**From desktop to mobile:**

**An evaluation of heuristics in user experience and analysis of *Spotify* and *YouTube***

Jeeho Hong

**1. Introduction**

Mobile devices such as smartphones and tablets have quickly come to rival desktop computers as the primary platform for accessing web-based services, and it is now standard practice for desktop sites and programs to have some sort of mobile equivalent. However, when it comes to the joint development of mobile and desktop interfaces for a singular product, considerations for usability and accessibility can vary greatly between the two versions. With this in mind, how do considerations for accessibility and usability differ between desktop and mobile versions of the same service? The practice of addressing the unique requirements of each platform has not yet been standardized, and different services can be observed utilizing their own methodologies to achieve this goal. In this paper, I will attempt to answer the aforementioned question by comparing two popular streaming platforms that offer both desktop and mobile iterations of their interface: music streaming service *Spotify,* and video sharing site *YouTube*.

**2. Methodology**

In section 3 of this paper, I will consult and assess a selection of peer-reviewed articles discussing various heuristics being utilized in both desktop and mobile interface development today. Section 4 will consist of a thorough analysis of two popular streaming services that offer both desktop and mobile iterations: *Spotify* and *YouTube*. Both versions of these sites will be graded for accessibility and usability, utilizing a grading rubrik consisting of various categories based on the findings from section 3. Each interface will be graded in each category on a scale of 1 to 3, as well as a cumulative score for gauging the overall user experience of each interface. In section 5, these scores will be compared and analyzed in order to provide a final assessment of how effectively each site is able to achieve desktop-mobile parity. Finally, section 6 will deliver a recommended list of the heuristics I observed to be most universally applicable in joint development for desktop and mobile interfaces.

**3. Related work**

In order to understand how a desktop or mobile interface might be optimized for accessibility and usability, we must first understand the key differences between them. Screen size is perhaps the most glaring of differences, as the average desktop monitor or laptop screen is considerably larger than the average smartphone or tablet. Mobile devices consequently have much less space to work with, which can make it difficult to fit as much information onto a mobile interface as one could on desktop. Input methods are another significant consideration, as the touch screens of mobile devices require unique interactions compared to the mouse, trackpad, or keyboard of a computer. Styluses and assistive input devices can also be used with either of these devices, each of which involve their own unique quirks that can impact usability (Cockburn, et al). Mobile devices are also much more limited in areas like bandwidth and computational resources, generally making for a much more restrictive platform to develop for when compared to desktop web development (Billi, et al.).

**3.1. Defining usability and accessibility**

Billi, et al. observe that definitions for usability and accessibility can vary significantly between institutions. After comparing numerous definitions from a number of different sources, they came to a general consensus that “usability” can be defined by how well a given interface is able to accommodate for diversity in its user base, whereas “accessibility” is a subset of usability that is categorized by specific accommodations for users with disabilities (Billi, et al.).

Billi, et al. also cite Petrie, who further classifies three common types of problems in user experience: *pure accessibility*, which affects those with performance limitations, *pure usability*, which affects those without performance limitations, and *universal usability*, which affects everyone (Petrie, H. & Kheir, O., as cited by Billi, et al.). With these observations in mind, it becomes evident that disabilities are a key consideration to factor in early on in the development process, as the sheer variety of potential performance limitations require a platform that is able to accommodate for unforeseen restrictions as they arise in testing.

**3.2. User Interface (UI) elements and organization**

Most interfaces tend to rely on visuals as their primary method of communicating with the user, as tactile or auditory feedback can require specific hardware that is less accessible compared to the ubiquitous digital display. Visual information can come in the form of text, images, videos, and interactive UI elements—all of which must share the screen without creating enough visual clutter to overwhelm the user. The way these visuals are organized depend heavily on the input method, particularly in regards to target size. When a user attempts to interact with a UI element, the size and location of that element are likely to impact how effectively the user can tap or click on that element without missing. Tapping and dragging UI elements with the finger, for example, tended to allow for faster speeds than with mouse or stylus, at the cost of accuracy (Cockburn, et al). This can be of particular concern for older people, as impairments to vision or motor function can impact how effectively one can hit their target (Díaz-Bossini & Moreno).

Díaz-Bossini, & Moreno also note the importance of color choice in UI elements, suggesting that it should only be used to *supplement* other methods of conveying information. When “required material is shown in red,” for example, they have observed that colorblind users often had difficulty distinguishing these red elements from other UI elements. To this end, they also suggested that blue and green tones should be avoided, as these colors tend to be most typically confused with others in most forms of color blindness (Díaz-Bossini & Moreno).

These elements should all factor into what the User Agents Accessibility Guidelines (UAAG) define as “operating environment conventions,” Billi, et al. argue. In essence, these guidelines specify that developers should observe the most common elements shared across various interfaces and consider how they might affect user expectations. For example, most users would expect a desktop site to feature some sort of navigation menu spanning the top or left edge of the screen. Developers need to be careful when diverting from convention, even if it has the potential for enhanced navigability, as doing so could create confusion for users who have come to develop certain expectations for how an interface should be interacted with.

**3.3. Assistive devices and accessibility features**

Assistive devices can be defined as any “item or piece of equipment that enables individuals with disabilities to enjoy full inclusion and integration in society” (Hakobyan et al.). In digital interfaces, assistive devices can come in a variety of forms, ranging from screen readers to remapped layouts customized for the user’s needs. Assistive technologies can exist on multiple planes, ranging from native features on the interface itself to external programs that further alter existing interfaces.

Díaz-Bossini and Moreno cite Google’s list of best practices for Android development as an example of accessibility-focused heuristics, which includes support for talkback & Explore by Touch, two assistive technologies provided by default in all Android devices. They also note the importance of providing descriptive text for all UI controls so that assistive technologies can work properly, which exist as “tags” that otherwise wouldn’t be seen by the average user (Díaz-Bossini & Moreno). This proved invaluable for those with visual impairments, as it significantly helped them navigate many “standard” sites that were not inherently designed to accommodate for visual impairments (Hakobyan et al). Similarly, Díaz-Bossini and Moreno observed that senior test groups benefited from applications that altered their phone’s interface to feature larger targets, simpler text, and minimized choices for the user (Díaz-Bossini and Moreno).

**3.4. Customizability**

One significant heuristic given by Billi, et al. is the allowance for personalization by providing layout options that the user can configure in accordance with contextual needs (Billi, et al). A good example of this would be the ability to change language, in order to make the interface more accessible for non-English speakers. Having features for customization allows all users, regardless of disability, to tailor their experience to better suit their specific needs.

**3.5. Responsive web design**

Responsive web design is somewhat of a compromise between mobile and desktop, as it runs on the principle of having a single interface that is able to adapt, or “respond” to various devices’ requirements. Most modern services feature responsive web design on the browser versions of their sites, utilizing UI elements that are shifted and resized depending on the dimensions of the display window. This “adapt**ive**” approach, as Zimmerman, et al. describe it, provides an advantage over “adapt**able**” sites that require manual activation of configuration settings. They argue that this approach does have some downsides however, including limited customizability and a general underappreciation by developers that leads to most implementations being rather lackluster (Zimmerman, et al). While responsive web design has the potential to incorporate a wealth of accessibility features, most companies seem to view it simply as a means of making their sites navigable on a wider range of devices.

**4. Grading scale for analyzing *YouTube* and *Spotify***

Through the observations I made in my literary analysis, I have settled on four core categories by which I will assess the desktop and mobile interfaces of *YouTube* and *Spotify*:

1. **Interactive element design**

In this category, I will assess the visual design of each interface’s interactive elements for how well they condone navigability. This can include the general layout of elements, the use of color, and the size or location of targets. I will also factor in how well these elements accommodate accessibility concerns, either through element design or the inclusion of assistive features. One final element to consider in this category is general aesthetics, or in other words the overall cohesiveness of the presentation of information. Sites should strive to be aesthetically pleasing, but never to such an extent that it compromises usability.

