Spotify Music Discovery: Scenario-Based Interpretation and Consolidation

Anisa Callis, Lauren Casale, Sidney Grabosky, Patrice Shih, and Damionie Vernon

HCIN 730 User-Centered Design Methods

Prof. Lawrence Roth

Rochester Institute of Technology

December 15, 2025

Table of Contents

Introduction

Music discovery is a core component of the Spotify listening experience. Yet, our contextual inquiry findings revealed pervasive challenges that prevent users from discovering and organizing music in ways that feel intuitive, personalized, and low-effort. Across interviews and consolidated models, participants expressed frustration with homepage clutter, opaque recommendation algorithms, and the cognitive burden of managing large “Liked Songs” playlists. These issues disrupt users’ goals of simple discovery, meaningful curation, and trust in Spotify’s recommendations, particularly in everyday contexts such as commuting, working, or driving.

Building on insights from our wall walk and interpretation sessions, this report presents team solutions for redesigning Spotify’s music-discovery workflow. Using visioning, storyboarding, and high-level use cases, we reimagine how users interact with recommendations, homepage content, and saved music. Our proposed design emphasizes flexibility, transparency, and user control, allowing listeners to shape their discovery experience without increasing cognitive load. The following sections describe our design ideas from the user’s perspective, articulate future improvements to Spotify’s discovery environment, and illustrate how this redesigned system supports users’ real-world listening behaviors through concrete scenarios and use cases.

Redesign Brainstorm Ideas

The first redesign focuses on some of the most prominent digital real estate in the application: the Spotify homepage. Reflecting numerous findings from our ethnographic studies, users find the homepage cluttered with irrelevant content and fail to surface content of interest. What users want to see on their homepage varies widely, reflecting the diversity of individual musical preferences. For the homepage to truly feel like home, it must be customizable to users’ needs, preferences, and workflows.

The current homepage lacks customization; it is narrowly tailored to the most popular music, effectively Spotify's version of the Billboard Top 40, creating a significant disconnect for users whose listening patterns differ. This also creates a stale user experience, as the most popular artists are stable over multi-year timeframes. Many users are uninterested in content such as podcasts or e-books, which consume a large amount of the homepage.

To resolve these issues, by personalizing, dynamizing, and making the homepage more relevant, users should be able to access trending content from similar users or niche-popular content within specific genres. Spotify could also draw inspiration from more traditional music lists that people already trust; numerous third-party lists and aggregators for every conceivable music genre could be incorporated. For example, users whose music tastes skew toward alternative or indie genres who are overlooked by the current homepage have numerous relevant publications available, such as Paste, Pitchfork, Stereogum, BrooklynVegan, and Consequence, as well as charts such as the "New Alternative 40." Other genres have these resources as well; there is no reason the homepage needs to cater only to the global top 40.

Another improvement would be to give users greater control over discovery itself. While we have not explicitly addressed this, a simple slider or mixer that allows you to adjust the balance between new music and familiar favorites could be beneficial for future implications. Some days involve exploration, while other days involve directly navigating to your preferred option. To this end, providing users with an easy way to switch between "adventurous" and "tired and true" would help playlists remain fresh without becoming stale or overwhelming.

Many users expressed dissatisfaction with the quality of algorithmic recommendations and the lack of a precise mechanism to address the issue. Algorithmic recommendations are integral to the Spotify user experience, and when they function poorly, the platform loses significant value. Users need a way to more directly correct a misaligned recommendation algorithm when this occurs, without the delays and uncertainty of attempting to address the problem with existing, highly decoupled methods such as liking and skipping songs. We

envision solutions that involve rapid iterations of generated sample playlists, with the ability to correct the algorithm via Tinder-style swiping gestures and optional feedback to help the algorithm better understand users' tastes.

Finally, our ethnographic studies and consolidation showed that, while the Liked playlist is a ubiquitous part of Spotify users' workflows, its use was often dysfunctional. The Liked playlist, being the only "first class" playlist offered in the user interface, became a "junk drawer" in which hundreds or thousands of songs accumulated without organization. The continually increasing size of this playlist subsequently rendered the task of organizing it a daunting and overwhelming undertaking. To resolve this issue, we envision separating "Liking" from "Saving" songs, making both actions "first class" in the user interface, and dynamically limiting the default 'Recently Saved' playlist so that it only shows the last month of activity, helping to prevent users from becoming overwhelmed when they go back to organize saved songs later.

Vision

In this section, we demonstrate environmental improvements and the user flow from the users' perspective using visual illustrations. Given that users prefer fewer features and have reported homepage clutter, we added functionality to organize the homepage to address this issue. Users can remove features they are not interested in and adjust the remaining ones. Because Spotify's current recommendation algorithm is causing user dissatisfaction, we developed a recommendation fine-tuning system. This feature enables users to assist with algorithmic adaptation by rating songs in a novel way, such as swiping left or right to indicate whether they like a song. In the interview, some users also mentioned frequent use of the "liked song" playlist, which may contain thousands of songs, and that they have difficulty navigating to the songs they want. Therefore, the new "Save For Later" list can temporarily store users' interested songs for later organization.

