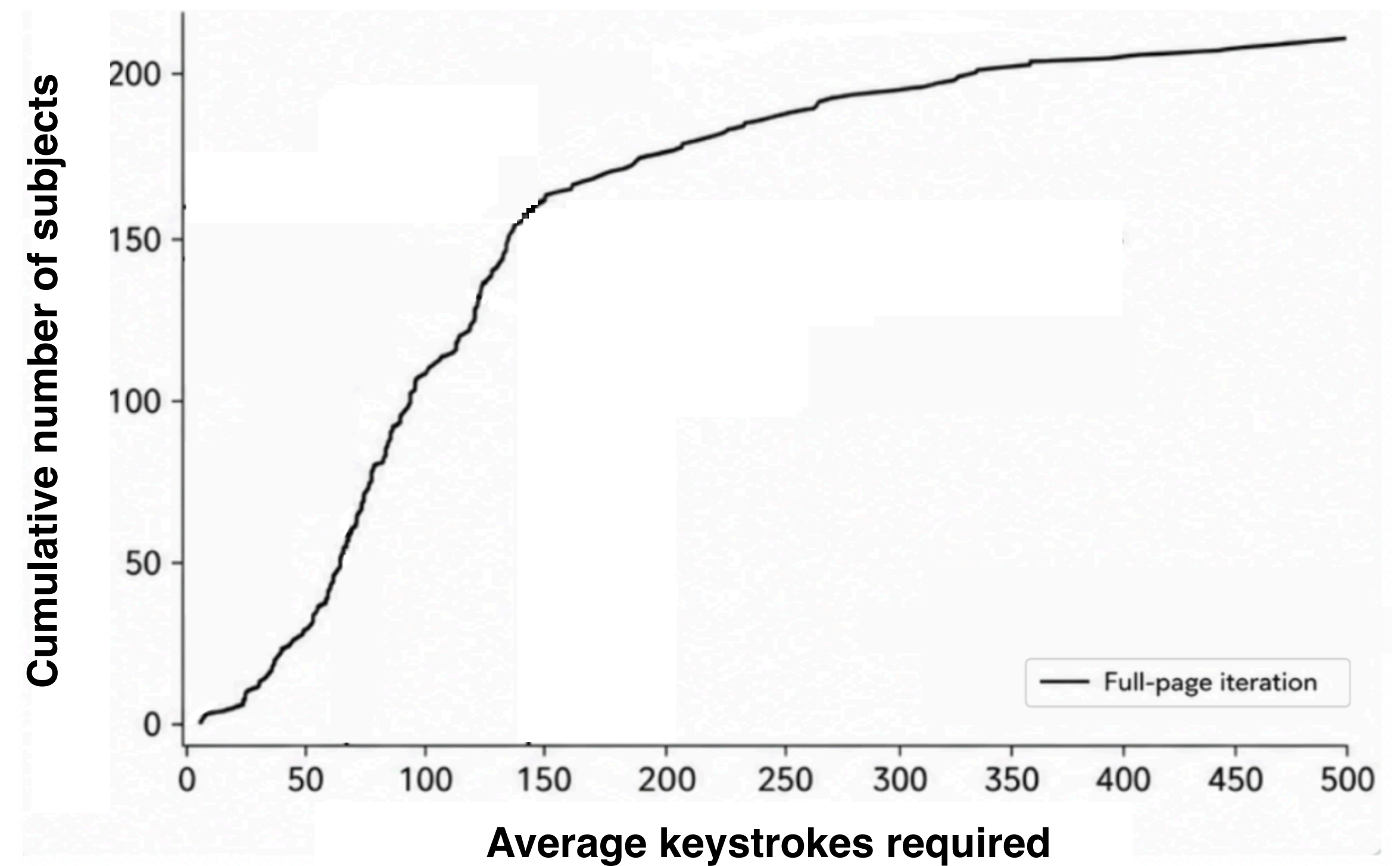
Avg. 125 keystrokes

3.1 minutes

to complete a full-page traversal
for keyboard-based users with a disability

$1.5 \text{ sec/keystroke} * 125 \text{ keystrokes} = 3.1 \text{ min}$

CDF of Average Keystrokes Required for a Full-Page Traversal



RQ1: Results & Discussion

Avg. 125 keystrokes

3.1 minutes

to complete a full-page traversal
for keyboard-based users with a disability

$1.5 \text{ sec/keystroke} * 125 \text{ keystrokes} = 3.1 \text{ min}$

25% of users

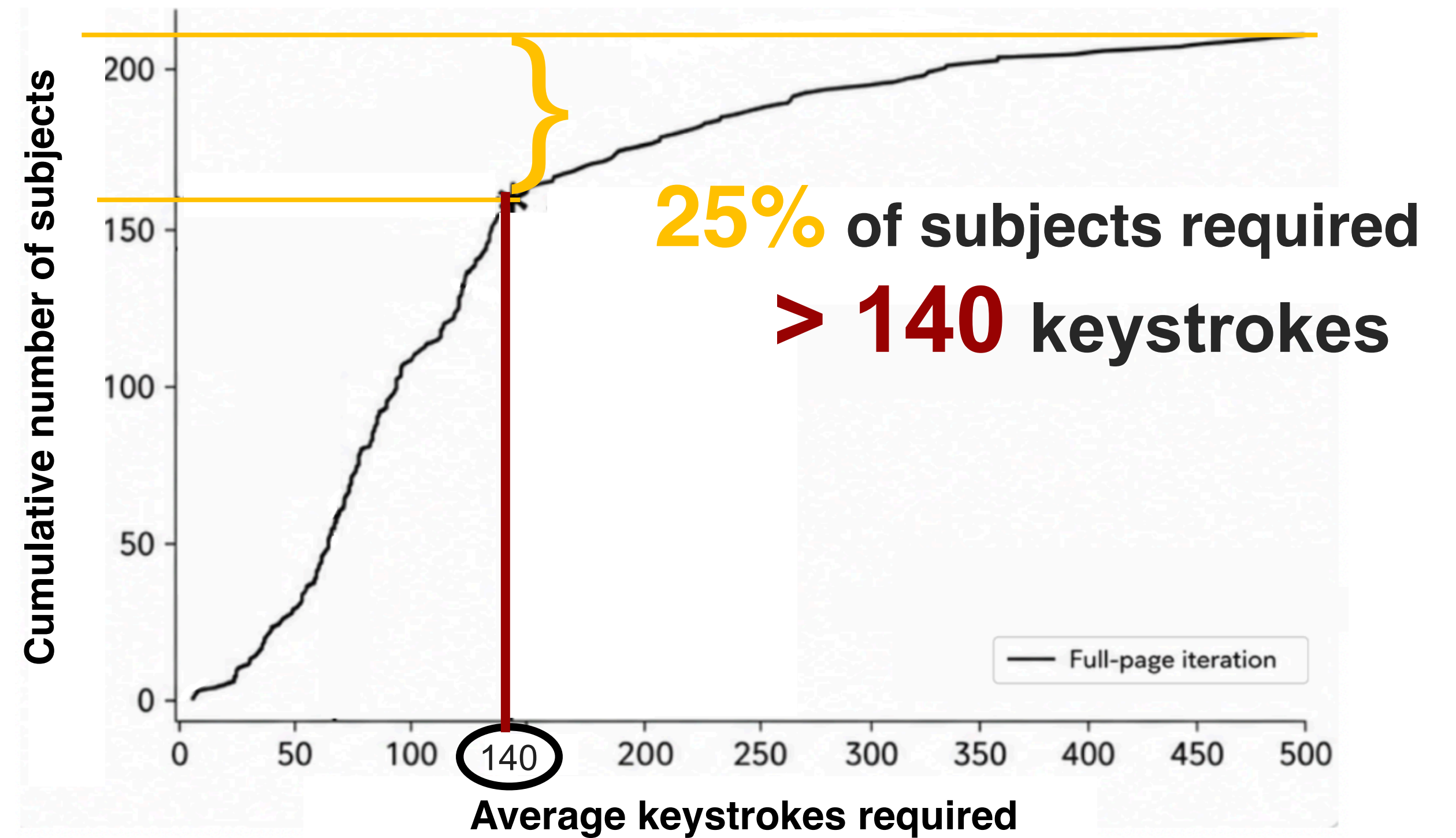
> 3.5 minutes

to complete a full-page traversal
for keyboard-based users with a disability

$1.5 \text{ sec/keystroke} * 140 \text{ keystrokes} = 3.5 \text{ min}$



CDF of Average Keystrokes Required for a Full-Page Traversal



Our Goal

Investigate and understand the **navigation effort** and **discrepancies in UI behavior** between keyboard and mouse navigation

RQ1



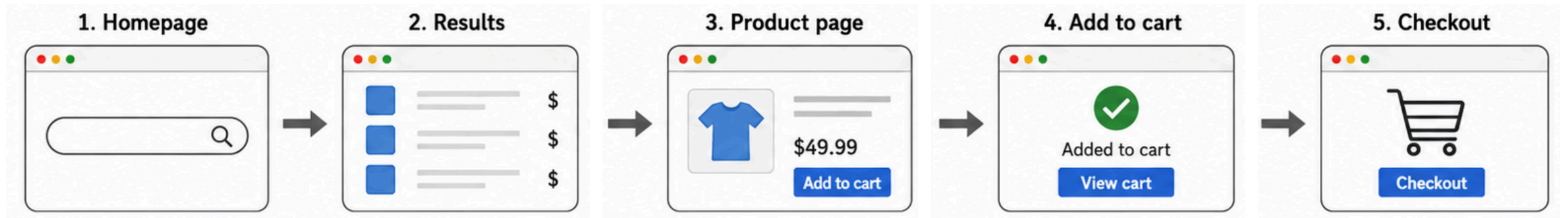
What is the accessibility interaction cost for keyboard-based users to **navigate** a web page?