1. **Assistive device support**

For the purpose of this analysis, I will be checking for support of common screen readers in particular. I will also be discussing support for non-conventional input methods that are intended to aid users with disabilities, such as trackballs and joysticks.

1. **Customizability**

In this case, the term “customizability” refers to any configurable options that are implicitly designed to allow the user to alter the interface in order to better suit their needs. Features like these can include “dark mode” options, the ability to change language, and subtitles on videos.

1. **Responsive web design**

Responsive web design is a peculiar case, as it describes the use of a single interface between both mobile and desktop platforms. While both *YouTube* and *Spotify* offer dedicated applications that are meant for specific platforms and display resolutions, the browser versions of these sites feature responsive web design that allows it to switch between desktop and mobile layouts on the fly. As a result, I will be grading these interfaces on how effectively they are able to balance both device types by assigning two separate scores for each individual interface. For example, if a given interface offers a good user experience on the desktop layout but suffers after shrinking the screen size, it will get a higher score in the “desktop” category than in the “mobile” category, despite both categories describing the same interface.

Using these categories defined above, I have constructed a grading rubrik to help visualize the ratings across each interface type and category (see table 1).

**Table 1**Template grading rubrik for assessing desktop and mobile interfaces

| Scale: 1-3 | **YouTube Desktop** | **YouTube**  **Mobile** | **Spotify**  **Desktop** | **Spotify**  **Mobile** |
| --- | --- | --- | --- | --- |
| Interactive element design |  |  |  |  |
| Assistive device support |  |  |  |  |
| Customizability |  |  |  |  |
| Responsive web design |  |  |  |  |
| **Cumulative Score** | **/12** | **/12** | **/12** | **/12** |

**4.1 Analysis of *Spotify* and *Youtube***

Before conducting my analysis for each interface, I first had to identify which qualities would warrant a score of “1,” as opposed to a “2” or “3.” I have decided that a “3,” in this case, indicates that there was little to no perceived difficulty or potential for confusion in a given category. A “2” indicates general competency in that category, describing an interface that is able to be interacted with as intended (at least most of the time), even if there are some shortcomings that could be improved upon. Finally, areas that are in immediate need of improvement will receive a score of “1.” This score, as opposed to a “2,” implies that the interface has largely failed to address the concerns outlined in a given category, and requires improvement before it can be considered adequate enough to be effective. Here, I have compiled the final scores into the grading rubrik outlined previously (see table 2). In sections 4.1.1 through 4.1.4, I will elaborate further on how I settled on these scores through my observations in each category.

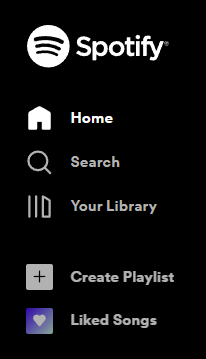
**Table 2**

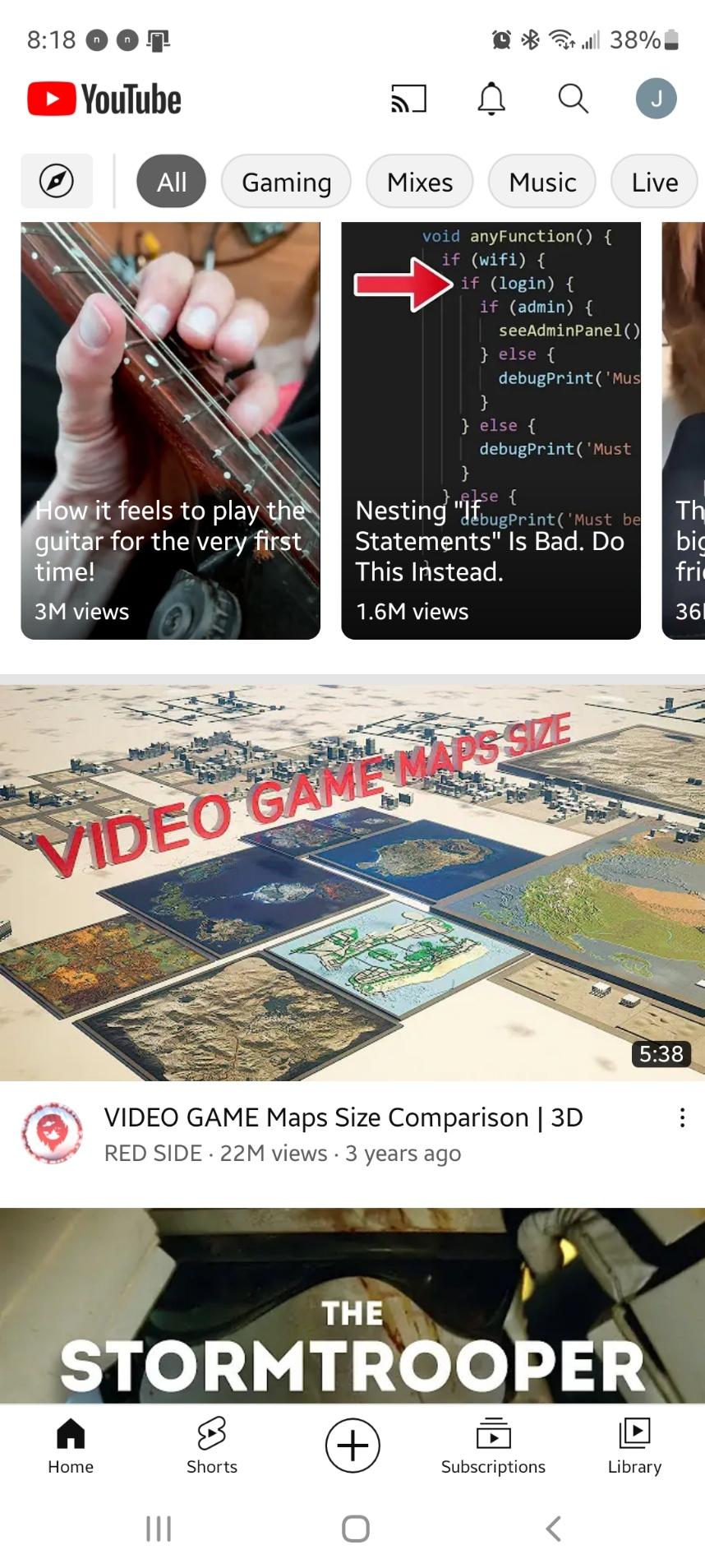
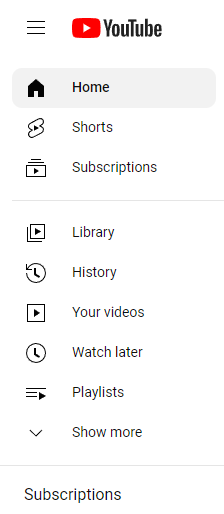
Final scores for accessibility and usability in interfaces for *YouTube* and *Spotify*

| Scale: 1-3 | **YouTube Desktop** | **YouTube**  **Mobile** | **Spotify**  **Desktop** | **Spotify**  **Mobile** |
| --- | --- | --- | --- | --- |
| Interactive element design | **3** | **2** | **3** | **3** |
| Assistive device support | **3** | **3** | **3** | **3** |
| Customizability | **2** | **2** | **1** | **1** |
| Responsive web design | **3** | **2** | **3** | **3** |
| **Cumulative Score** | **11/12** | **9/12** | **10/12** | **10/12** |