1. Homepage Customization

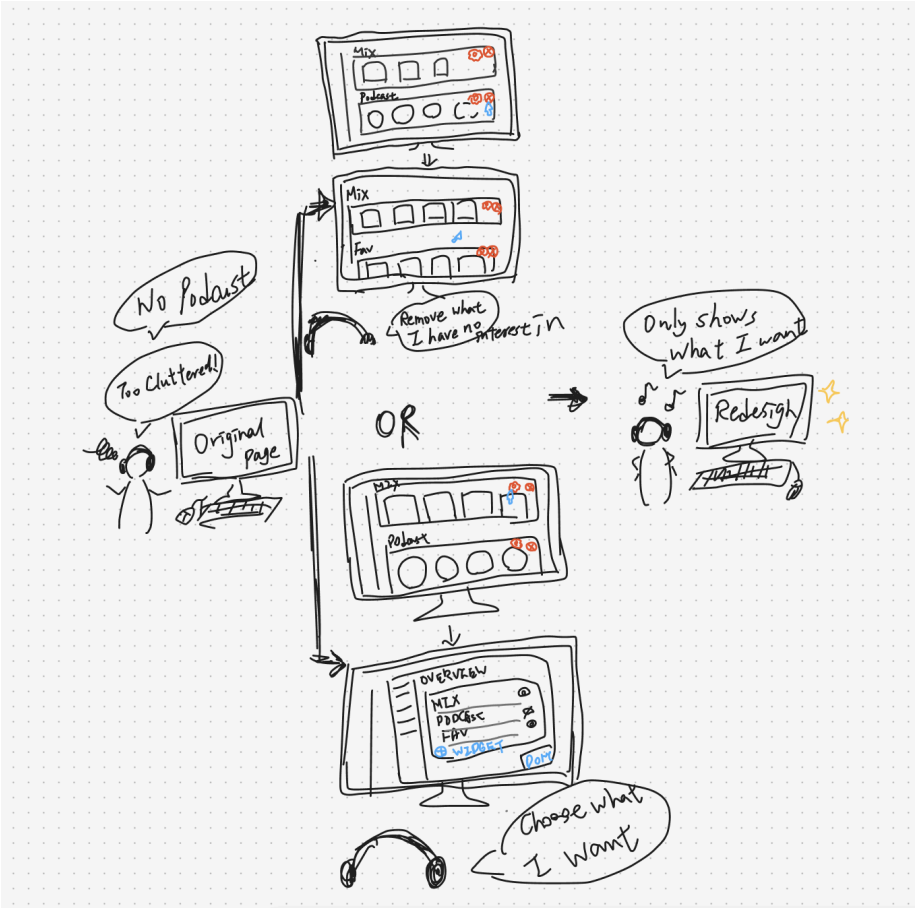


Figure 1.1

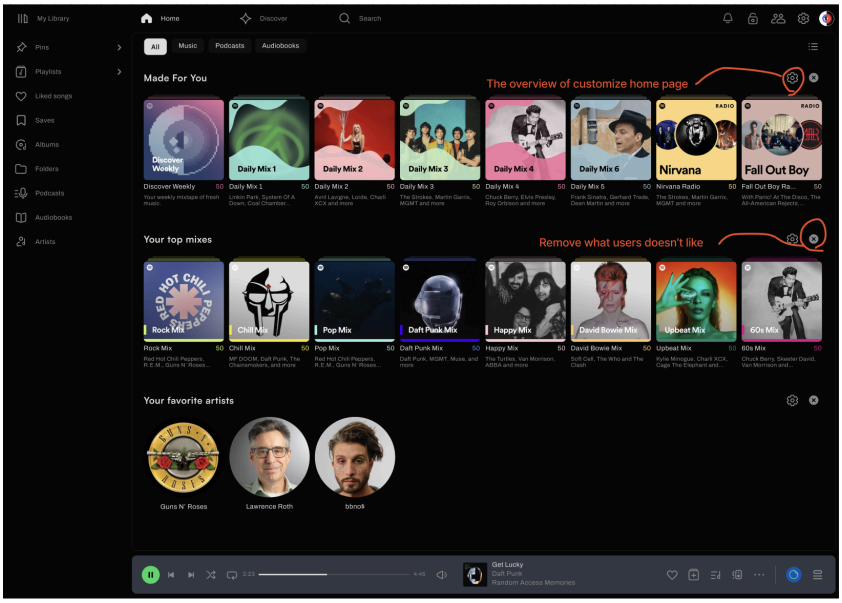


Figure 1.2

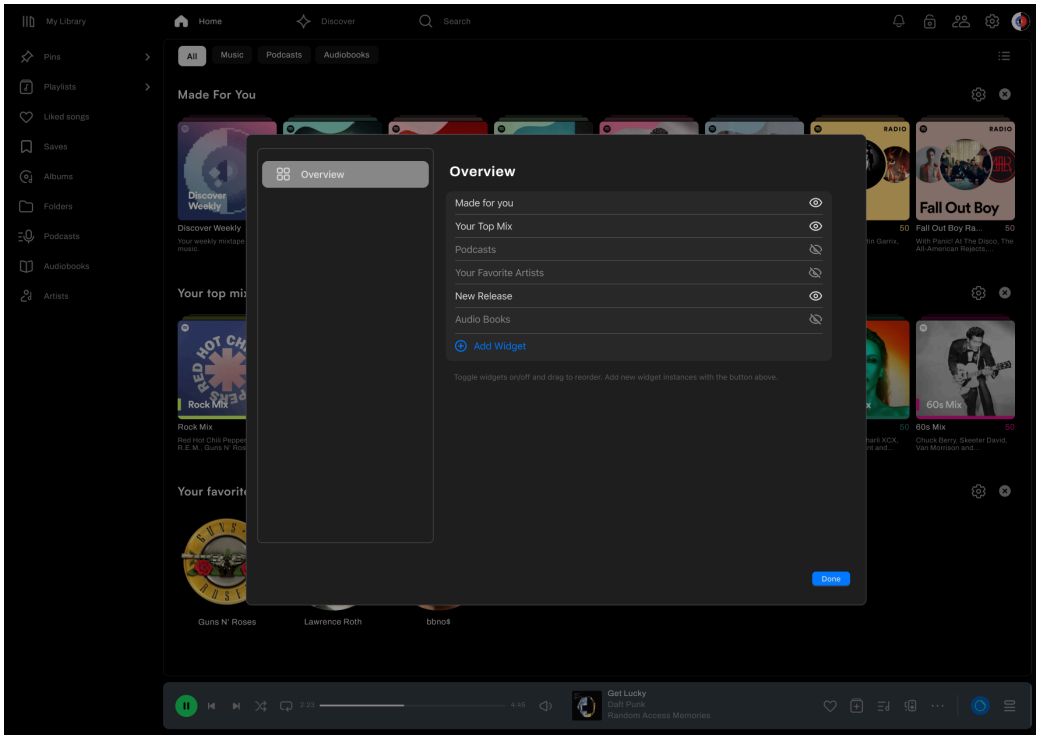


Figure 1.3

2. Recommendation Fine-Tuning

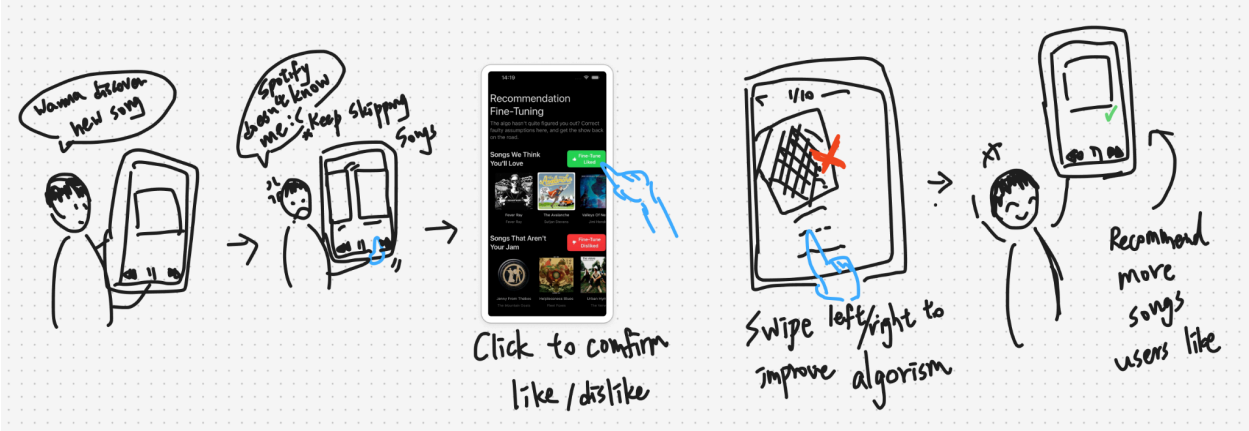


Figure 2.1

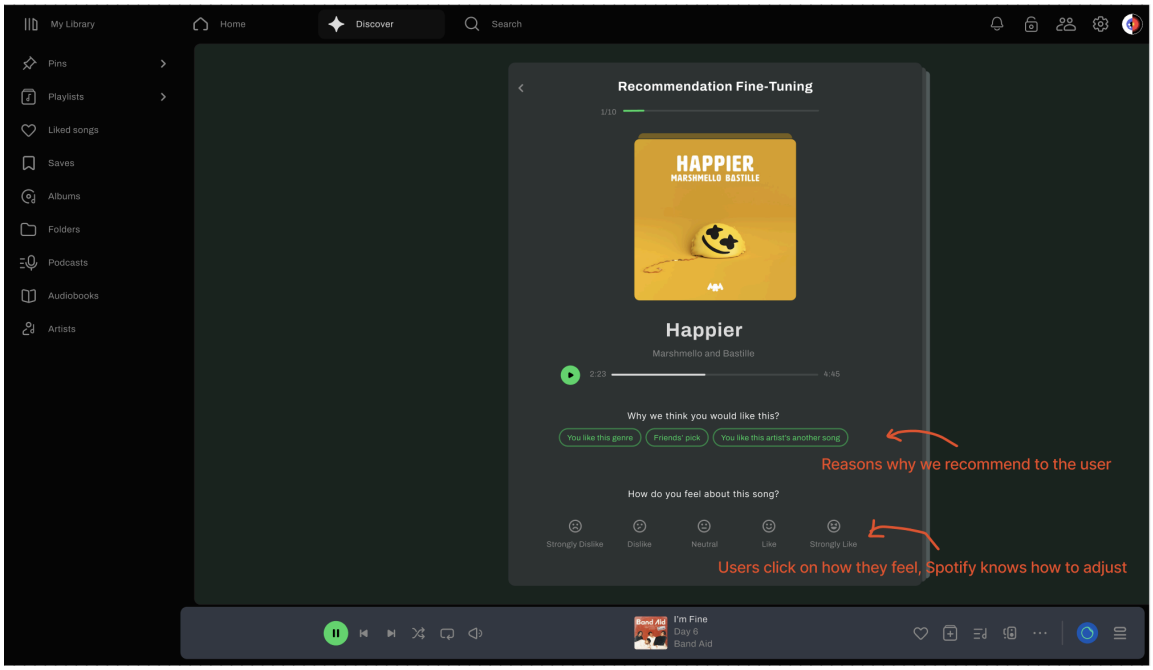


Figure 2.2

3. Recently Saved

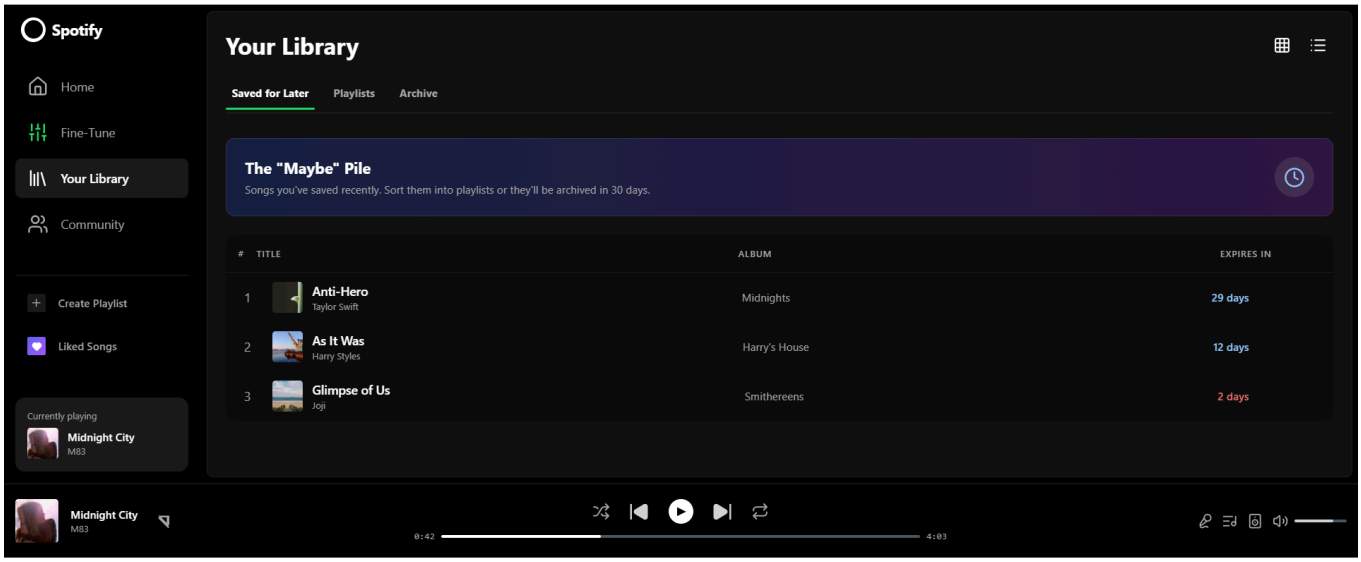


Figure 3.1

Storyboards

While organizing the data analysis from our contextual inquiries, we noticed patterns, themes, and key findings. These insights guided us in developing storyboards based on the participants' data. Some of the main conclusions we illustrated included Spotify's informational load within the interface. Participants specifically stated that they prefer not to be distracted and wished that the homepage gave recommendations based on their interests. They often complained that some pages were filled with unnecessary and/or unwanted information, affecting their algorithmic styles, cognitive load, and customization. There were also complaints about the algorithm's inaccuracy and its recommendations of certain songs. Users also struggled to organize the playlist of liked songs because it was too large. However, some users found it easier to like a song when driving.

Consistent with this, the storyboards below provide visual insights into these prominent issues and suggest approaches to address these setbacks. In Figure 4.1, a user named Bianca begins her day by searching for a song on Spotify's homepage. She notices the amount of information on the homepage and becomes frustrated by it. Bianca then decides to find a song on the recommended page. However, she quickly discovers that Spotify is inconsistent in its recommendations page. Some of her disliked songs, genres and artists are recommended to her. At this point Bianca is highly dissatisfied with Spotify's algorithmic inconsistencies and cluttered interface.

The illustration in Figure 4.2 focuses on the algorithm refiner solution. John is upset because he believes skipping and liking songs should have a greater effect on the algorithm. However, Spotify doesn't know whether skipping a song indicates that the user dislikes it or simply doesn't want to listen to it at that moment. John's friend, Samantha then informs John about the new algorithm refiner, which requires swiping left or right on the song, thereby directly affecting the algorithm. This way, users can like or skip a song without affecting the algorithm, while also allowing them to control the algorithmic style directly. In Figure 4.3, a user named Snookie is currently listening to a song, but she wants to save it for later rather than add it. Unfortunately, Spotify does not offer a "save for later" feature,