RQ2

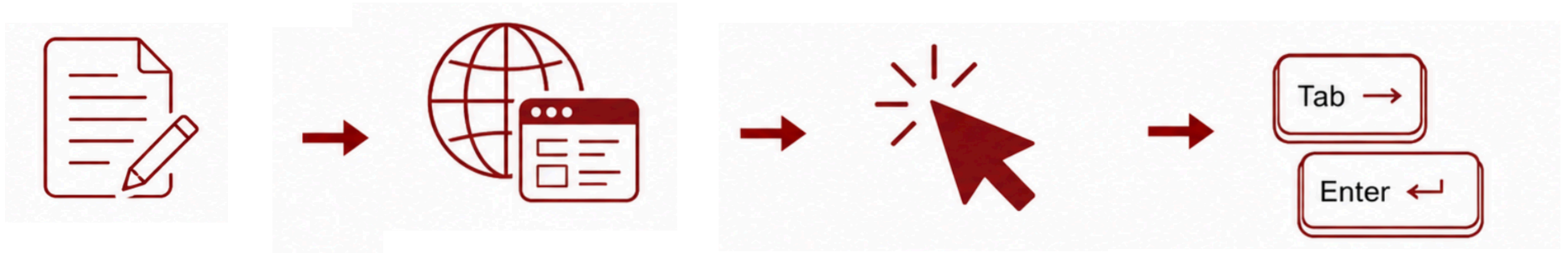


What effort is required for keyboard-based versus mouse users to **complete typical use cases** in web apps?

RQ2: Effort for Completing Typical Use-Cases



RQ2: Effort for Completing Typical Use-Cases



1

Defined **13 Use Case Categories** (e.g., Search & Purchase, etc.)

2

Chose one **individual Use Case** for each of the **216 websites** (e.g., Buying a product)

3

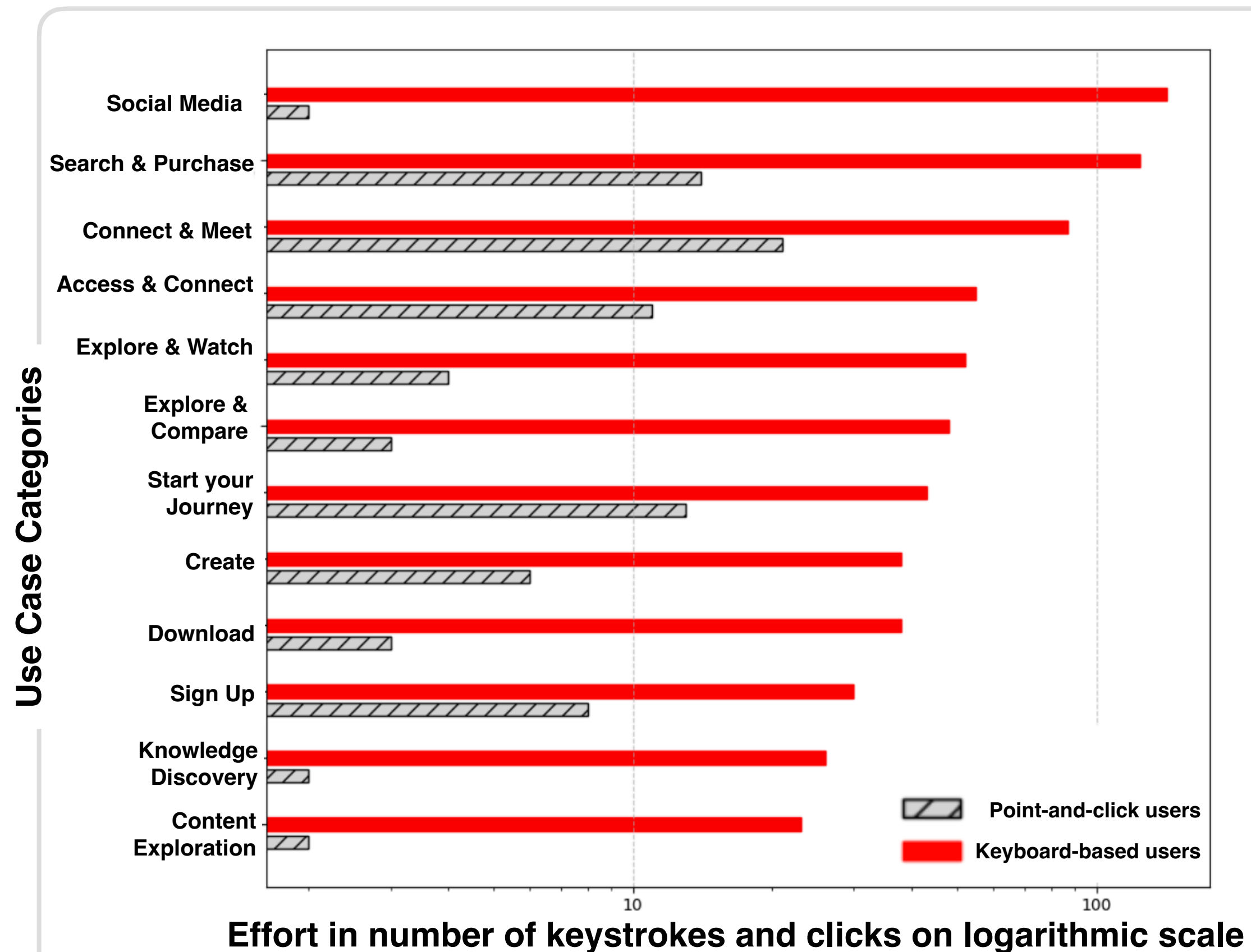
Executed each Use Case and counted **mouse clicks**

4

Executed each Use Case and counted **keystrokes**

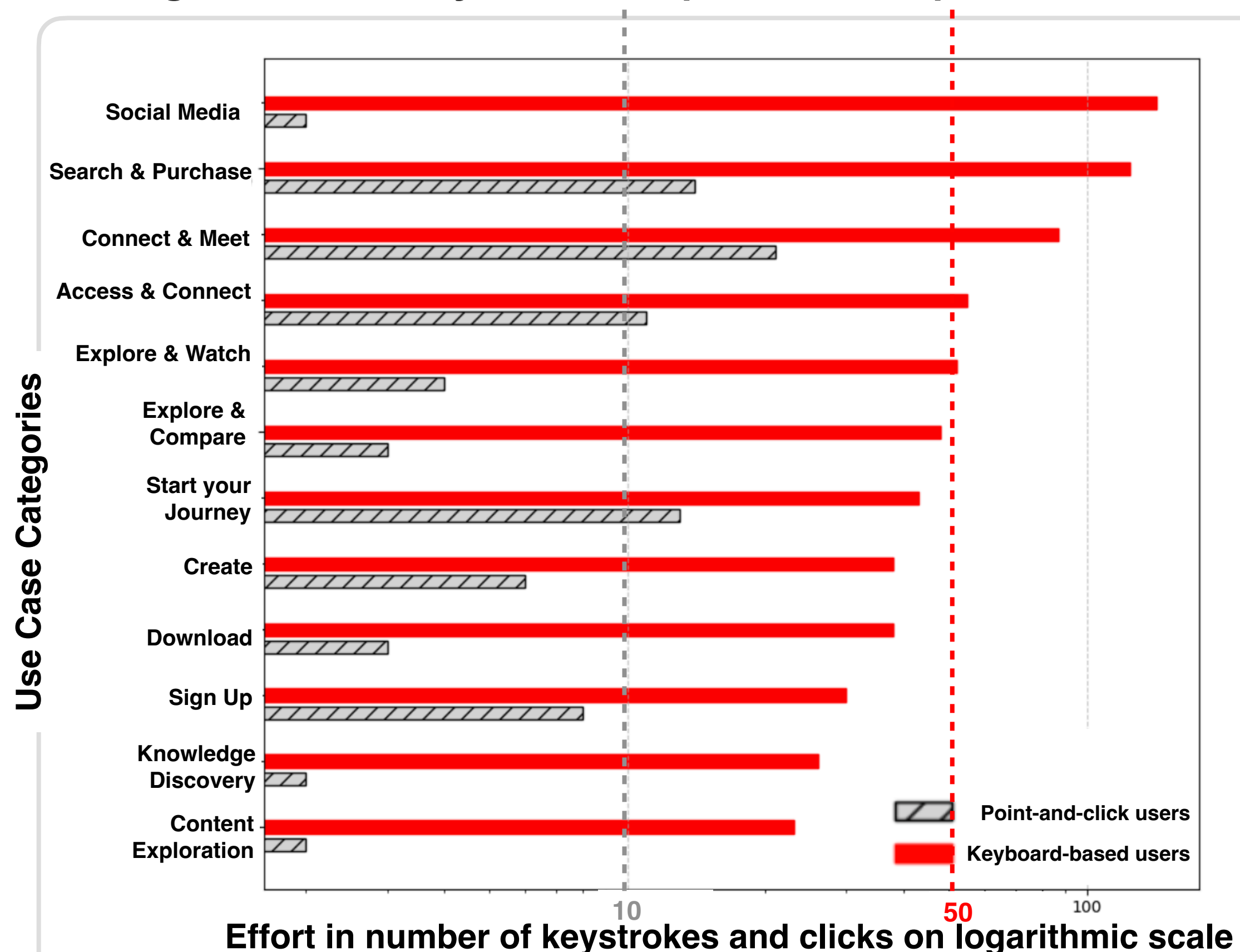
RQ2: Results & Discussion

Average clicks vs. keystrokes required to complete the use-cases



RQ2: Results & Discussion

Average clicks vs. keystrokes required to complete the use-cases



5x more effort

on average for keyboard-based users

Average: 50 keystrokes versus 10 mouse clicks



75 seconds

for a keyboard-based user with a disability

versus 11 seconds

for a mouse user

1.5 sec/keystroke * 50 keystrokes = 75 sec
 1.1 sec/click * 10 = 11 sec



Our Goal

Investigate and understand the **navigation effort** and **discrepancies in UI behavior** between keyboard and mouse navigation

RQ1



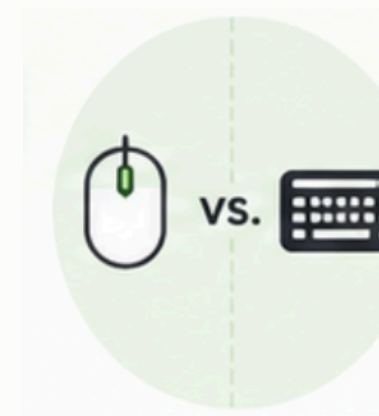
What is the accessibility interaction cost for keyboard-based users to **navigate** a web page?