**4.1.1. Analysis of interactive element design**

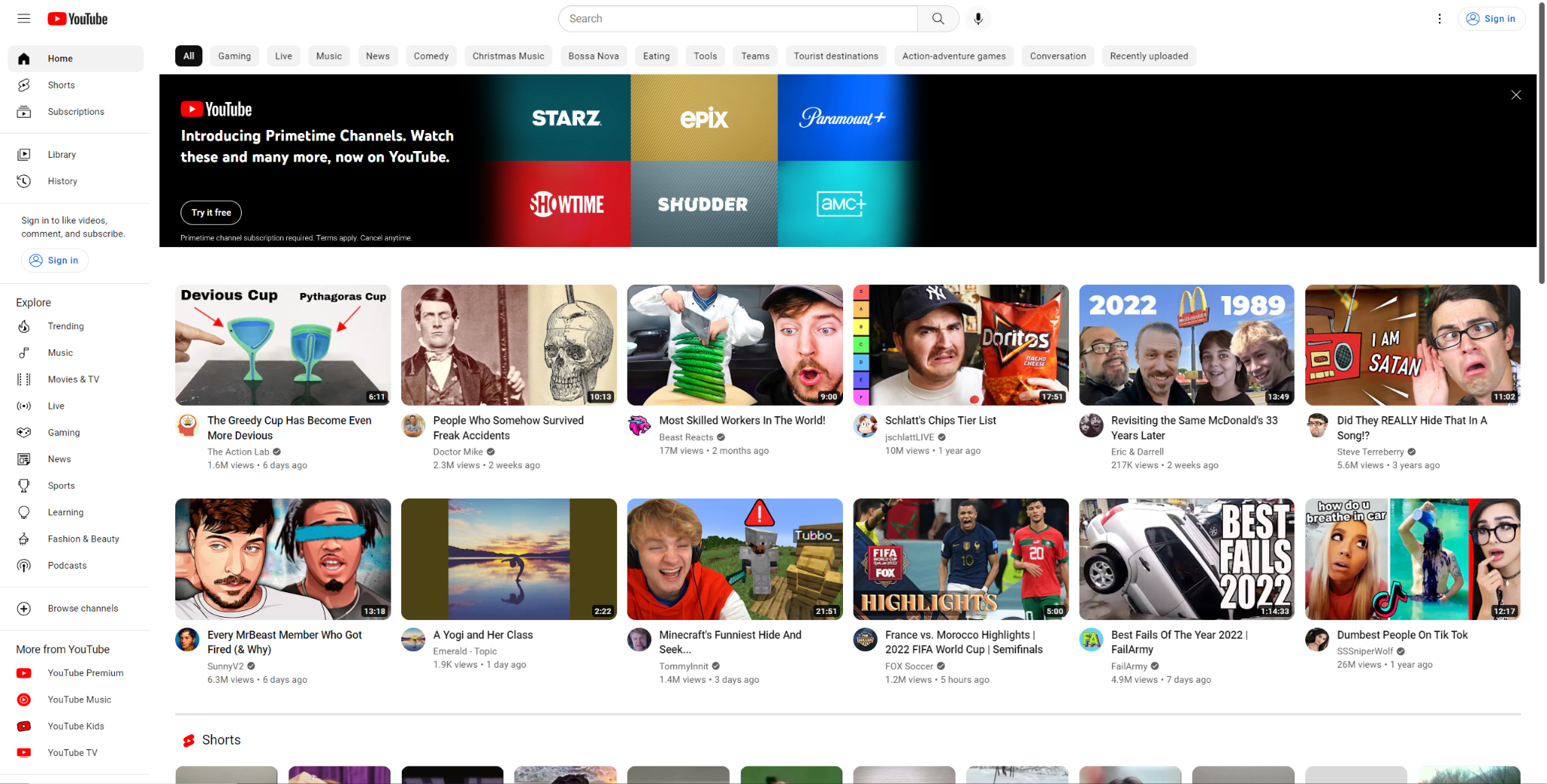
In general, *Spotify* and *YouTube* appear to follow the same basic layout conventions for both desktop and mobile offerings of their service. Both are rather conservative with their use of colors, straying from black or white only to place additional emphasis on interactive elements of particular importance. On desktop, both sites feature a primary navigation bar spanning the left side of the screen. *YouTube* allows users to collapse this menu using a “hamburger” icon in the top-left corner regardless of screen size, however *Spotify* offers no such option—the menu permanently occupies the left side here, and there is no way to minimize it. Both menus utilize a “library” system, which directs users to a list of personalized features like custom playlists and creators that the user is following/subscribed to. This is a primary means of navigation for both sites, as the “library” offers a means of accessing nearly every commonly-used feature on their respective sites. On their mobile applications, this same navigation menu is simplified and brought down to span the bottom of the screen rather than the left edge (see Figure 1).