inevitably causing Snookie to add the song to her playlist. Eventually, Snookie wants to play the song she listened to earlier, but she is having difficulty locating it in her playlist. She expresses that her playlist contains more than 1,000 songs, making it overwhelming to locate a specific song. She attempts to navigate through her playlist, but she has no luck. After another disappointing outcome, Snookie wishes Spotify offered a “save for later feature” to ensure she is provided with flexibility, comfort, and convenience when searching for and saving a specific song.

In the last storyboard, Figure 4.4 shows Snookie from before, but with our new feature. Snookie begins in the same way, but she wants to save the song she is currently listening to for later rather than adding it to her playlist. Fortunately, Spotify has a recently saved playlist that she can add it to. This playlist stores these songs and periodically reminds the user to sort approximately five songs to avoid becoming overwhelmed. After 30 days, these songs are moved to an archive, where Spotify will no longer remind the user to sort them. This helps users feel less overwhelmed by sorting numerous songs at once and creates a sense of urgency to organize, given the 30-day limit. However, after 30 days, the songs do not disappear, which will not affect the user's interaction within the app.

Storyboard - Homepage Clutter

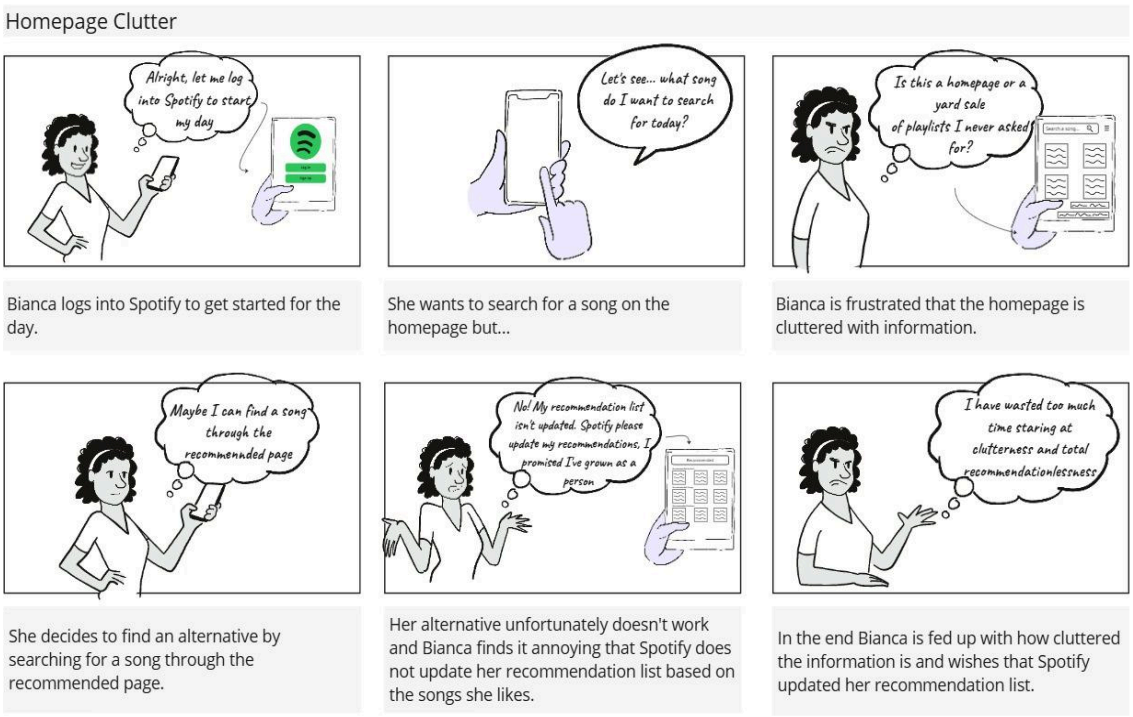

Homepage Clutter
Alright, let me log into Spotify to start my day
Bianca logs into Spotify to get started for the day.
Let's see... what song do I want to search for today?
She wants to search for a song on the homepage but...
Is this a homepage or a yard sale of playlists I never asked for?
Bianca is frustrated that the homepage is cluttered with information.
Maybe I can find a song through the recommennded page
She decides to find an alternative by searching for a song through the recommended page.
No! My recommendation list isn't updated. Spotify please update my recommendations, I promised I've grown as a person
Her alternative unfortunately doesn't work and Bianca finds it annoying that Spotify does not update her recommendation list based on the songs she likes.
I have wasted too much time staring at clutterness and total recommendationlessness
In the end Bianca is fed up with how cluttered the information is and wishes that Spotify updated her recommendation list.