RQ2



What effort is required for keyboard-based versus mouse users to **complete typical use cases** in web apps?

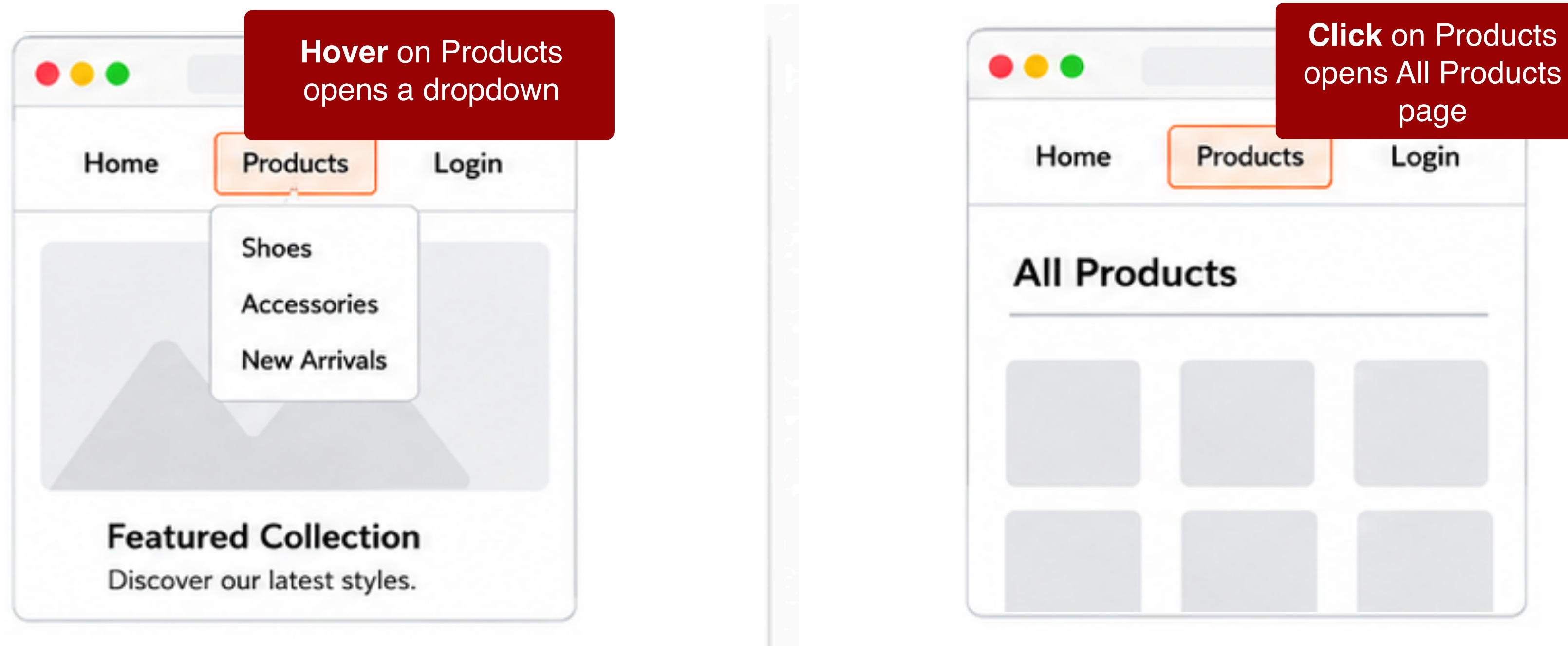
RQ3



What are the **differences in UI behavior** when interacting with a web page via the keyboard versus the mouse?