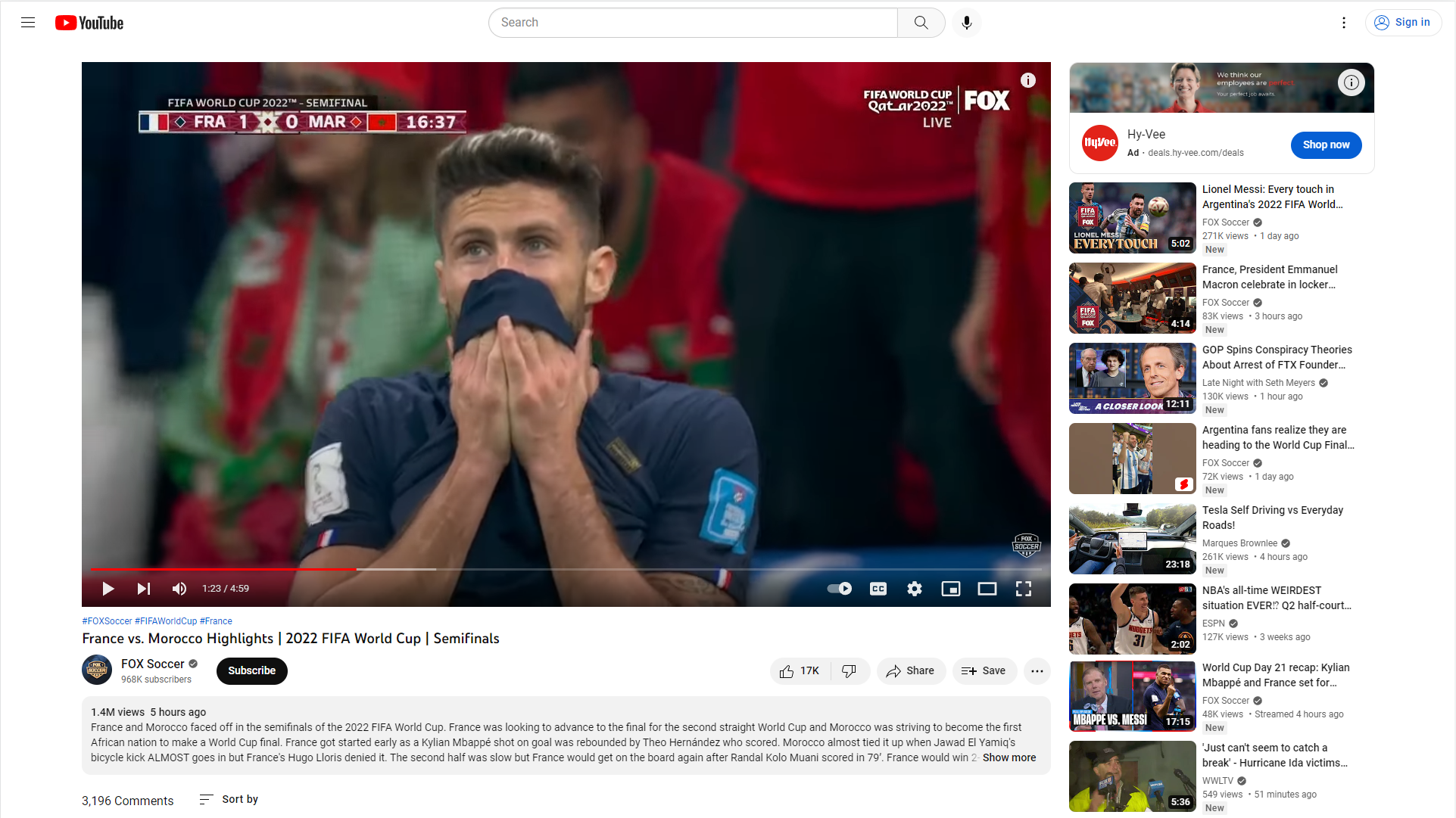




**Fig. 1.** Comparison of navigation menu on desktop (left) and mobile (right) interfaces for *Spotify* and *YouTube*

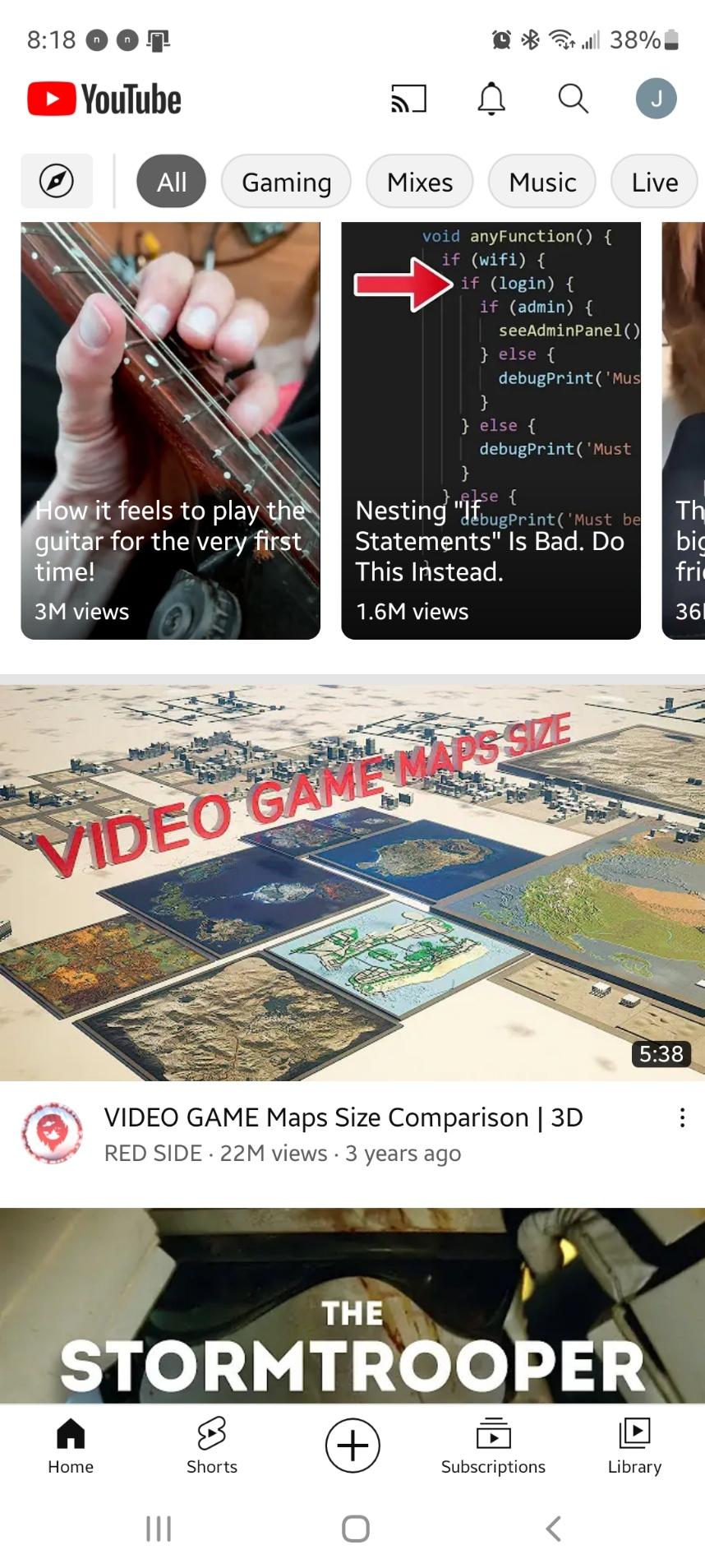
On both iterations of YouTube’s interface, the rest of the page features a curated list of recommended videos displayed in a grid of thumbnails accompanied by the video’s title, uploader’s username, view count, and upload date (See Figure 2). While the core design of these elements remains consistent between desktop and mobile, the size of the display dictates how many can be shown on screen at once. The larger screen and superior accuracy of the mouse allows the desktop interface to display a greater quantity of these thumbnails, which are displayed neatly in a grid on the home page (see Figure 2). When watching a video, related videos are suggested through smaller thumbnails displayed in a vertical list along the right side of the video player, with additional details about the video listed underneath.





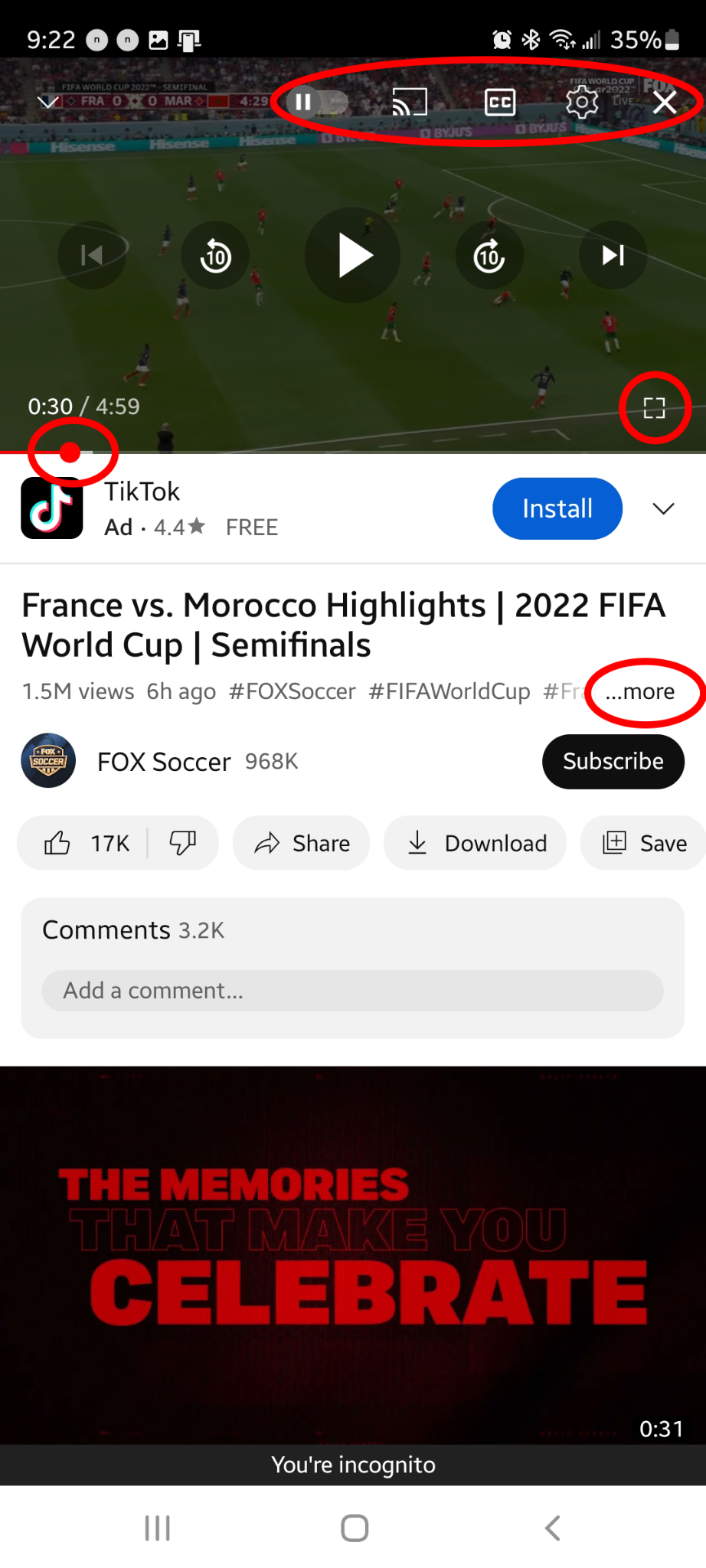
**Fig. 2.** Screenshot of YouTube desktop home page (top) and video player (bottom)

On the mobile application, each individual thumbnail takes up a significantly larger portion of the screen; in fact, a single thumbnail spans the entire width of the page. This consequently allows for only two or three thumbnails to be visible at a time, however it does allow for massive touch targets that make it very easy for users to select videos that pique their interest (see Figure 2.1). It is worth noting, however, that most tablets will adopt the grid layout from desktop. The display of these thumbnails remain unchanged when watching a video on the mobile application, however the user does need to scroll down past the description in order to browse through them.