Figure 4.1

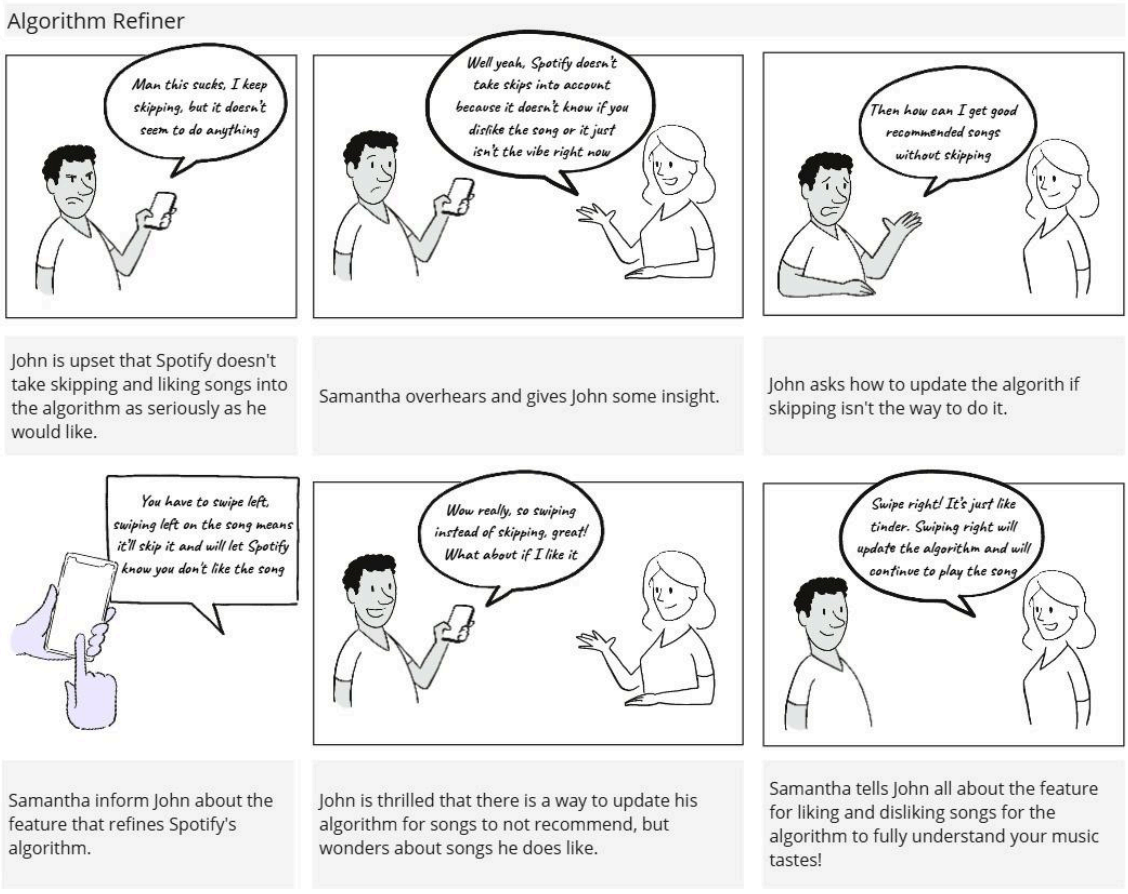

Algorithm Refiner
Man this sucks, I keep skipping, but it doesn't seem to do anything
John is upset that Spotify doesn't take skipping and liking songs into the algorithm as seriously as he would like.
Well yeah, Spotify doesn't take skips into account because it doesn't know if you dislike the song or it just isn't the vibe right now
Samantha overhears and gives John some insight.
Then how can I get good recommended songs without skipping
John asks how to update the algorith if skipping isn't the way to do it.
You have to swipe left, swiping left on the song means it'll skip it and will let Spotify know you don't like the song
Samantha inform John about the feature that refines Spotify's algorithm.
Wow really, so swiping instead of skipping, great! What about if I like it
John is thrilled that there is a way to update his algorithm for songs to not recommend, but wonders about songs he does like.
Swipe right! It's just like tinder. Swiping right will update the algorithm and will continue to play the song
Samantha tells John all about the feature for liking and disliking songs for the algorithm to fully understand your music tastes!