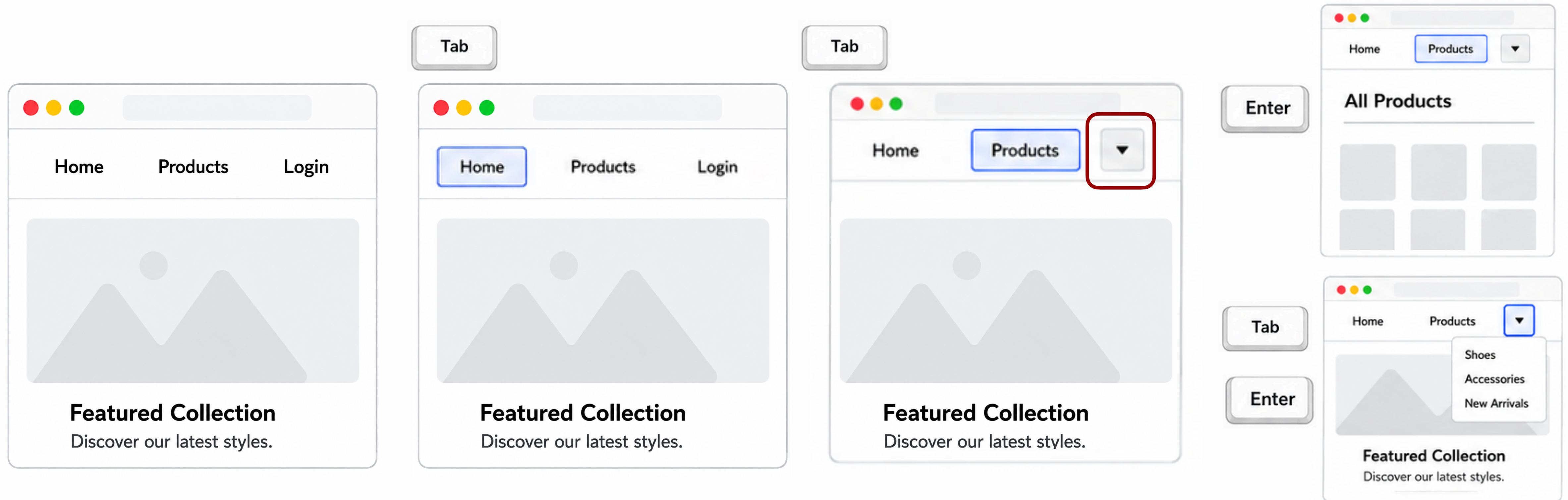
RQ3: UI Behavior Differences

Mouse Navigation



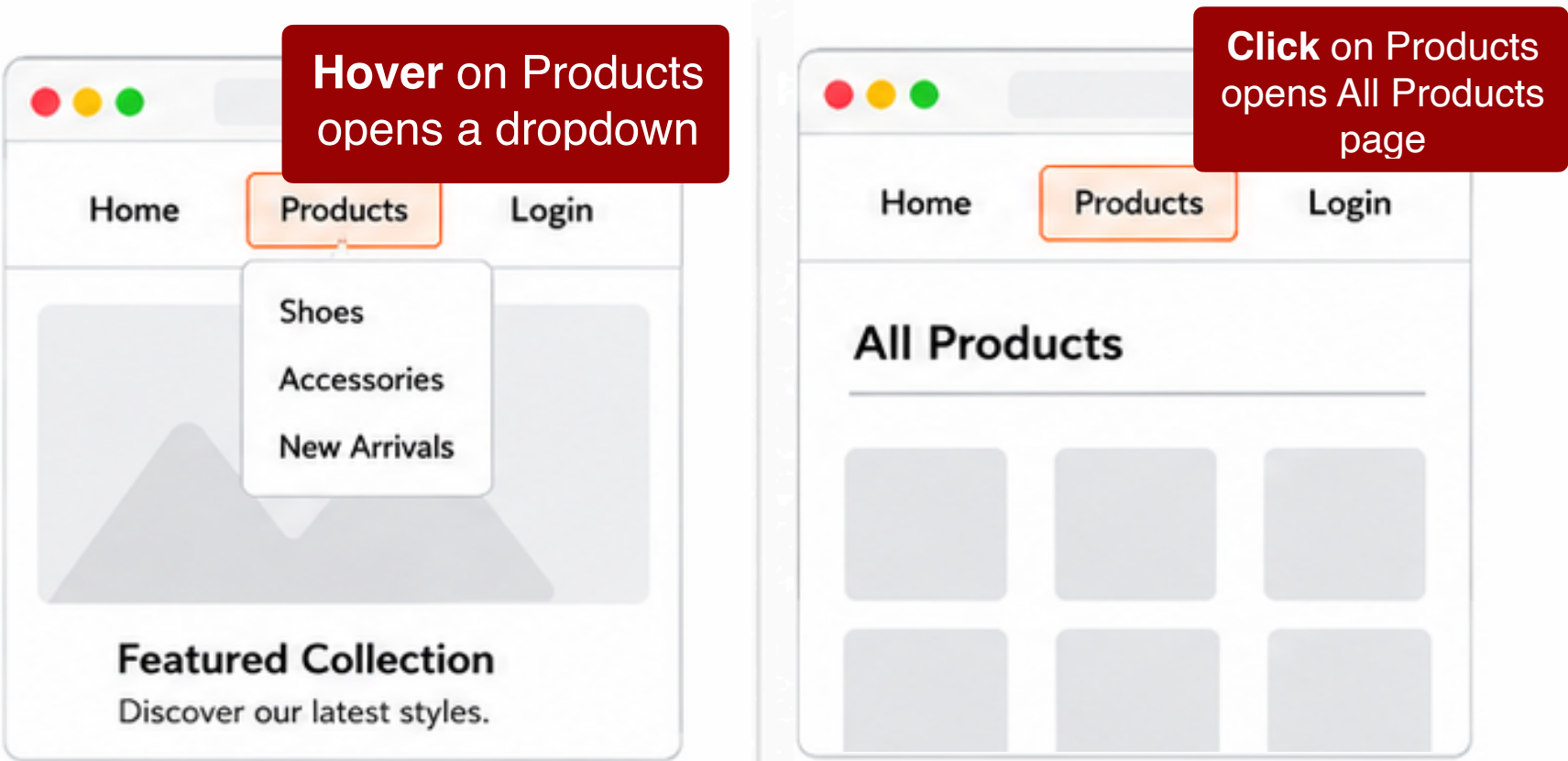
RQ3: UI Behavior Differences

Keyboard Navigation



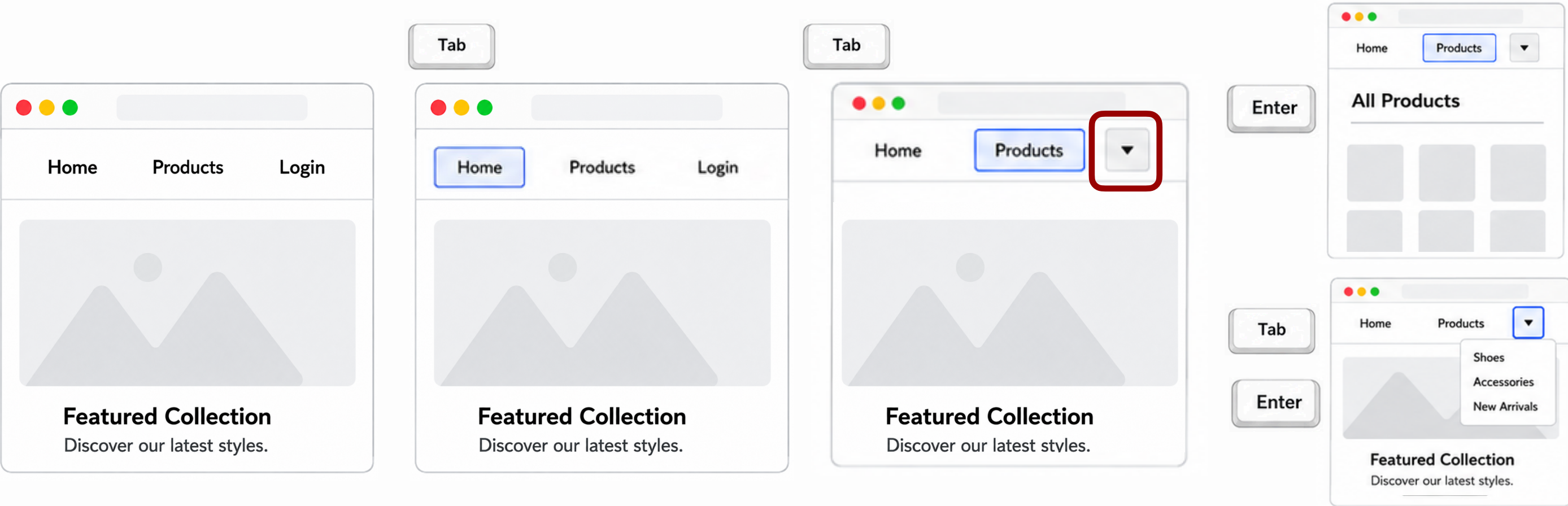
RQ3: UI Behavior Differences

Mouse Navigation



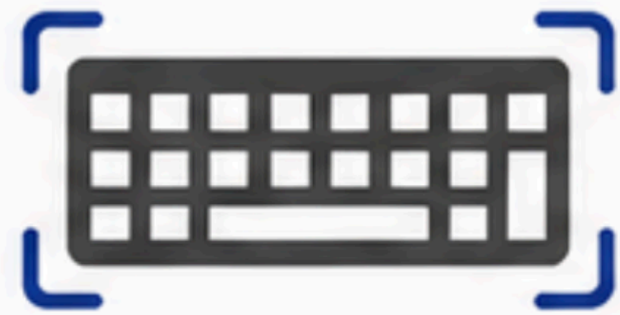
Crawljax & Qualstate

Keyboard Navigation



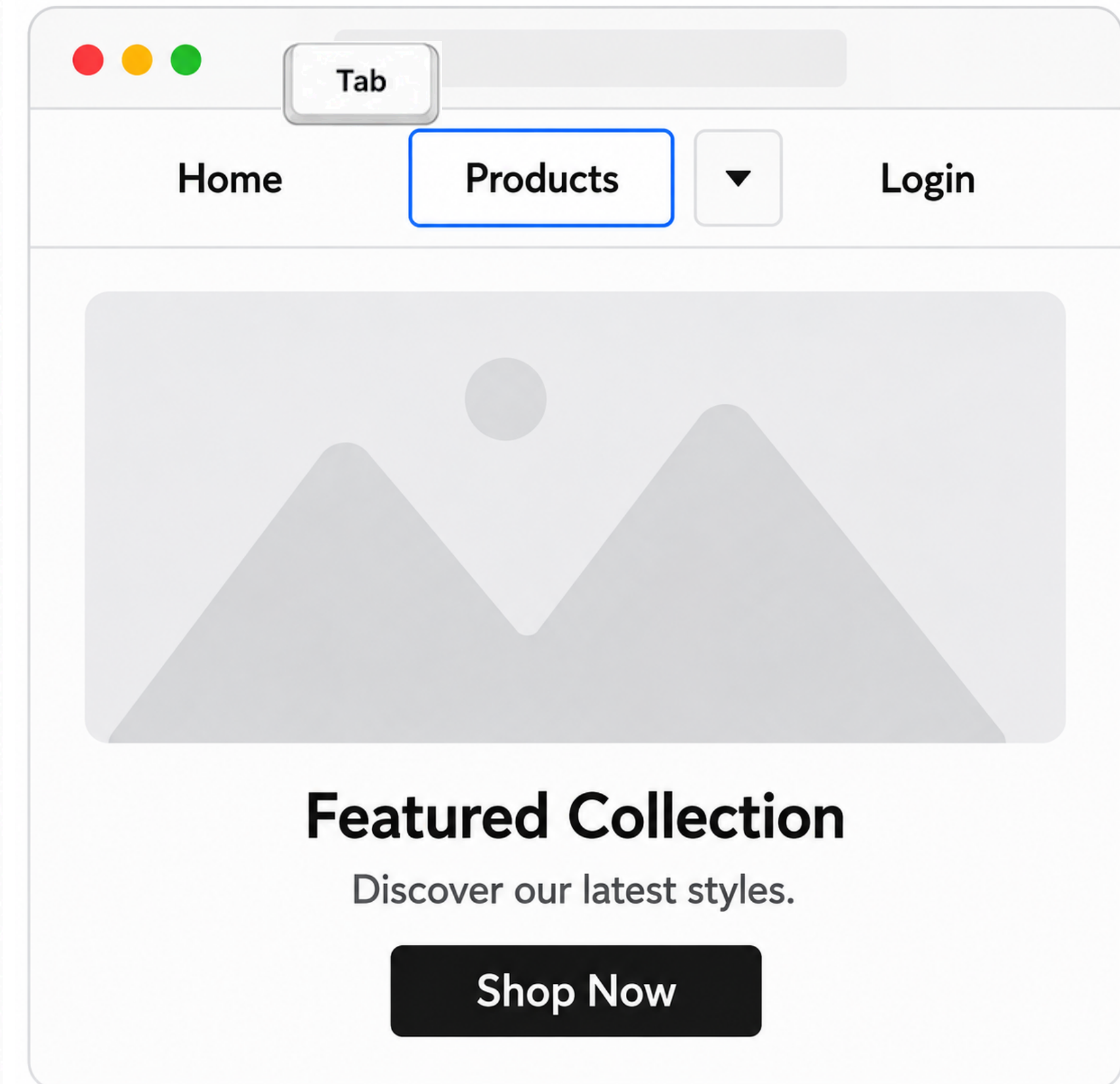
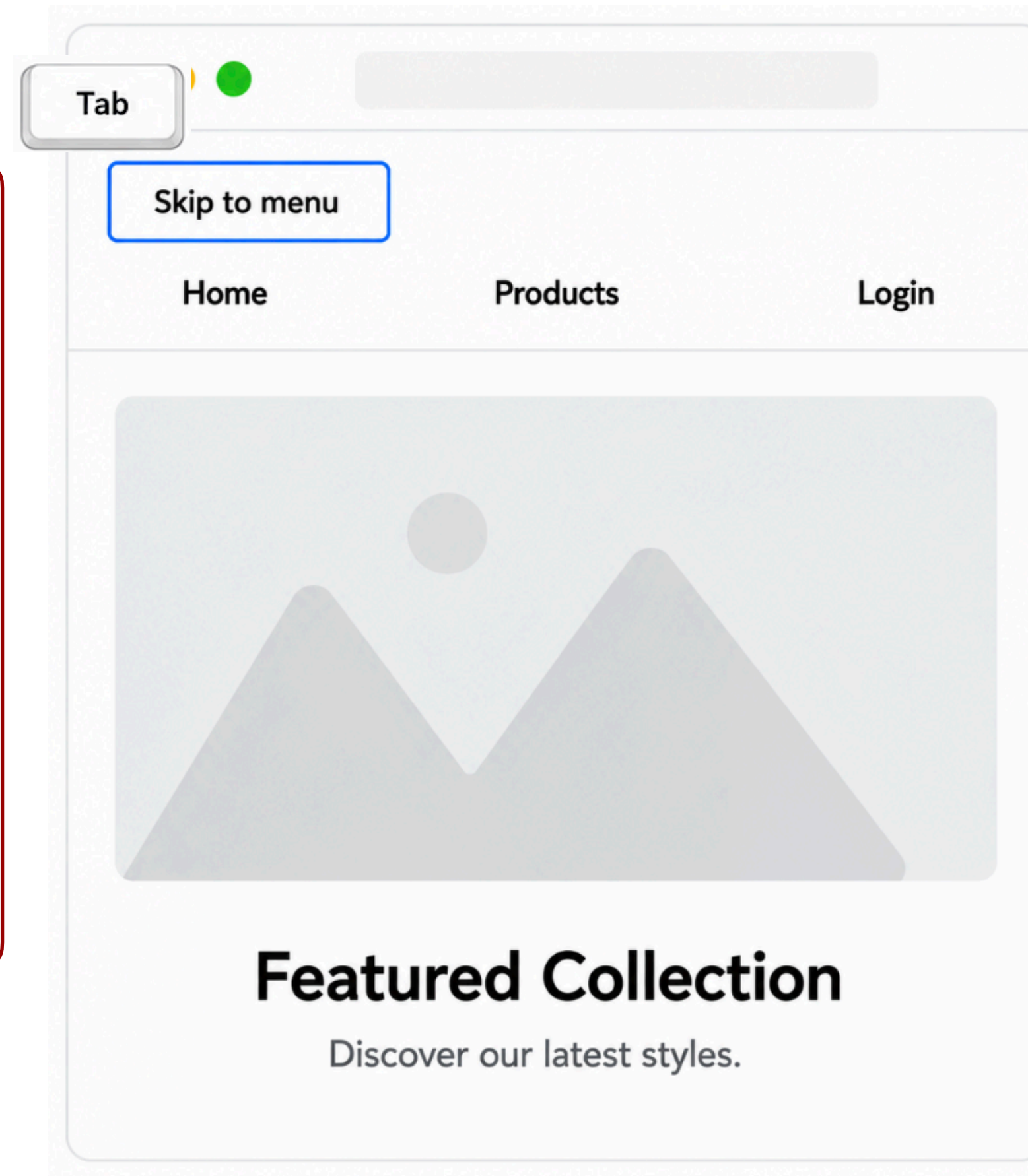
Krawler

RQ3: Results & Discussion



38% of subjects

Krawler found extra
Keyboard-only states



RQ3: Results & Discussion



19% of cases

Mouse-only states **inaccessible**
by keyboard

Welcome Back

Please sign in to your account

Username

Password

Log In

Tab

!