**Fig. 2.1.** Screenshot of YouTube mobile application homepage

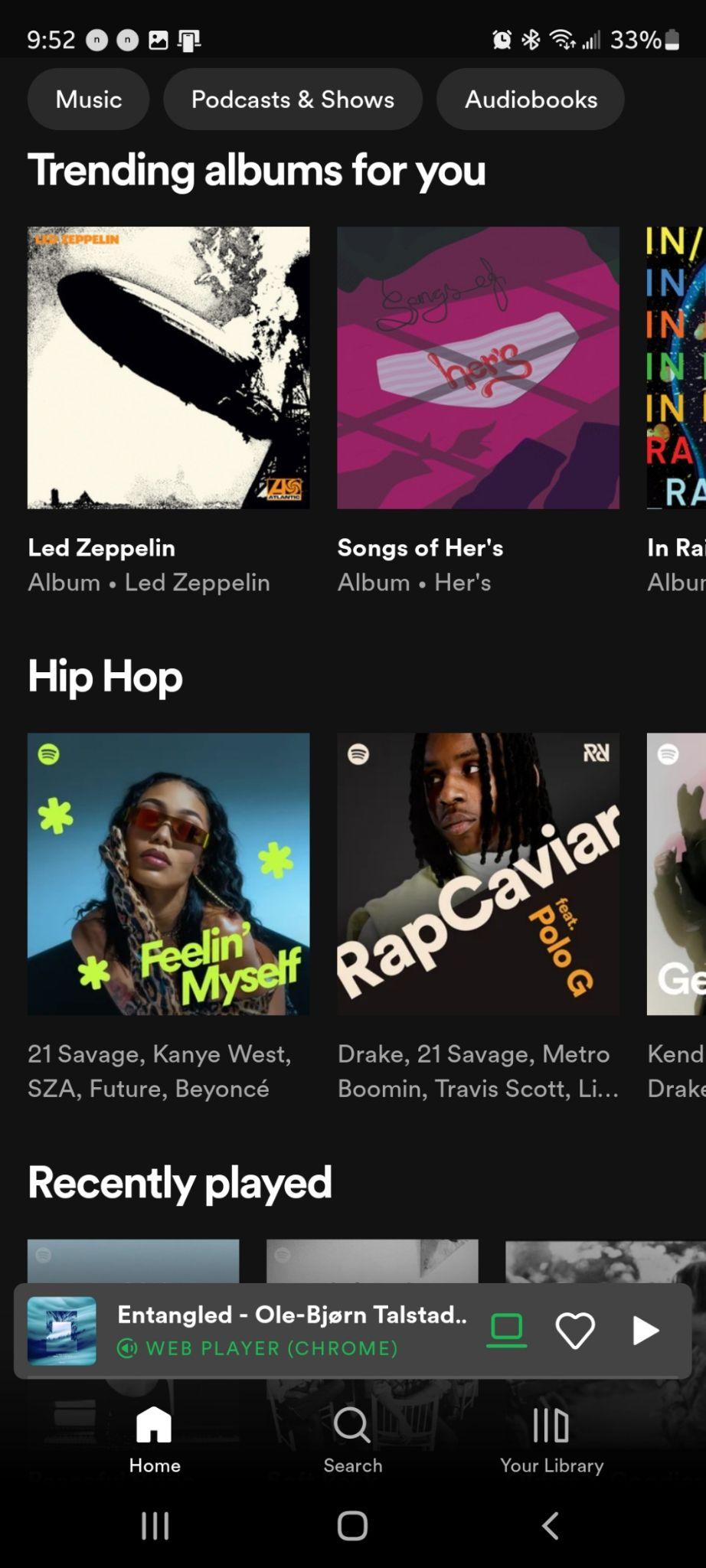
One issue exclusive to *YouTube’s* mobile application is that many of the tap targets, namely in the video player portion, can be difficult to access at times. These targets are very small, even by mobile standards, and lack any sort of background or border to indicate where the target boundaries begin or end. Many of these elements are also placed very closely together on the screen, altogether resulting in an increased likelihood of misinputs (see Figure 2.2).



**Fig. 2.2.** Elements circled in red are small and cramped, increasing misinputs

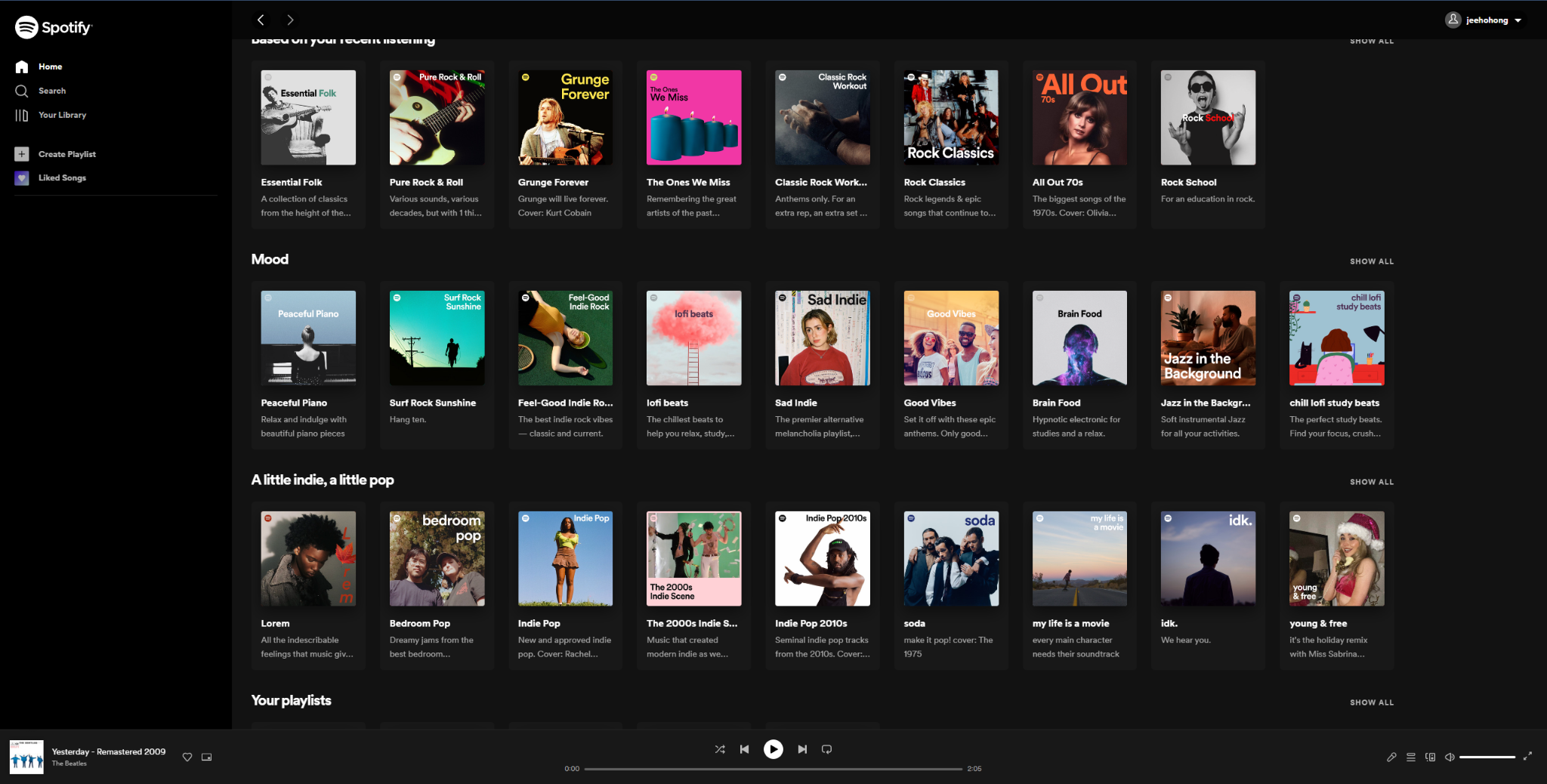
Despite these minor issues, *YouTube’s* interface design becomes quite intuitive with minimal use on either platform. The target issues on mobile can lead to some frustration, but are easy enough to access to function as intended most of the time. For the average user, this only appears to become an issue when tapping haphazardly, or accidentally when the user touches the screen when they didn’t intend to. For these reasons, I gave the desktop interface a rating of “3” in this category, whereas the aforementioned issues with the mobile application knocked its rating down to a “2.”