Figure 4.2

Storyboard - Recently Saved Playlist

Figure 4.3

Storyboard - Recently Saved Playlist

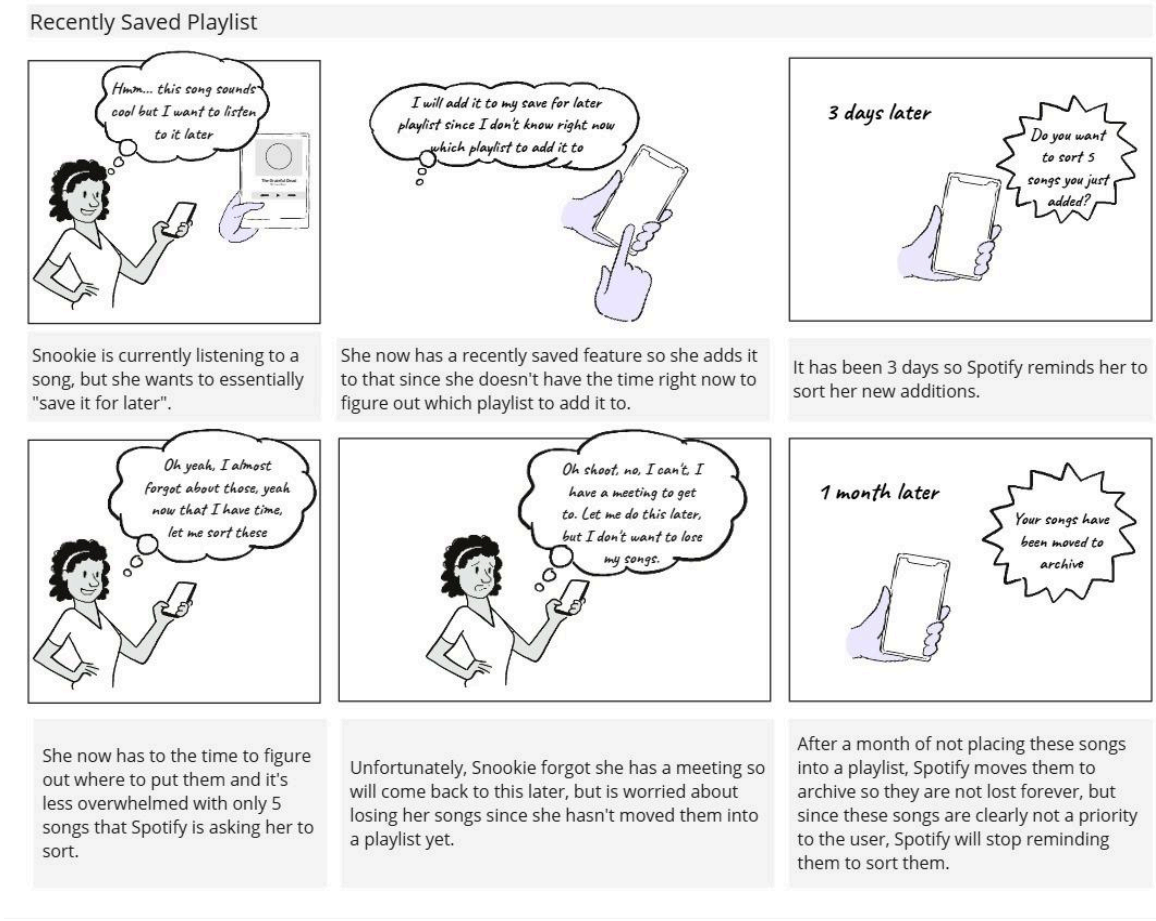


Figure 4.4

Use Cases

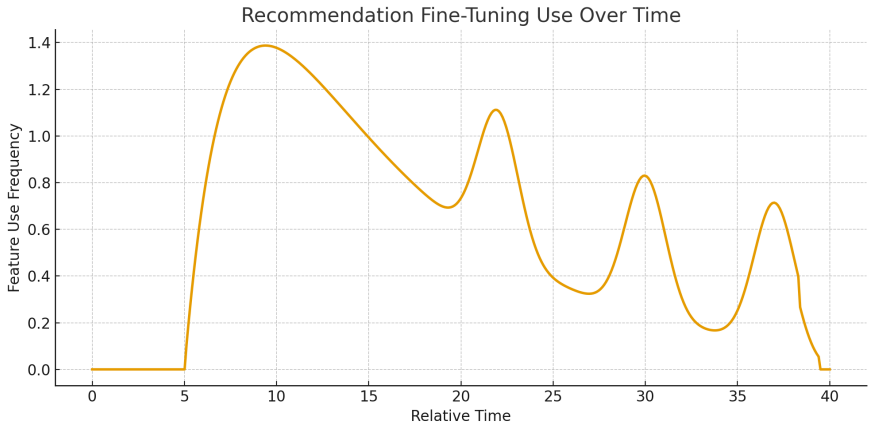
Use Case 1: Recommendation Algorithm Fine-Tuning

Use Case ID:		1	
Use Case Name:		Recommendation Algorithm Fine-Tuning	
Created By:	Sidney G	Last Updated By:	Sidney G
Date Created:	2025-12-02	Date Last Updated:	2025-12-05

Actor:	A music listener who uses various recommendation algorithm features
Description:	This feature allows users who are dissatisfied with their music recommendations to fine-tune the recommendation algorithms, providing direct corrective controls, transparency, and explainability.
Preconditions:	● The user has listened to sufficient music for the recommendation algorithms to have created customized recommendation profiles based on their listening history, expressed likes and dislikes, and other metadata. ● The user is dissatisfied with the current recommendation algorithm suggestions.
Postconditions:	● The recommendation algorithm/model has been fine-tuned. ● The user is satisfied with the quality of the recommendation algorithm suggestions.
Priority:	● Medium-High ○ High impact on affected users, these users are likely to seek different products. ○ High implementation difficulty (implementing machine learning fine-tuning)
Frequency of Use:	● Depends on the user and the algorithm. ○ Ideally, the algorithm gets it right, and the user never needs the feature ○ If the algorithm consistently recommends songs the user dislikes, they may use the feature occasionally. ● Divergent, problem-triggered uptake with latent dropoff. ○ Group 1: ■ Never used by users who are happy with recommendations ○ Group 2:

<table>
<tr><td></td><td>

Initially unused, the algorithm requires time to learn, and then more time for the user to realize that it's not working as intended.
Initial spike in usage for corrective intervention
Drop-off in use as the recommendation algorithm aligns with user preferences
Periodic use in the future, if the algorithm drifts or user preferences change over time



</td></tr>
<tr><td>Normal Course of Events:</td><td>

After some time streaming music, the user notices that the recommended songs do not match their preferences.
They exhibit an initial burst of fine-tuning activity for corrective action, followed by a gradual decline as the recommendations approach their preferences.
Interaction loop: make fine-tuning adjustments in the adjustment UI, then an evaluation period where the user receives recommendations and determines if additional fine-tuning is needed
While using the recommendation UI:

System presents artists, albums, and songs that it believes the user should like.
The user can explicitly indicate, on a 5-class feedback input, whether they "Love, Like, Neutral, Dislike, Hate" the music.
The system integrates feedback and generates new sample playlists and recommendations.

</td></tr>
<tr><td>Alternative Courses:</td><td>

Course 1:

User is content with their recommendation quality, and does not need to use the feature

Course 2:

User is content with the recommendation quality, but returns after the algorithm drifts over time

Course 3:

User is content with the recommendation quality, but

</td></tr>
</table>

	returns after their own musical tastes and preferences change over time
Exceptions:	● Exception 1: ○ Insufficient listening history. Recommendation algorithms need an adequate threshold of input to provide customized suggestions or to fine-tune them. ● Exception 2: ○ Exhaustion of music to recommend. Unlikely, probably caused by erroneous input, marking entire genres as disliked.
Includes:	● Algorithmic recommendation features: ○ Discover Weekly ○ Release Radar ○ Daily Mixes ○ Spotify Radio ○ Personalized Mixes ○ Autoplay ○ Spotify DJ
Special Requirements:	● Explainability: The fundamental problem is that recommendation algorithms are a black box. The feature should provide context for why specific genres, artists, or songs are recommended, so that the user can understand and correct any erroneous assumptions. ● Non-destructive: The fine-tuning must enable the user to experiment with recommendation data without fear of degrading performance. Changes must be transparent and easily reversible for users to use the feature with confidence. ● Intuitive: The feature must be powerful enough to allow the user to fine-tune an algorithm that's gone awry, but it must not be intimidating or require technical expertise. ● Latency and transparency: It may take some time for the algorithm to update. It must be made clear to users whether the fine-tuned model is in place; otherwise, they may encounter errors when attempting to fine-tune further a model that has not yet been updated with their recent adjustments. ○ A lower-fidelity model that can provide sample output within seconds should be made available.
Assumptions:	● Users understand the concept of expressing musical preferences using like/dislike. ● The system does not require technical expertise or musical expertise ● User music preferences are typically stable or change gradually over time

Notes and Issues:	● Is there a risk of over-tuning? This may result in a narrow and uninformative pool of recommendations. ● What is the appropriate level of detail for classifying likes and dislikes? Binary Like/Dislike? Trinary Like/Neutral/Dislike? 5-class Love/Like/Neutral/Dislike/Hate? Will users want or benefit from the granularity? ● Should users be able to train more than one recommendation algorithm for different moods or occasions? ● How will the system work for users who listen to already niche genres and have a smaller pool of music from which to choose?

Use Case 2: Homepage Customization

Use Case ID:		2	
Use Case Name:		Homepage Customization	
Created By:	Sidney G	Last Updated By:	Sidney G
Date Created:	2025-12-04	Date Last Updated:	2025-12-07

Actor	Spotify User
Description:	This feature allows users to customize the homepage to their preferences using a new “Widget” system. They may eliminate irrelevant sections, rearranging widgets in their preferred order, and add new widget variants with content from various sources.
Preconditions:	User: ● The user finds the homepage cluttered, missing relevant information, or arranged non-optimally System: ● The system has a set of configurable widgets ● Widgets can be toggled on or off ● Widgets can be rearranged ● Widgets include options for data sources and visual layouts
Postconditions:	● The homepage state has been persisted. ● The homepage has been customized to the users’ preferences. ● The user is satisfied with the homepage content.
Priority:	● High ○ Extremely high visibility feature (Homepage) ○ Easy implementation difficulty ○ Mentioned by many participants

Frequency of Use:	● The homepage is used frequently (at least once per session). ● Customization is used periodically, based on user preferences.
Normal Course of Events:	● User realizes the homepage is not showing them the content they wish they were seeing ● The user removes some irrelevant widgets from the homepage, such as podcasts or popular artists. ● The user uses drag-and-drop functionality to reorder the widgets, placing the most important ones at the top of the homepage. ● The user goes into the settings pane to add new widgets. ○ They customize the layout of some widgets (e.g., change a horizontal row to a grid) ○ They customize the data source for certain widgets (e.g., add a Popular Rock Songs widget rather than the default, which covers all genres).
Alternative Courses:	● Course 1: ○ User is content with the default homepage, and does not need to use the feature ● Course 2: ○ The user experiments with customization but exits before finishing. ○ System restores the previous state. ● Course 3: ○ The user hides various widgets but ultimately decides they liked them after all. ○ The user toggles the widgets' visibility back on to restore them.
Exceptions:	● Exception 1: ○ The user selects a layout type for a widget suitable for a large display (e.g., a grid) but not for a small display. ○ Later, the User loads their homepage on a small display. ○ Large layouts gracefully degrade to more compact layouts on compact form-factor devices.
Includes:	● Various algorithmic recommendation features are used to generate content for the homepage widgets: ○ Daily Mixes ○ Discover Weekly ○ Release Radar ● Song Organization Widget from **[Use Case 3]** intersects with this use case. ○ This widget makes organizing recently saved songs a "bite-sized" task by forwarding just a few unsorted, saved songs to manage in a prominent location on the homepage.