Focus Problem

Welcome Back

Please sign in to your account

Username

Password

Log In

Cannot activate
using **ENTER**

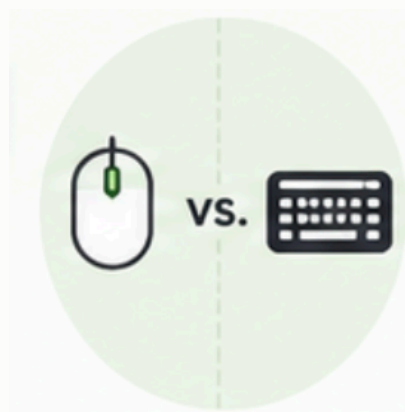
Conclusions

RQ1

A keyboard-based user needs on average **125** keystrokes to **navigate** a web page

RQ2

To **complete typical use cases**, a keyboard-based user needs on average **5 times more** effort compared to mouse users

RQ3

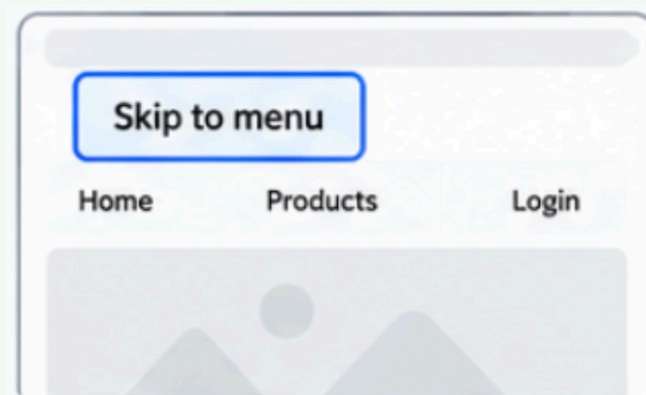
Keyboard-based and mouse users often experience **differences in UI behaviors**

Design Patterns & Antipatterns

What helps or hurts keyboard navigation?

Design Patterns

1



Include skip content links

Reduce effort on full-page traversal ~15% (RQ1) and completing tasks ~50% (RQ2)

2



Use focus shift

Support smoother task flows ~77% (RQ2)

3

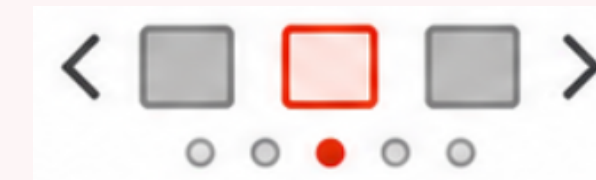


Use native HTML elements

Ensure built-in keyboard accessibility (RQ3)

Design Antipatterns

1



Avoid grouped components with no escape mechanism

Increase keyboard effort ~27% (RQ1)

2



Avoid infinite scroll

Hard to reach footer content (RQ1) and complete tasks (RQ2)

3



Avoid mismanaged keyboard events

Keyboard-based users could be stuck in navigation (RQ1)



Assessing the User Interaction Cost of Keyboard Navigation in Web Applications

Christina Z. Chaniotaki, Paul T. Chiou, William G.J. Halfond

email: cchaniot@usc.edu

Thank you!

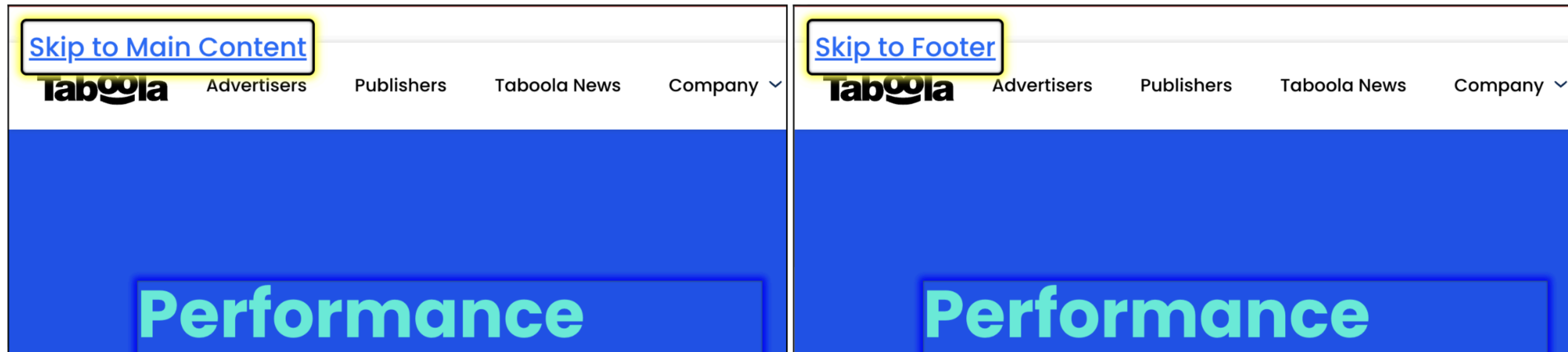


[Link to the paper](#)

References

- **Slide 2:** World Health Organization, “Disability,” 2023, accessed on November 28, 2023. [Online]. Available: <https://www.who.int/news-room/fact-sheets/detail/disability-and-health>
- **Slide 7 Images:** <https://accessive.com/blog/screen-reader-accessibility/>, <https://www.personneltoday.com/hr/freedom-speech-voice-recognition-software-healthier-way-working/>, <https://microsoft-surface.ru/xbox-adaptive-controller/>, <https://www.caregiverproducts.com/typing-aid.html>
- **Slide 9-11:**
 1. C. R. D. U.S. Department of Justice, “Guidance on web accessibility and the ada,” 2022, accessed on May 28, 2025. [Online]. Available: <https://www.ada.gov/resources/web-guidance>
 2. P. T. Chiou, A. S. Alotaibi, and W. G. J. Halfond, “Detecting and localizing keyboard accessibility failures in web applications,” in Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '21), August 2021.
 3. W. A. R. W. M. Isa, A. I. H. Suhaimi, N. Ariffn, N. F. Ishak, and N. M. Ralim, “Accessibility evaluation using web content accessibility guidelines (wcag) 2.0,” in Proceedings of the 4th International Conference on User Science and Engineering (i-USER), August 2016.
- **Tools:** **Krawler:** <https://github.com/USC-SQL/krawler>, **Crawljax:** <https://github.com/crawljax/crawljax>, and **Qualstate:** <https://github.com/qualweb/qualstate>
- **Slide 19:**
 - Tranco List: V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, “Tranco: A research-oriented top sites ranking hardened against manipulation,” in Proceedings of the 26th Annual Network and Distributed System Security Symposium, 2019.
 - For a statistically representative sample, we used the Cochran sample size formula with 95% confidence and 5% margin of error. W. G. Cochran, Sampling techniques, 3rd ed. John Wiley & Sons, Inc, 1977.