*Spotify* shares a lot in common with *YouTube* in its layout, utilizing a very similar system of “thumbnails” to suggest curated playlists and albums. Songs are displayed in the same way, most typically with a thumbnail of the album cover or artist accompanied by the title and artist’s name. Unlike *YouTube*, these elements are much smaller on average, and there is a distinct difference in the way playlists are represented as opposed to individual songs. When a user clicks on a thumbnail to view a playlist, they are then greeted with a simple list of all the songs in said playlist (see Figure 3).



**Fig. 3.** *Spotify* mobile application home screen (left) and playlist/album view (right)

When a song is playing, This “player” screen can be put into “fullscreen mode” on both versions, blowing up the song’s associated image to occupy the majority of the screen with media controls panning the bottom. When minimized, however, the mobile version sacrifices all media controls except pause/play in order to fit within the narrower screen width (see Figure 3). These controls are much more prominent on the desktop version, and are permanently affixed to the bottom of the screen (see Figure 3.1).



**Fig. 3.1.** Screenshot of *Spotify* desktop interface

Because its layout is so similar to that of *YouTube’s*, *Spotify’s* interface is equally as intuitive to navigate. Tap targets on the mobile application, unlike *YouTube*’s equivalent, are much more clearly defined and optimally positioned. As a result, both desktop and mobile iterations of *Spotify’s* interface received a “3” in this category.

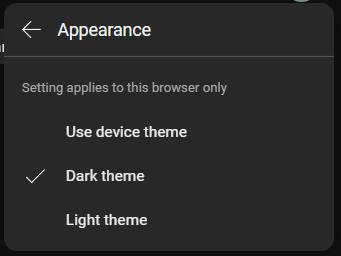
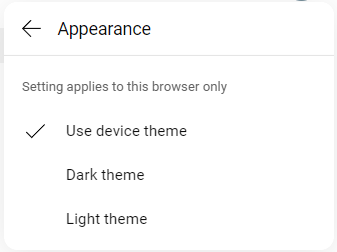
**4.1.2. Analysis of assistive device support**

One of the most common assistive technologies in use today are screen readers, which assist visually impaired users by reading aloud text that is on the screen. Today, it has become standard for most commonly used devices to natively provide a screen reader in their operating systems. All *Apple* devices, for example, offer an in-house developed screen reader called *Voiceover*; this includes iPhones, iPads, and Mac computers. *Android* devices offer a similar alternative, called *Talkback*. While *Windows* computers offer no screen reader natively, they do support the ability to install third-party assistive software like *NVDA, JAWS,* and *Speechify*—all popular screen readers which are available for a multitude of devices. Many of these programs also allow the user to connect to assistive devices like braille displays, further enhancing the experience in specific cases. Both *Spotify* and *YouTube* support screen readers, with their desktop and mobile versions being compatible with the most common programs on their respective devices.

While I had initially intended to test these interfaces with assistive input methods, I was unable to procure any for this analysis. Reports on compatibility are spotty at best, as assistive input devices are such a niche industry that it can be difficult for a developer to support *all* potential input methods. This can often be worked around, however, as many assistive devices will emulate the inputs of more common devices like keyboards and computer mouses (The Arc of the Bay). Naturally, *Spotify* and *Youtube* support these types of input methods, as they already support a variety of input methods like keyboards, mouses, touch screens, and so on. While further research may reveal more shortcomings in accessibility features, I have yet to observe any obvious faults in the interfaces covered in this analysis. As such, I have given each interface a tentative rating of “3.”

**4.1.3. Analysis of customizability**

Customizability appears to be the weakest link between the two services, as options to alter the interface are sparse to nonexistent in any of the platforms I have observed. While browsing the “settings” tab of each respective interface, the first option I noticed was the ability to change the language of the text. This is obviously quite important for those whose first language is not English, and the option is thankfully provided by both *YouTube* and *Spotify*. *YouTube* also provides an option to toggle to a “dark” theme, which inverts the interface’s color scheme to feature white text over a dark background (see Figure 4). This can be particularly useful in darker environments, where a bright backlit screen could cause eye strain. *Spotify* supports this design by default, however it does not provide any option for changing to a lighter color scheme.