	● Widget configuration system
Special Requirements:	● Non-destructive: Changes must be reversible, so the user can feel comfortable with experimenting. ○ For example, by default, “deleting” should toggle widget visibility state rather than truly deleting a widget. ● Highly customizable: The user should be able to make the homepage feel truly like home. ○ They should be able to remove any widget, customize a widget's contents, create variants of a widget, reorder widgets, and modify a widget's visual presentation or layout.
Assumptions:	● Users vary: some will never wish to customize the homepage, while others will do so frequently. The tools should be neither intrusive nor difficult to access.
Notes and Issues:	● Have we covered all the types of widgets that a user may want on their homepage? Are there too many or not enough? ● Will this feature be sufficiently discoverable and intuitive, especially the more complex customization settings?

Use Case 3: Recently Saved Playlist & Archive

Use Case ID:		3	
Use Case Name:		Recently Saved Playlist	
Created By:	Sidney G	Last Updated By:	Sidney G
Date Created:	2025-12-04	Date Last Updated:	2025-12-07

Actor	Spotify users, especially those who tend to store large quantities of songs for later in the default playlist.
Description:	This feature introduces a second, dynamic default playlist called “Recently Saved,” which appears alongside “Liked” in first-class UI locations. Instead of placing every “save for later” song in Liked, users can save songs to Recently Saved with minimal commitment. A rolling time window keeps the playlist manageable rather than an ever-growing junk drawer, and encourages users to organize or

	promote songs periodically. Meanwhile, older items are moved to an Archive playlist to prevent loss.
Preconditions:	● The user wishes to save a song for later but does not wish to, or is unable to, select a specific playlist in which to save it at that moment. System: ● Save is a distinct action from Like
Postconditions:	Immediate: ● The song has been saved ● The song is *not* automatically marked as Liked. This is a distinct interaction. Deferred: ● Saved songs are preserved in the dynamic Recently Saved playlist for 1 month ● Later, they are moved to a dynamic Archive playlist
Priority:	Medium-High ● High-use feature that causes widespread dysfunctional workflows ● Interaction with [Use Case 1] recommendation algorithms significantly affects the user experience, reducing recommendation quality.
Frequency of Use:	● Very frequent. Users may save numerous songs per session.
Normal Course of Events:	● A user hears a song they want to save for later, but they are busy or don't wish to decide where the song belongs in the moment. ● The user is unsure whether they like the song or doesn't want to think about it, so they press the Save button. ● Later, the user browses their Recently Saved songs to refer back to them. ● They are now in an environment where they can think about the music, unlike when the song was initially playing. ● They notice that some songs have been saved almost a month ago, and the system reminds them of this with a section towards the top of the playlist of songs that will soon be moved to the Archive playlist. ● The user sorts the songs into appropriate playlists.
Alternative Courses:	● Course 1: Organization via Homepage ○ User sees the Song Organization Widget [see **Use Case 2**] on their homepage ○ This widget presents them with a brief task to organize just a few recently saved songs, which seems straightforward. ○ User sorts the songs into playlists

	● Course 2: The User does not organize ○ User saves songs, but does not return to organize them later ○ The user rarely or never reviews saved tracks ○ The system automatically moves songs into the Archive ○ Recently Saved remains timely and compact if the user chooses to reference it, whereas Archive returns the full catalog of saved songs.
Exceptions:	N/A
Includes:	● Interaction with various recommendation algorithms [**Use Case 1**] ○ Providing "Liked" as an easy, 1st-class playlist to save music for later trains the recommendation algorithm to predict that users "Like" songs they haven't yet decided on. ● Song Organization Widget **[see Use Case 2]** intersects with this use case. ○ This widget makes organizing recently saved songs a "bite-sized" task by forwarding just a few unsorted, saved songs to manage in a prominent location on the homepage.
Special Requirements:	● Should apply a low-stakes sense of urgency: ○ Ethnographic studies showed that users accrue thousands of songs in the default playlist ○ Default playlist gets hit hard because it's easy at the time of listening ○ Becomes insurmountably large and overwhelming over time ○ A dynamic, rolling time window (1 month) keeps the playlist small and timely ○ Paired, dynamic Archive playlist ensures the user is not punished for failure to act in time ● Distinguish between Liked and Saved in default playlists: ○ Both are ways to preserve songs to find later easily ○ The status quo of Liked-only frequently gets songs that aren't honestly liked, because it's easy to add to ○ This adversely influences recommendation algorithms and organization schemes ● Keep the task of organization bite-sized, not daunting
Assumptions:	● Users wish to preserve songs to find again later easily ● Users often can't or don't want to decide on permanent, specific organization destinations for songs at the time the song is played.

	○ Users are often willing to perform light, low cognitive load interactions to mark a song for reference later in these same contexts ● Users are more likely to organize songs if they have to manage fewer of them at once ● Users will understand the semantic difference between saving a song and liking a song.
Notes and Issues:	● Is the 1-month window appropriate? ● Will users be confused or frustrated by songs moving from Recently Saved to Archive? Will they realize that they can still access these songs?