Skip to Content Example

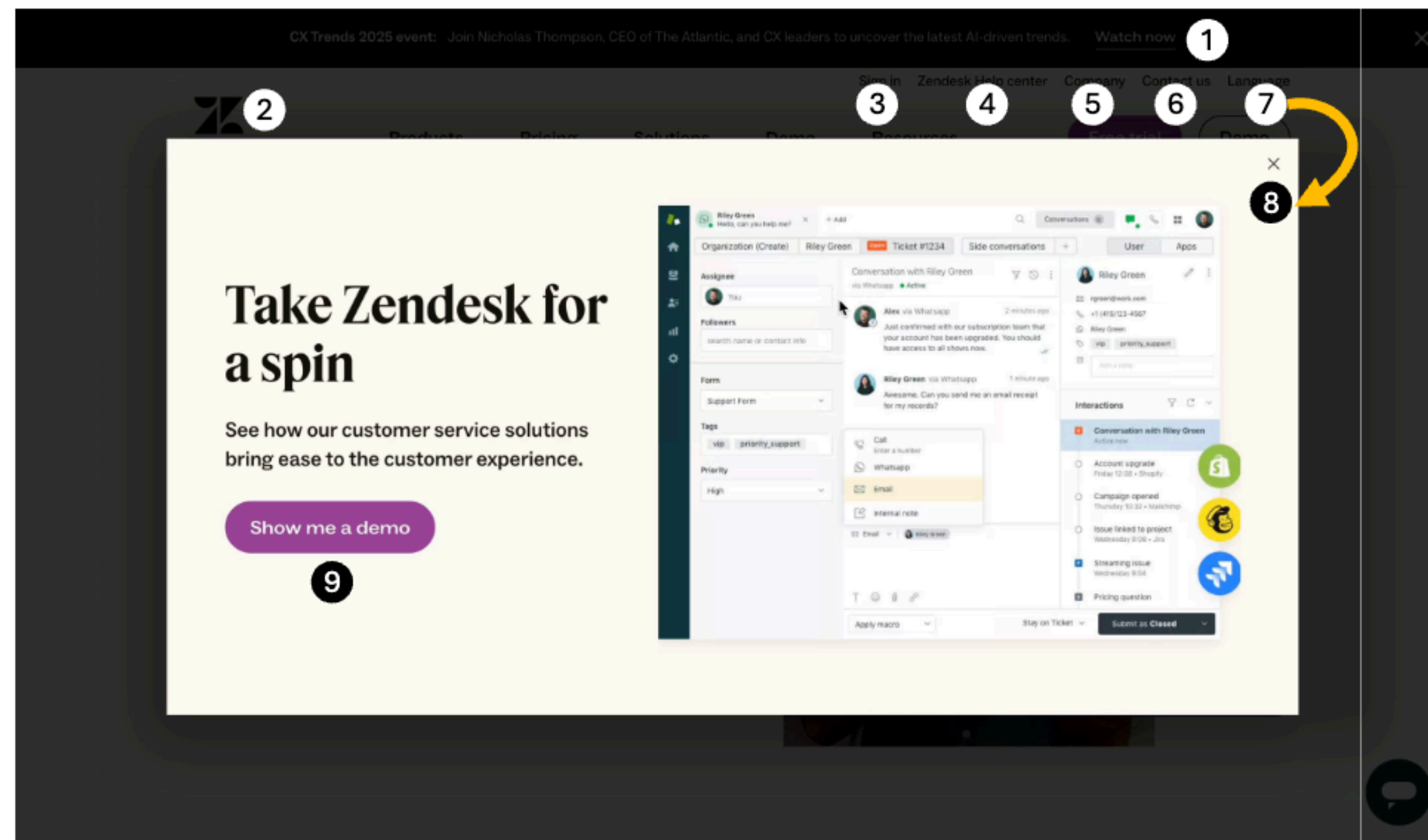


(a) First button link saves 32 keystrokes to Main Content

(b) Second button link saves 83 keystrokes to Footer

Skip to content feature enhances keyboard navigation by allowing users to bypass sections of the web page

Use of Focus Shift



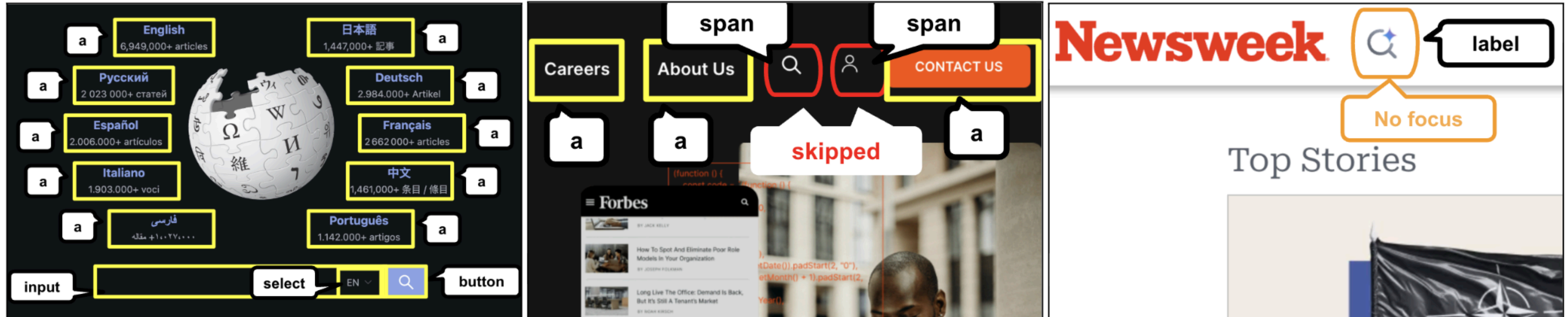
(a) Key focus automatically shifts into dialog (*Zendesk*)



(b) Key focus does not automatically shift into dialog (*Autodesk*)

Moving keyboard focus to a dialog when it appears, reducing effort for keyboard users.

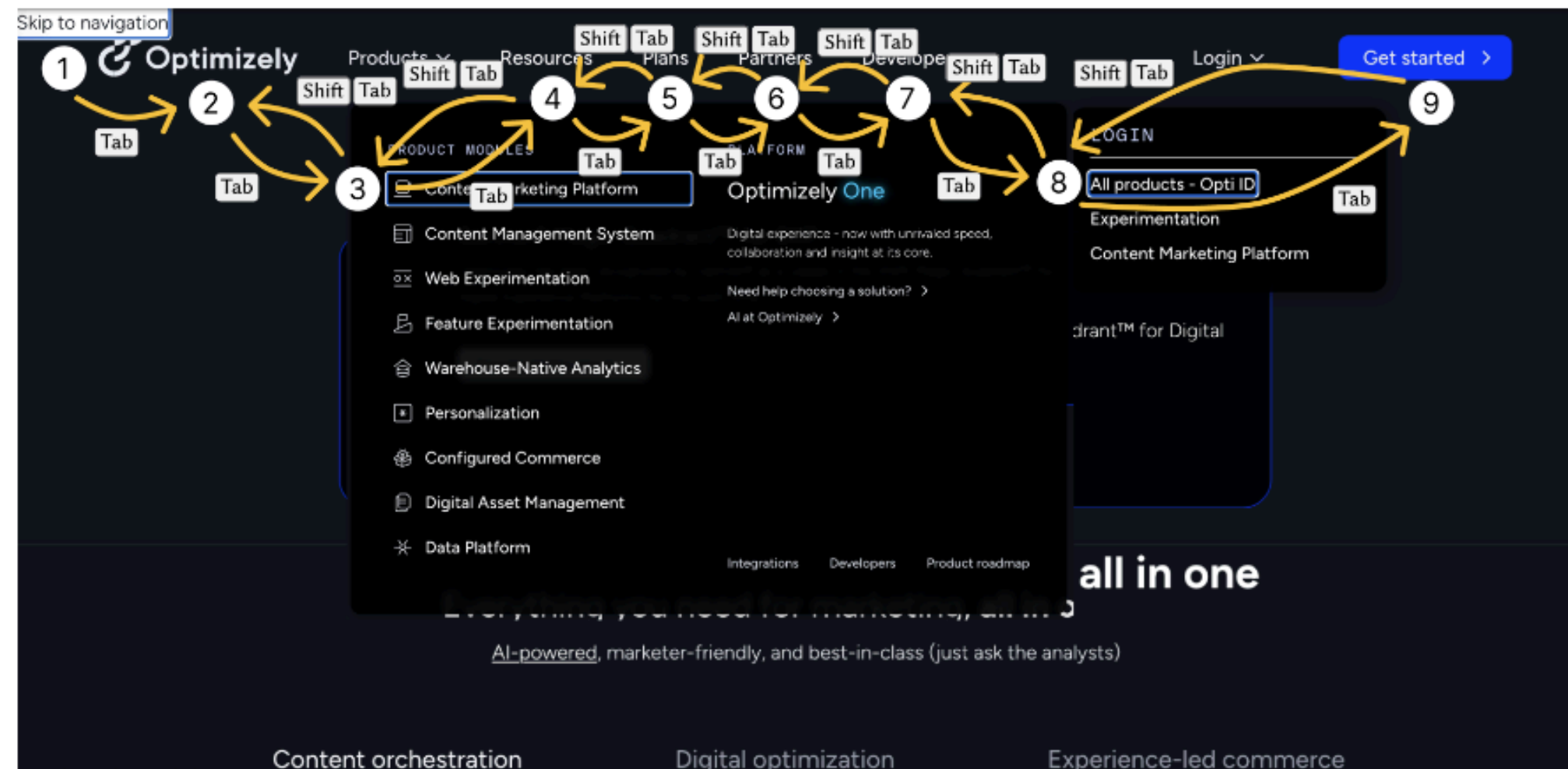
Use of Native HTML Elements



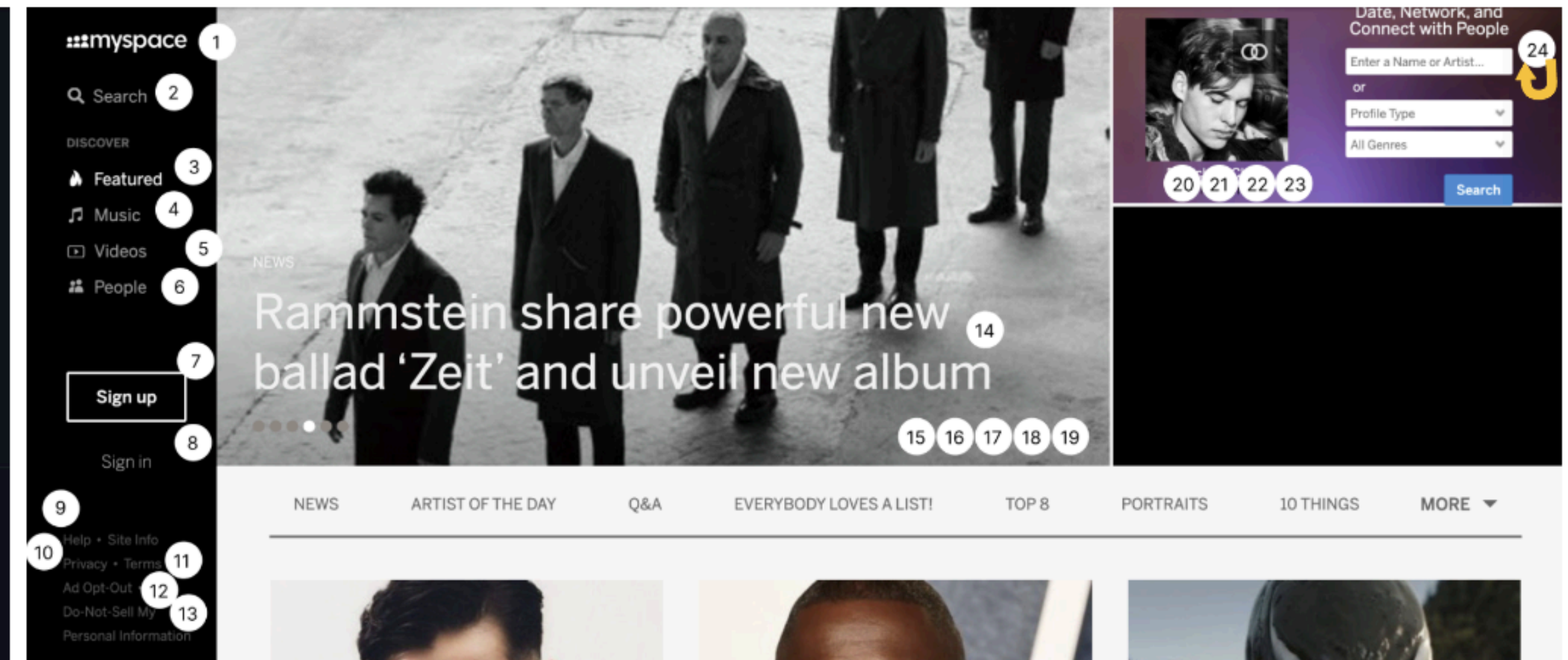
(a) Focusable and actionable (*Wikipedia*) (b) Not focusable, not actionable (*Equativ*) (c) Not focusable, actionable (*Newsweek*)

Standard HTML controls **provide built-in** keyboard accessibility (a).
Non-standard HTML controls can **hinder** keyboard navigation (b, c).