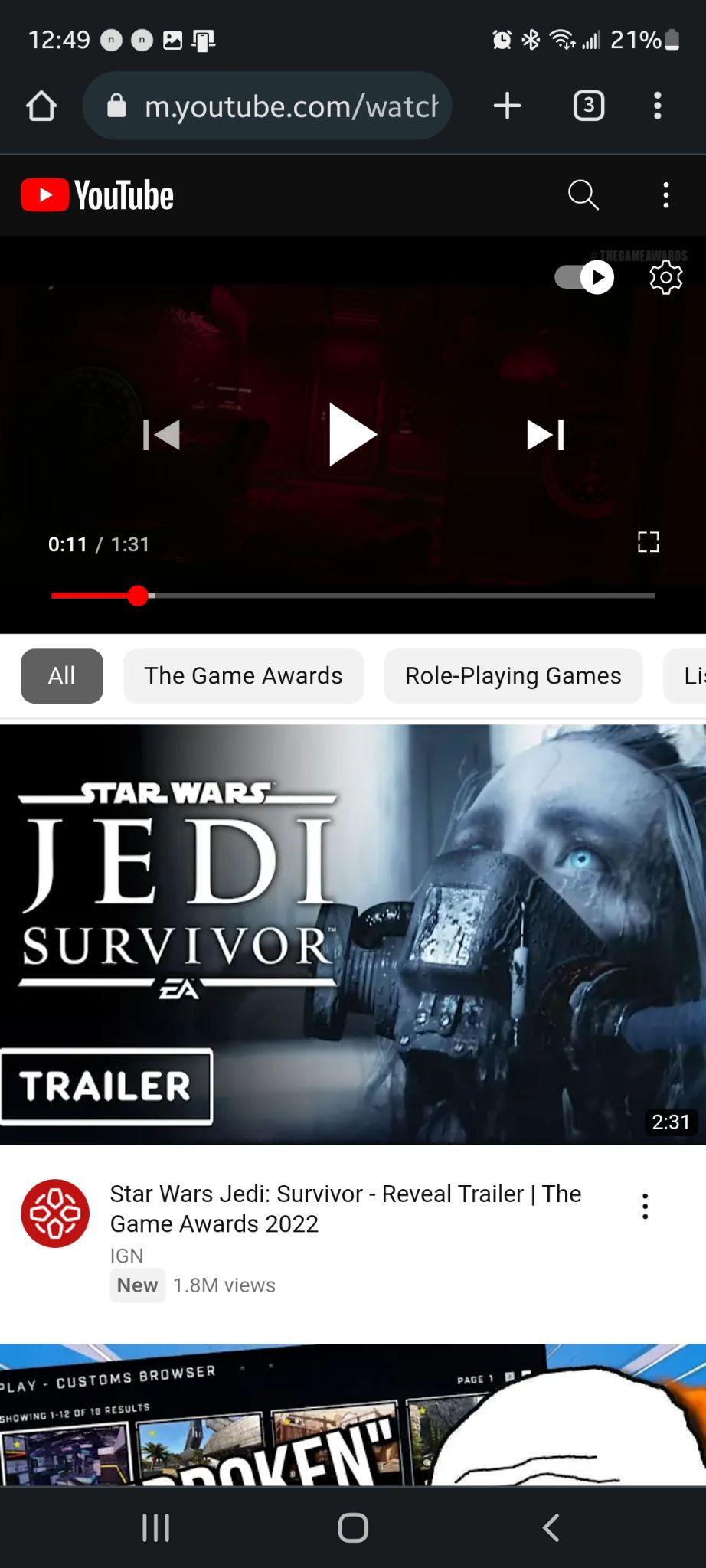
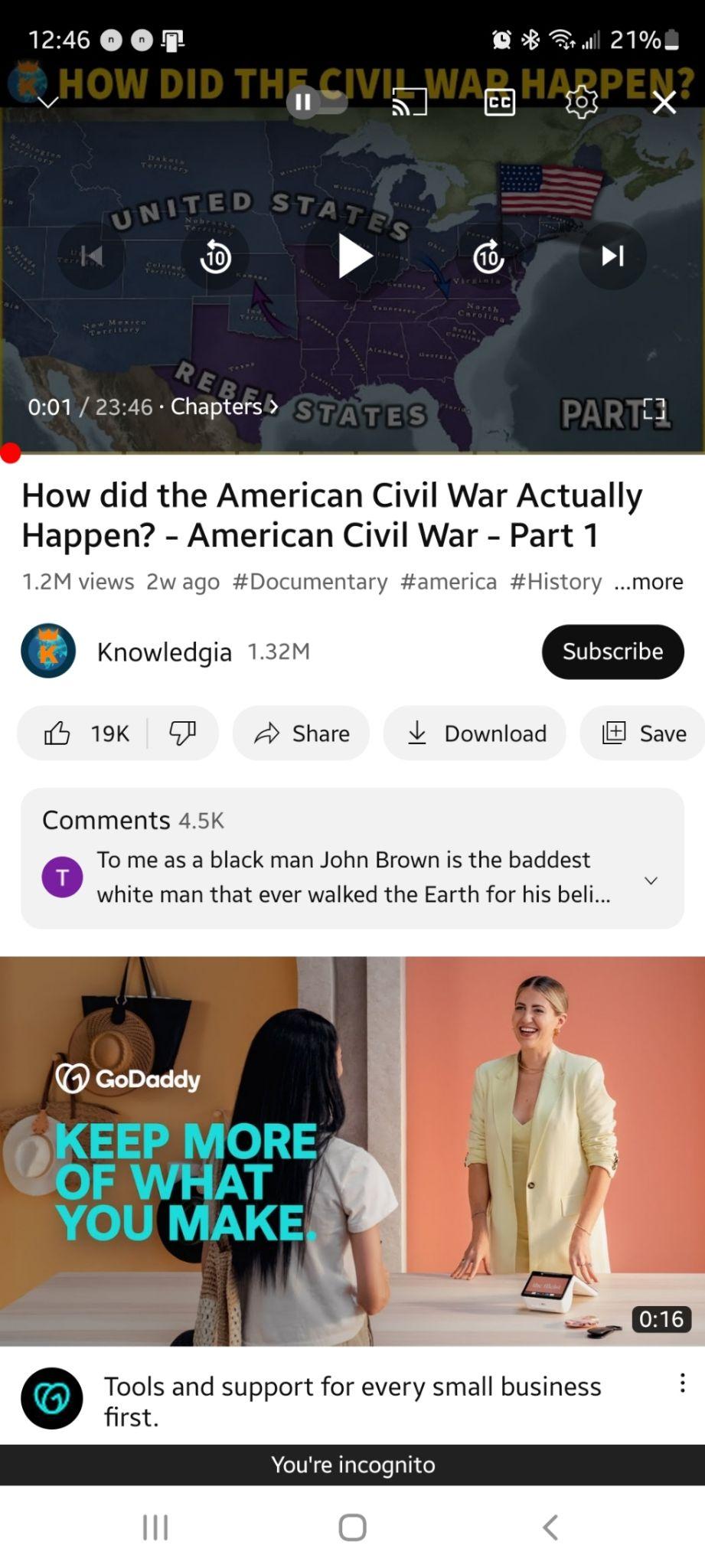


**Fig. 4.** “Light” and “Dark” theme settings on *YouTube* desktop site

*YouTube’s* customization options are also much more intuitive to access, as they are accompanied by graphic icons to supplement the text. Although both are lacking, it is evident that *YouTube* has put a bit more thought into accessibility options than *Spotify* has. Both mobile and desktop versions of *YouTube* received a “2” for their flawed but evident attempt at customizability, whereas *Spotify* received a “1” for both of its interfaces for lacking all but the very basics.

**4.1.4. Analysis of responsive web design**

When viewed through a web browser, both *Spotify’s* and *YouTube*’s websites feature responsive web design. For the most part, this interface replicates either the desktop or mobile applications depending on the size of the screen. The navigation bars on the left function in exactly the same way, and minimize to the bottom once the width of the screen dips below a certain threshold. Because these interfaces replicate the dedicated applications so closely, the differences between them are minute. *Spotify’s* browser site functions almost identically to both the desktop and mobile applications, earning a “3” for both in this category. When the desktop version of *YouTube* is brought down to a mobile screen size, however, it fails to provide some minor features that exist on the application. These include closed captioning, screencasting, and the ability to swipe down on the player to return to the home screen (see Figure 5). This version also suffers from all the aforementioned shortcomings as the mobile application, however these flaws can be remedied if simply viewed on a larger screen. As a result, the desktop version of *YouTube’s* responsive interface earns a “3,” while its ungraceful transition to mobile brings its score down to a “2.”



**Fig. 5.** *YouTube* playback options on browser (left) and mobile application (right)

**5. Concluding assessment**

After comparing the final scores for each category, it became apparent that each of the two services had their own unique weak points. Few of *Spotify*’s desktop features are sacrificed in its translation to mobile, however both versions are still lacking somewhat in customizability and accessibility features. While these interfaces are very intuitively designed and relatively accommodating to common disabilities, *Spotify* provides no additional features or modifiers that would allow the user to further tailor the interface for their specific needs. One could argue that this lack of features is moreso a result of the simplicity of streaming music, as an audio-only medium is inherently more accessible than one that requires *both* audio and video at the same time. Regardless, simple options like the ability to change to a “light” mode or enable audio-based feedback to inputs could go a long way in making the interface easier to navigate for certain users, even if the user experience is already considered acceptable for most. *YouTube* fares slightly better in providing customizability and accessibility options, the ease of navigability in its interfaces are about on par with that of *Spotify*. However, compared to *Spotify,* there is a greater sense of disconnect between *YouTube’s* desktop and mobile iterations.

Even if a desktop or mobile interface works well for a certain platform independently, one must also consider how well the experiences of using these two platforms translate from one to another. If the two versions are too different, the expectations established by one may lead to confusion when attempting to navigate the other. As such, both desktop and mobile versions of the same site should strive to structure information similarly so that the experiences of using either interface are more likely to overlap. *YouTube’s* mobile application may seem comparable to its desktop version on the surface, however there are some oversights that significantly impact the actual experience of navigating the interface. As discussed in Section 4.1.1., for example, *YouTube's* mobile interface suffers from poor target design and a significant shift in the way thumbnails are displayed. The desktop version also benefits from the mouse’s more granular input methods, such as the option to preview video contents by “hovering” the mouse cursor over a thumbnail. Features like these were largely removed or automated in the mobile app, resulting in a lesser sense of control for the user. The differences between *Spotify’s* desktop and mobile versions are far less pronounced, leading to a more seamless transition in user experience.