Prototype Screenshots

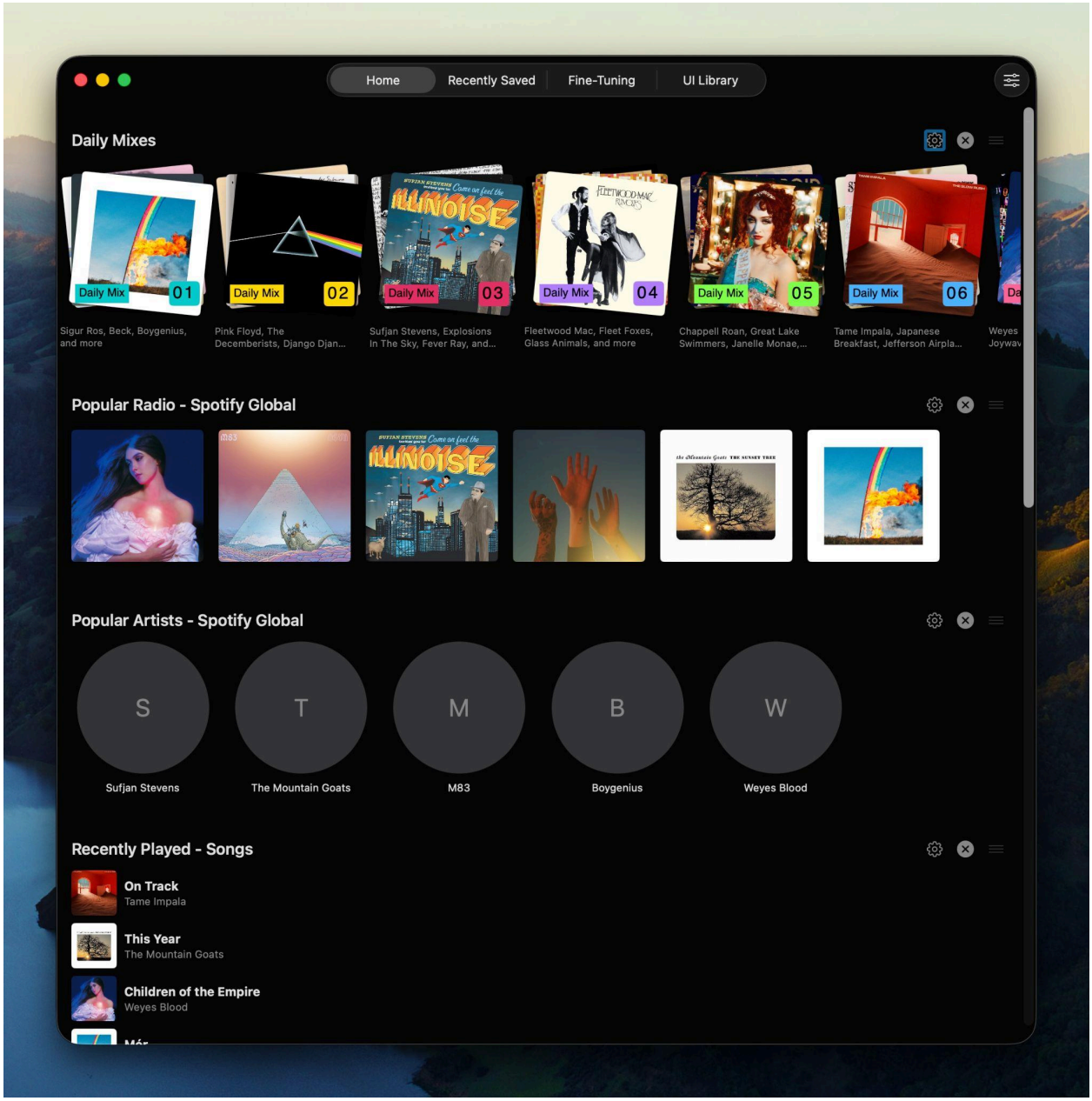


Figure 5.1: Redesigned customizable MacOS Homepage

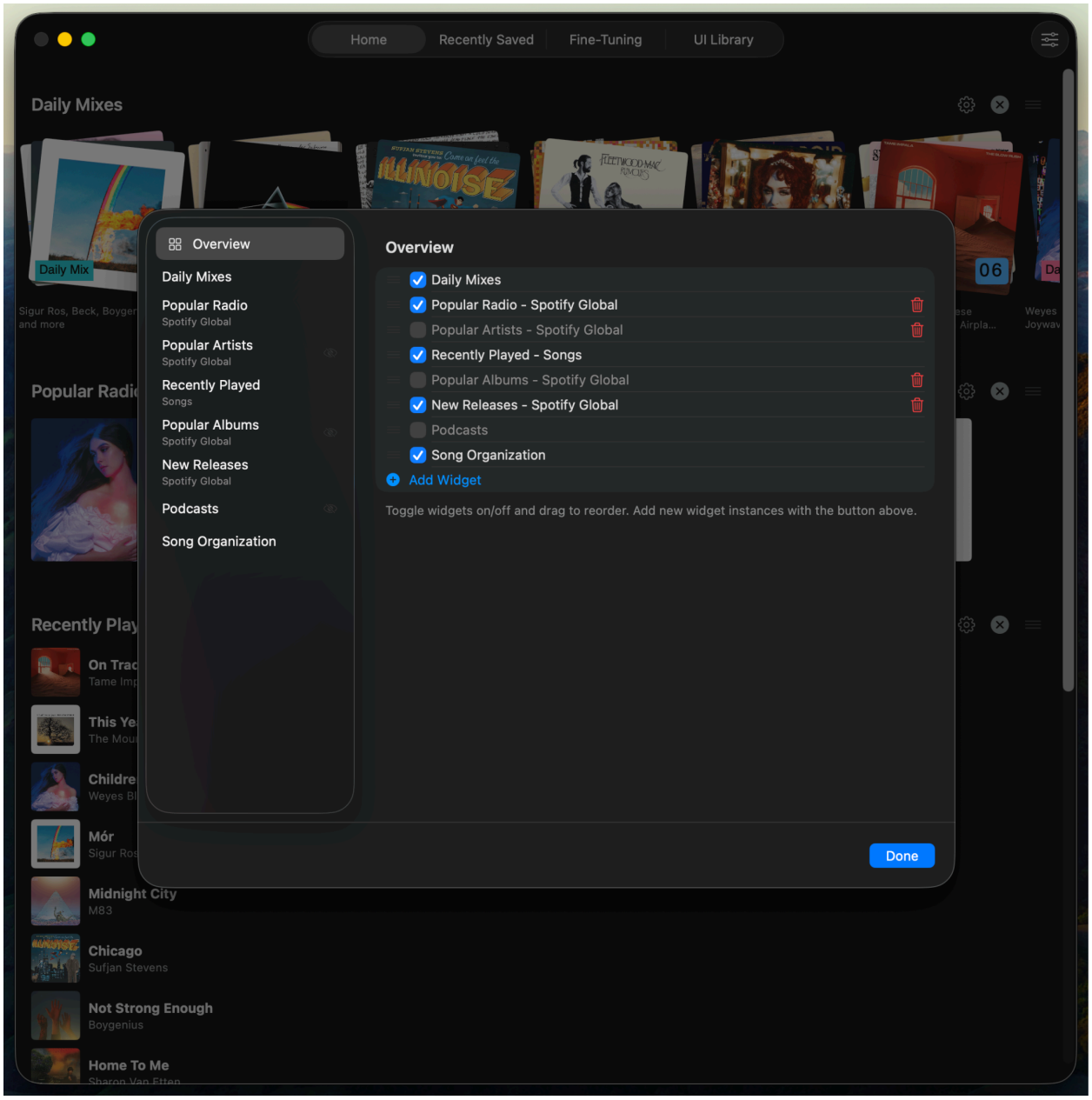


Figure 5.2: Widget Overview for redesigned customizable Homepage

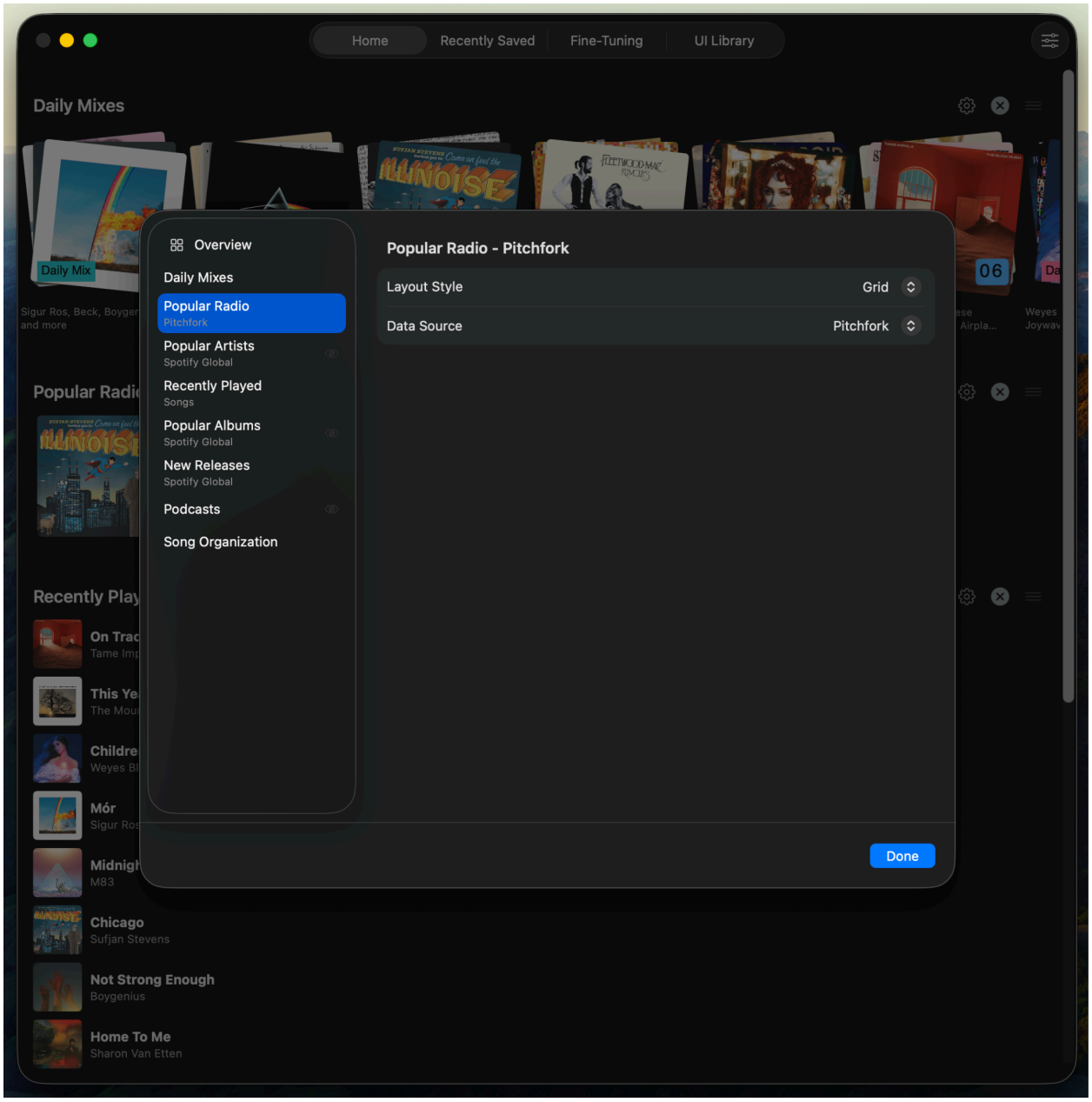


Figure 5.3: A customized widget, with Pitchfork magazine as the data source for popular radio