Examples of Keyboard Traps



(a) Focus trapped within elements 2–9, with no way to exit using **Tab** or **Shift**+**Tab** (*Optimizely*)



(b) Focus remains trapped in element 24 until the input is completed, preventing escape (*Myspace*)

Keyboard traps can make parts of, or even the entire web page, inaccessible to keyboard users

Improperly hidden interactive elements, could increase the keyboard navigation effort

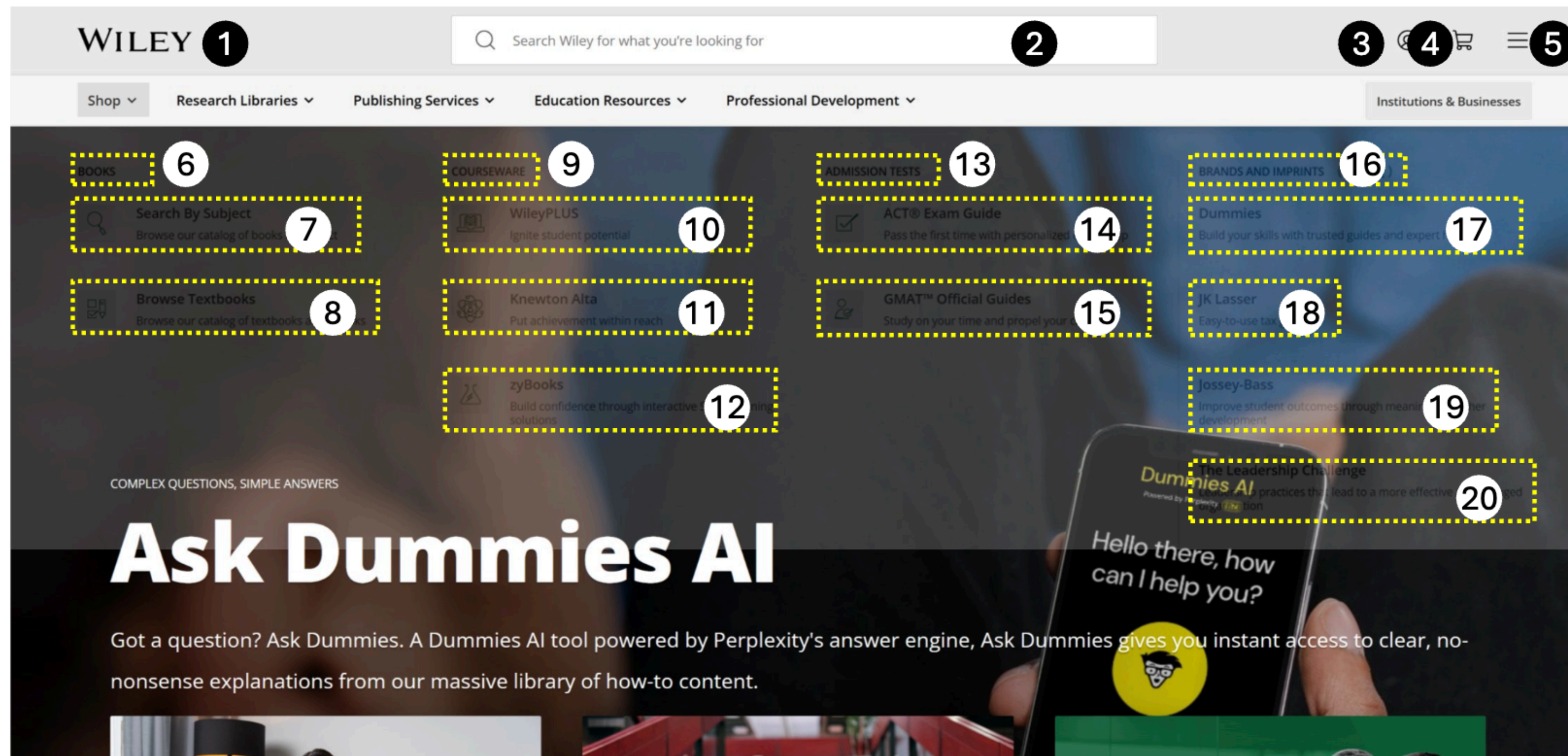


Fig. 8. Wiley home page showing improperly hidden sub-menu items in the top navigation menus of the page

Infinite scroll increase the keyboard navigation effort

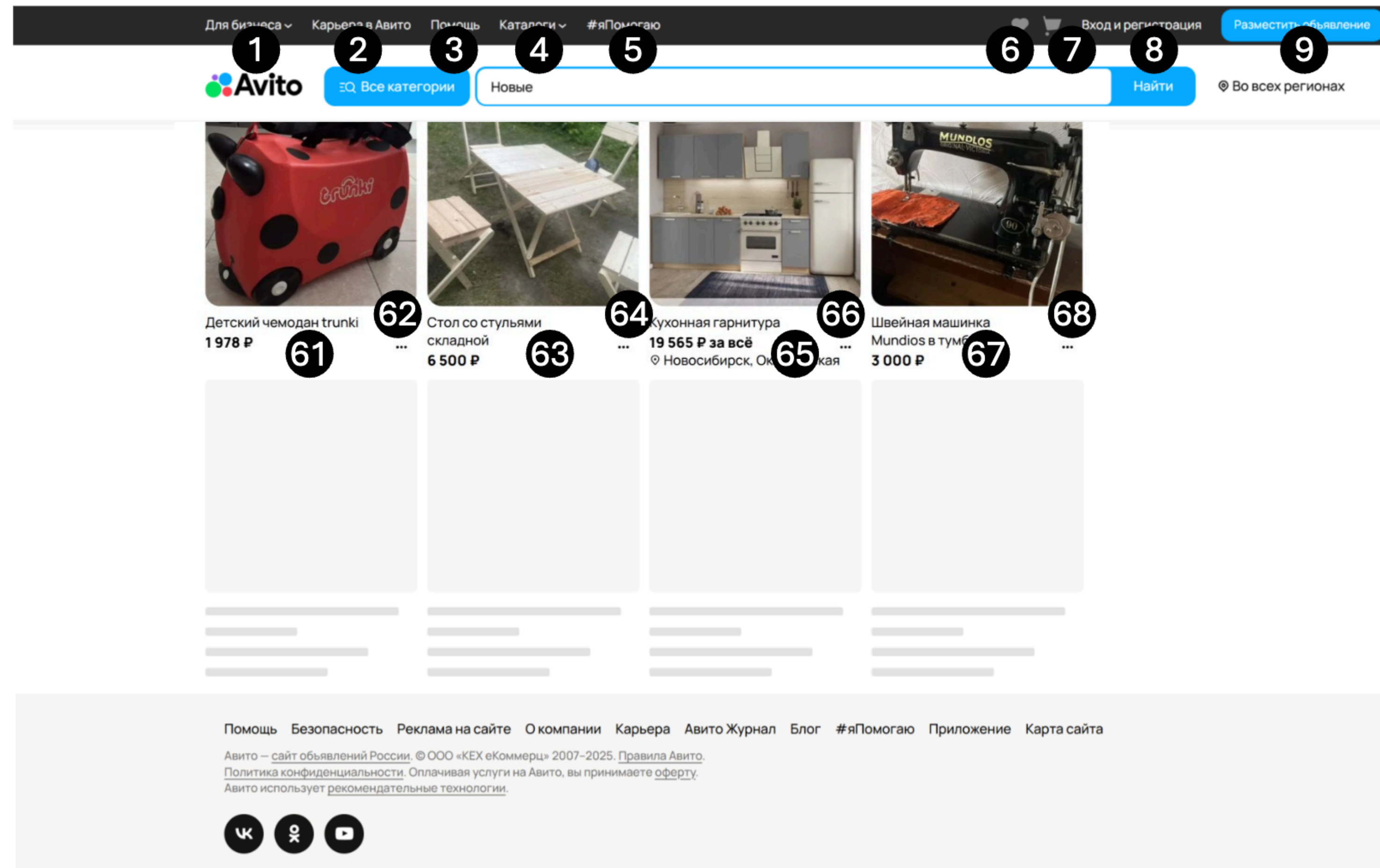
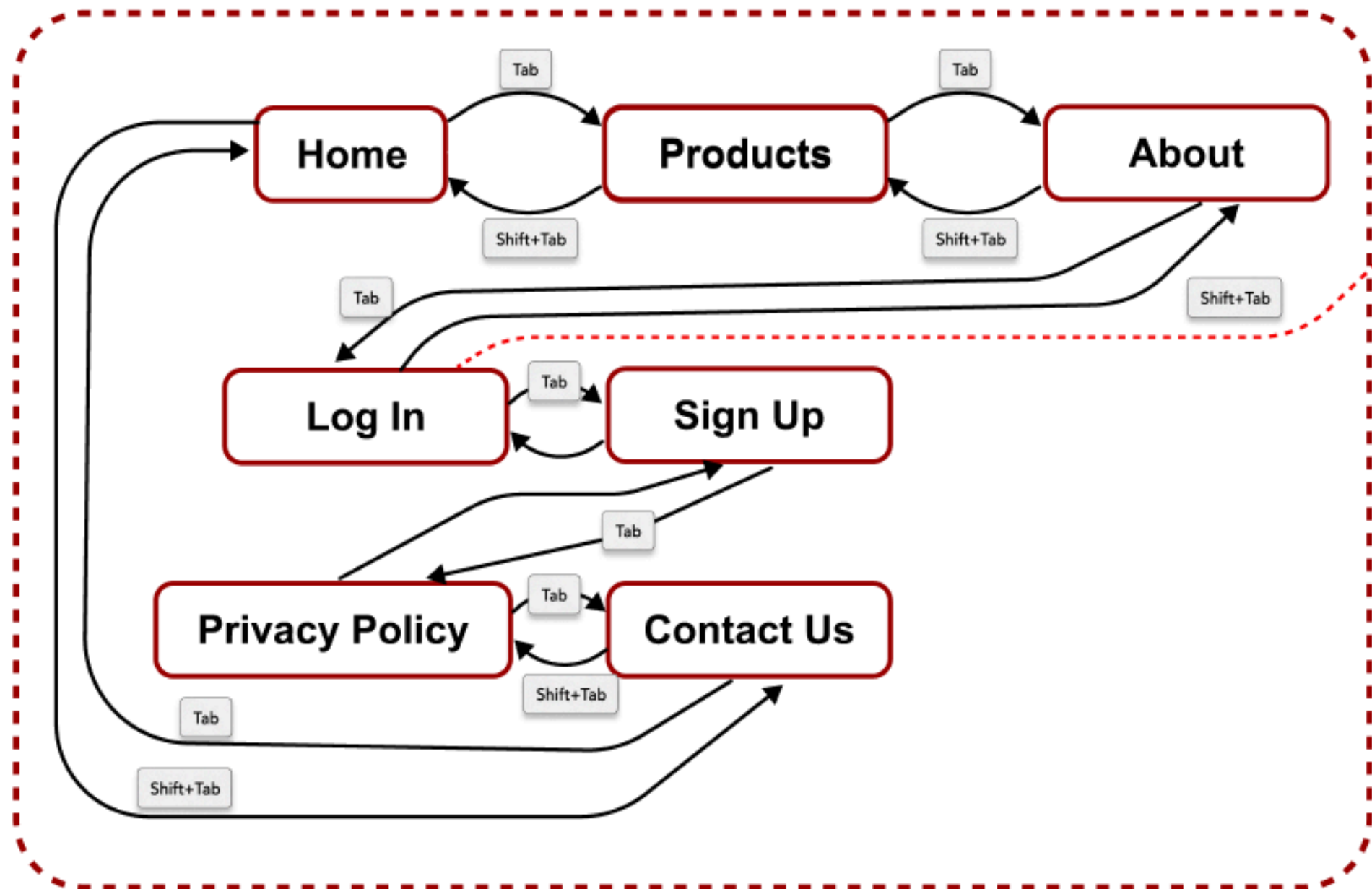