**6. Recommended heuristics for joint mobile/desktop development**

As the primary deliverable for this paper, I have composed a list of the development heuristics that seemed to have the largest impact on user experience. My analysis of *Spotify*  and *YouTube* has shown that these heuristics significantly benefited the experience when adhered to, and harmed the experience when omitted. Through the course of my research, I observed the following heuristics to be most important when developing desktop and mobile interfaces for the same service in tandem.

1. **Start development on mobile before desktop**

Screen size has shown to be the primary factor that determines how much information can be shown on screen, so adapting an existing desktop interface to a mobile platform would require compromising on certain features or layout decisions. Input methods also play a role here, as a mouse and keyboard allows for more granular control methods and far greater precision than interacting with a touchscreen by hand. This, combined with the larger screen size, grants desktop interfaces much more leeway in how UI elements can be designed and oriented. As such, it is generally recommended that UI designers start development on the mobile iteration first, as a later expansion to desktop would require fewer sacrifices during the adaptation process (Morales).

1. **Emphasizes importance of user testing**

There are a wide variety of potential issues that can come up during the development process, and these issues can often be extremely hard to predict. One way of combating this is through extensive user testing with multiple focus groups. Because the consideration for disabilities plays such a critical role in the early stages of development, testing should begin early on with focus groups that reflect these potential impairments. The *YouTube* mobile app could definitely benefit from greater testing, for example, as it could aid the developers in picking up on the flawed design of tap targets discussed earlier.

1. **Support common assistive technologies**

Support for assistive devices has become somewhat of a standard across mobile and desktop UI designs, and for good reason. Assistive devices can significantly enhance the user experience for many users, and is often a necessity for users who would not be able to use the interface otherwise. Popular screen readers like *Voiceover*, *Talkback,* and *NVDA* should be supported, along with other assistive technologies like magnifiers and braille displays.

1. **Provide customizability options for the user**

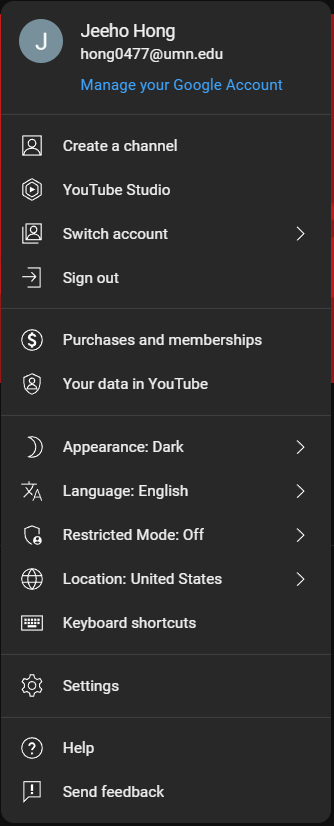
There is no “one size fits all” option for user interface design, as each individual in a user base can interpret and interact with interfaces in slightly different ways. This can be because of a disability, as mentioned above, or because of previous experiences with an interface that follows completely different design conventions. By providing options that would allow users to customize their experience, developers can often improve the experience of using their interface significantly with relatively minimal effort.

1. **Know when to follow conventions**

Through years of experimentation, UI designers seem to have largely converged on a select few techniques that are likely to yield good results. While these techniques are by no means gospel, there is a reason why so many websites have a central navigation bar spanning the top or left of the screen—people have simply come to *expect* this type of layout from their experiences with other similar interfaces. Diverting from these conventions *could* yield positive results, but developers must realize that straying too far from the norm can alienate users that have pre-set expectations. Adherence to convention is often a balancing act, as sticking to standardized techniques can provide reliable results, whereas experimentation with novel techniques has the potential to yield significant advantages over competing platforms—if done *right*.

1. **Utilize redundancy in conveying information**

When a user interacts with a UI element, it is standard practice to provide some sort of feedback to let the user know that the device has successfully registered their input. This is critical in giving the user a sense of control, it enforces the idea that every input will lead to a tangible effect. In *Spotify* and *YouTube*, this is typically done visually by briefly altering the color tone of the UI element when a user interacts with it. Other common methods are to enlarge the size of the element, or the use of audio cues like tapping or beeping. These various techniques are often combined, as any one effect could go unnoticed by the user. By doubling up on these techniques, developers can increase the likelihood that their feedback will actually be recognized by these features. A good example of redundancy can be seen in *YouTube’s* “Settings” menu, as each option is accompanied by a visual graphic that reflects what kind of content it leads to (see Figure 6).



**Fig. 6.** “Settings” menu on the *YouTube* desktop site

1. **Aesthetics should never take precedence over functionality**

Any interface may benefit from an aesthetically pleasing design, but developers often make the mistake of hindering usability in favor of visual flair. Once again, the *YouTube* mobile application’s player controls serve as a good example of what *not* to do (see Figure 2.2). The smaller, more minimalistic icons may *look* nice, but in practice they have shown to be less effective than the standard “button” style icons on an interface like *Spotify’s.* As for the overall layout, an interface’s visual presentation should be simple, as to avoid overwhelming the user with information, and colors outside of black, white, and gray should be used conservatively. Developers should focus first on making sure their interface is accessible and comfortable to navigate, and *then* apply a pleasing visual theme that will not conflict with the functionality of the interface.

**Works Cited**

Billi, M. et al. (2010). A Unified Methodology for the Evaluation of Accessibility and Usability of Mobile Applications. *Universal Access in the Information Society*. 9, 337-356. <https://link.springer.com/article/10.1007/s10209-009-0180-1>

Caldwell, B. et al., editors. (2008). Web Content Accessibility Guidelines (WCAG) 2.0. *World Wide Web Consortium (W3C).* <https://www.w3.org/TR/WCAG20/>

Christie, J., & Soe, Y. (2022). Ideas for Spotify Customization: Enhancing the User’s Experience. *Communications in Computer and Information Science.* 1580. <https://link.springer.com/chapter/10.1007/978-3-031-06417-3_3#citeas>

Cockburn, A. et al. (2021). Understanding Performance in Touch Selections: Tap, drag and Radial Pointing Drag with Finger, Stylus and Mouse. *International Journal of Human-Computer Studies.* 70(3), 218-233. <https://www.sciencedirect.com/science/article/pii/S1071581911001546>

Díaz-Bossini, J., & Moreno, L. (2014). Accessibility to Mobile Interfaces for Older People. *Procedia Computer Science.* 27, 57-66. <https://www.sciencedirect.com/science/article/pii/S1877050914000106>

Hakobyan, L. et al. (2013). Mobile Assistive Technologies for the Visually Impaired. *Survey of Ophthalmology.* 58(6), 513-528. <https://www.sciencedirect.com/science/article/pii/S0039625712002512>

Lukoff, K. et al. (2021). How the Design of YouTube Influences User Sense of Agency. *Association for Computing Machinery. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems.* 368, 1-17. <https://dl.acm.org/doi/abs/10.1145/3411764.3445467>

Morales, Justin. (2021). Mobile First Design Strategy: The When, Why and How. *Adobe.* <https://xd.adobe.com/ideas/process/ui-design/what-is-mobile-first-design/>

The Arc of the Bay. Assistive Computer Devices & Software. *St. Andrew Bay Center, Inc.* <https://bayarc.org/assistive-computer-devices-software/>

Zimmerman, G. et al. (2018). Inclusive Responsiveness – Why Responsive Web Design Is Not Enough and What We Can Do About This. *Advances in Intelligent Systems and Computing.* 776. <https://doi.org/10.1007/978-3-319-94622-1_20>