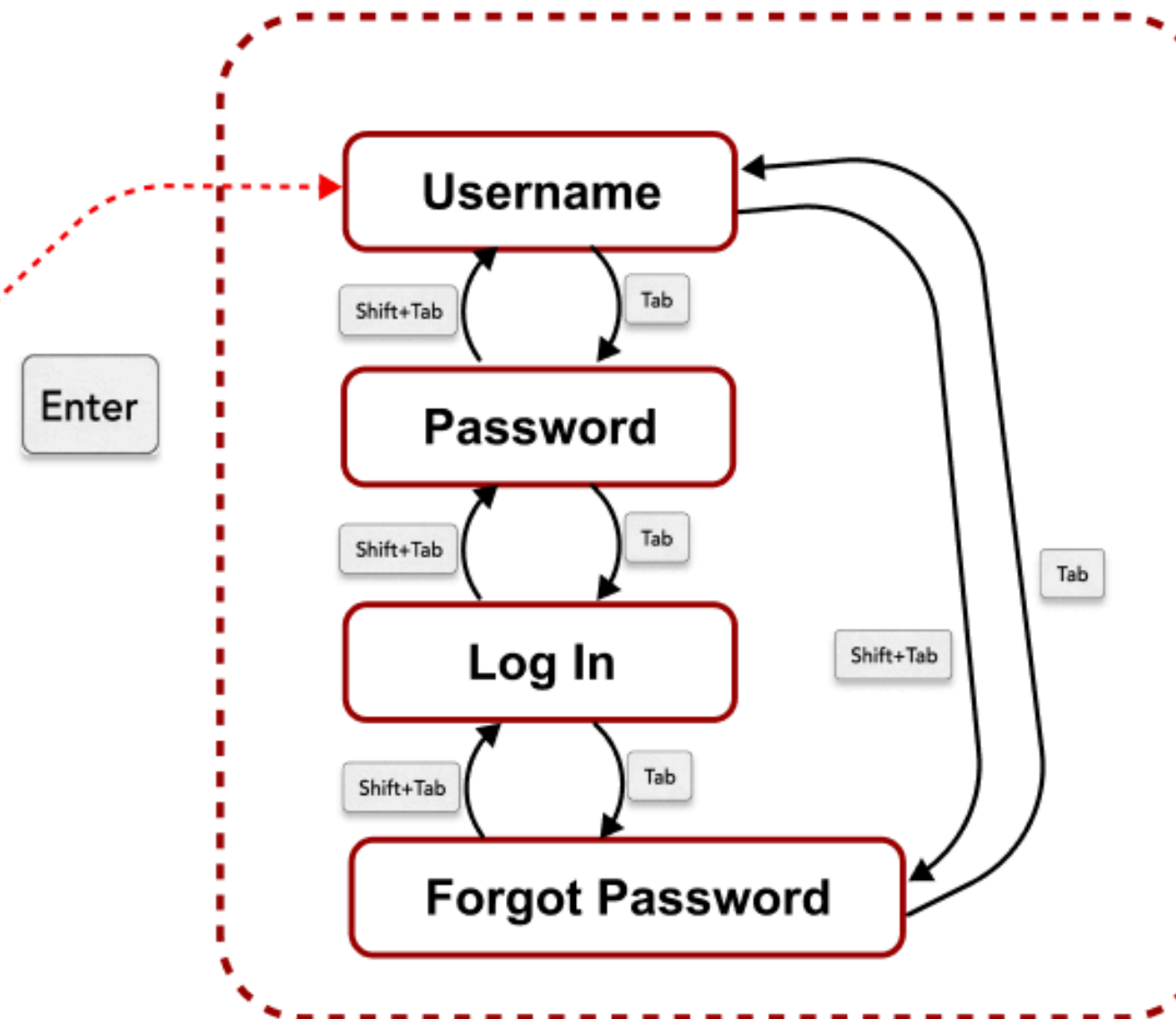
Fig. 9. Avito home page showing infinite scrolling component to constantly load new product listings to the page

The Keyboard Experience Representation (KER) Model

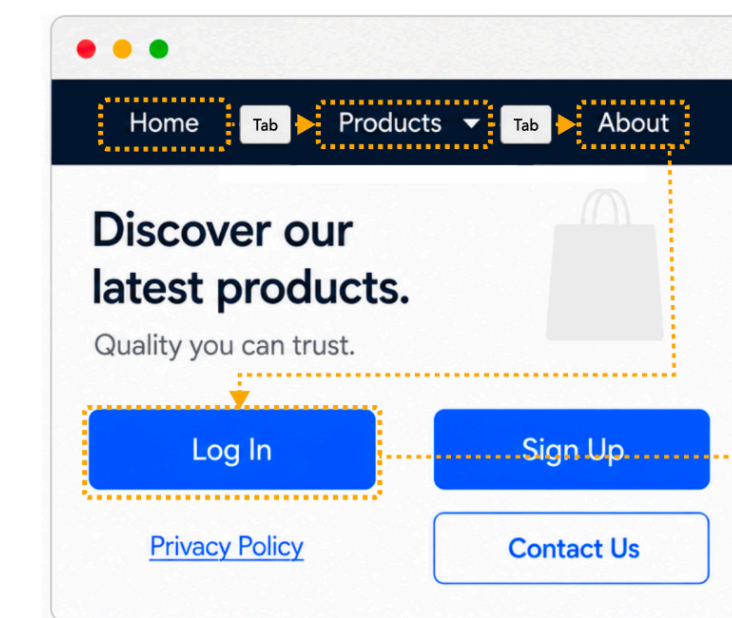
UI State 1



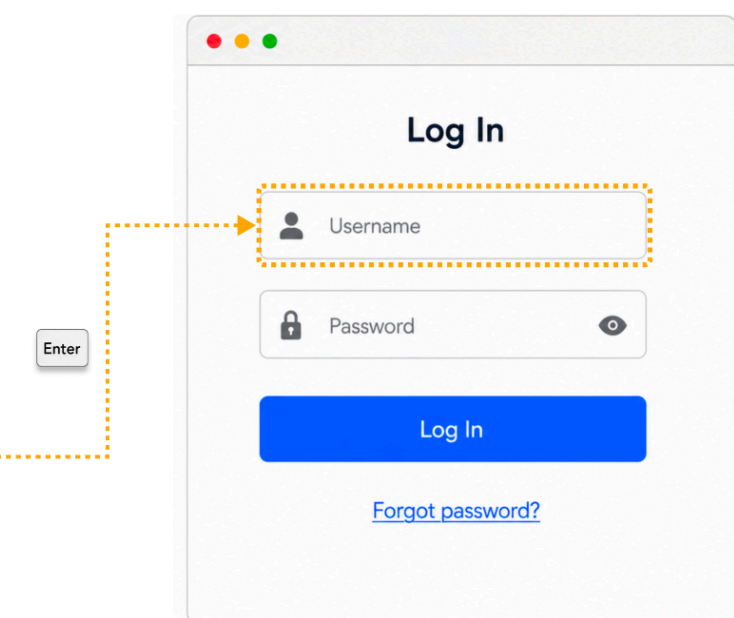
UI State 2



UI State 1



UI State 2



= Interactive element
 = Intra-state edge
 = Inter-state edge