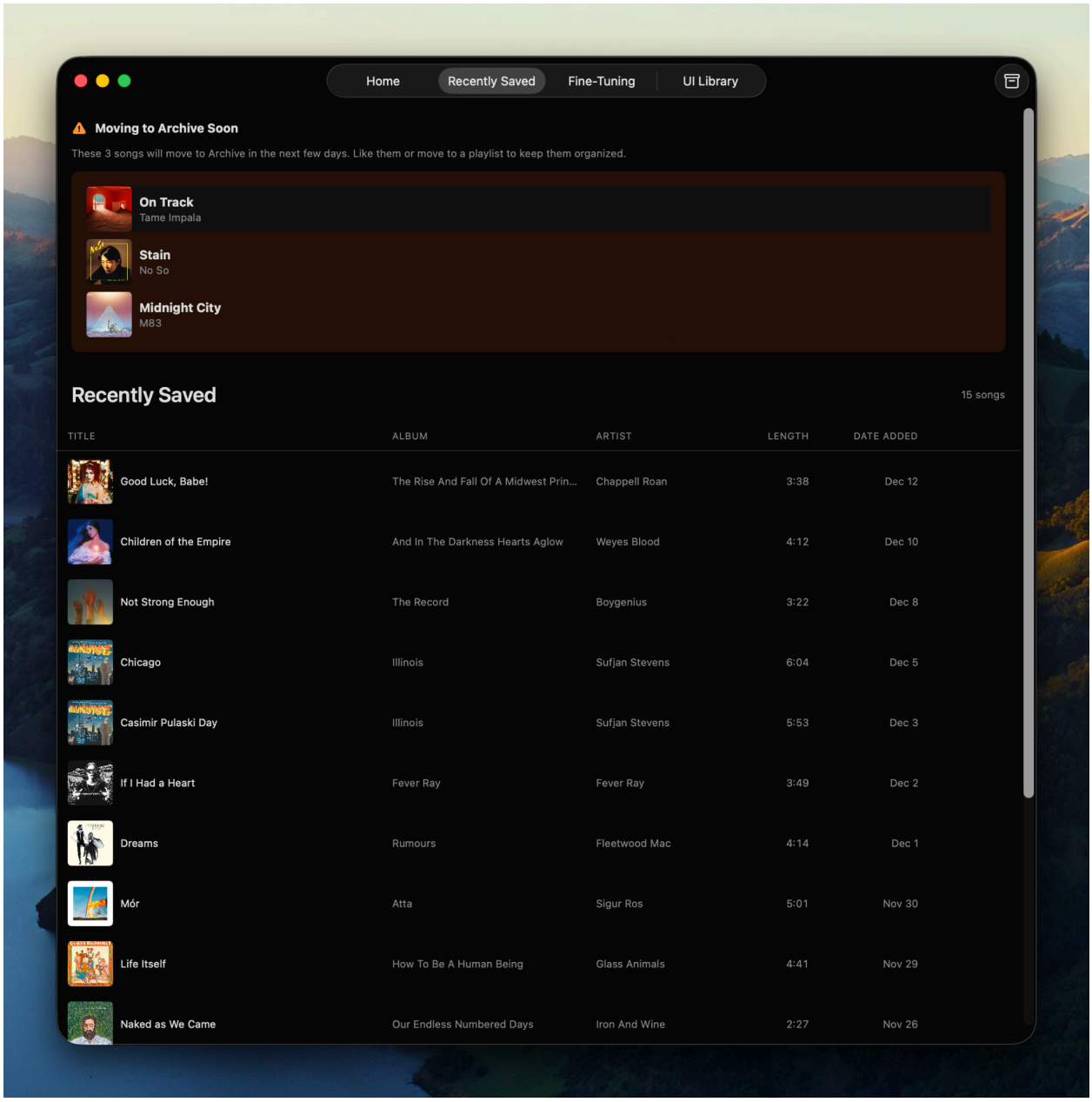


Figure 5.4: Redesigned dynamic “Recently Saved” playlist

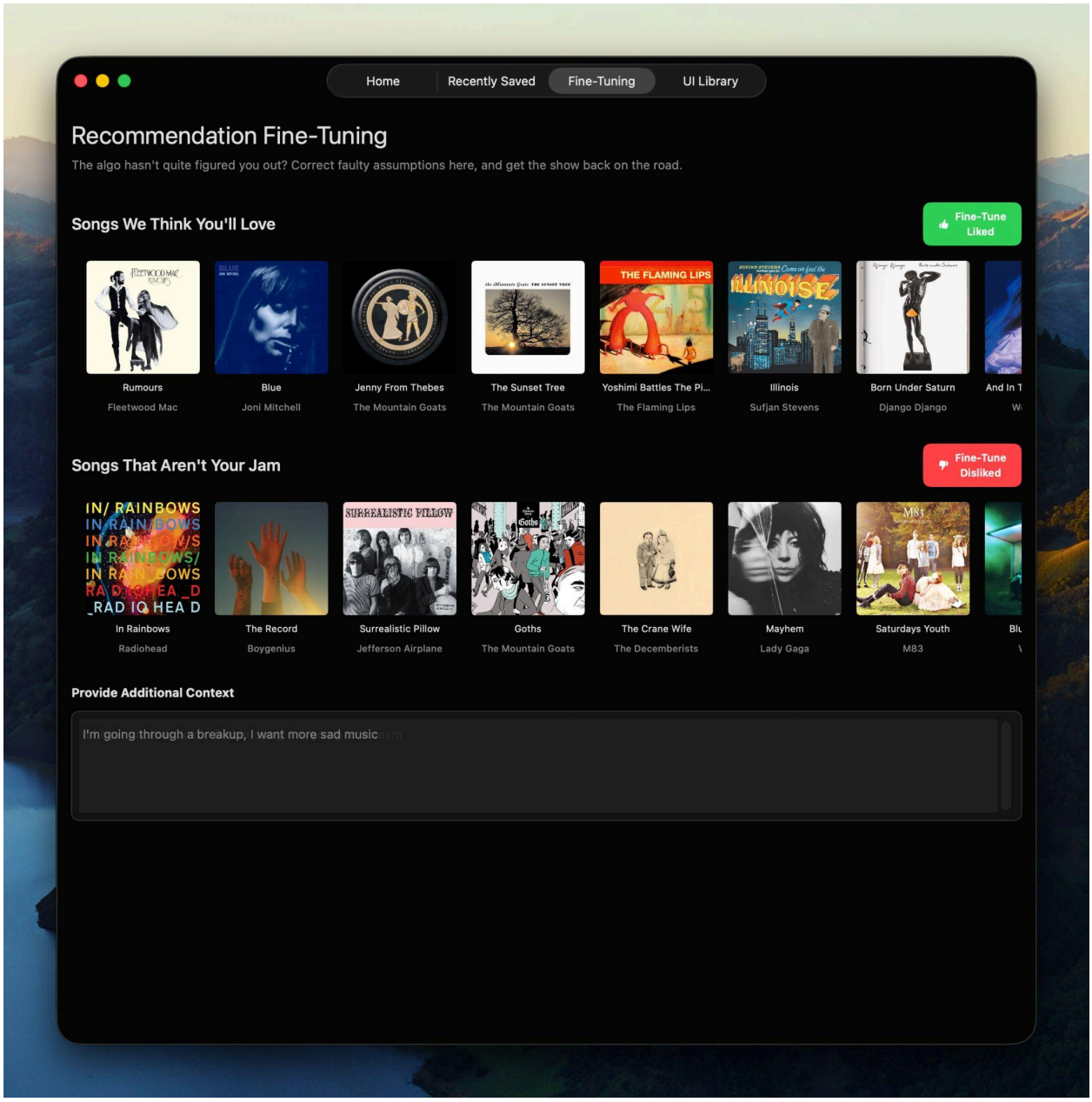


Figure 5.5: New Recommendation Fine-Tuning UI with auto-generated user taste snapshot playlists

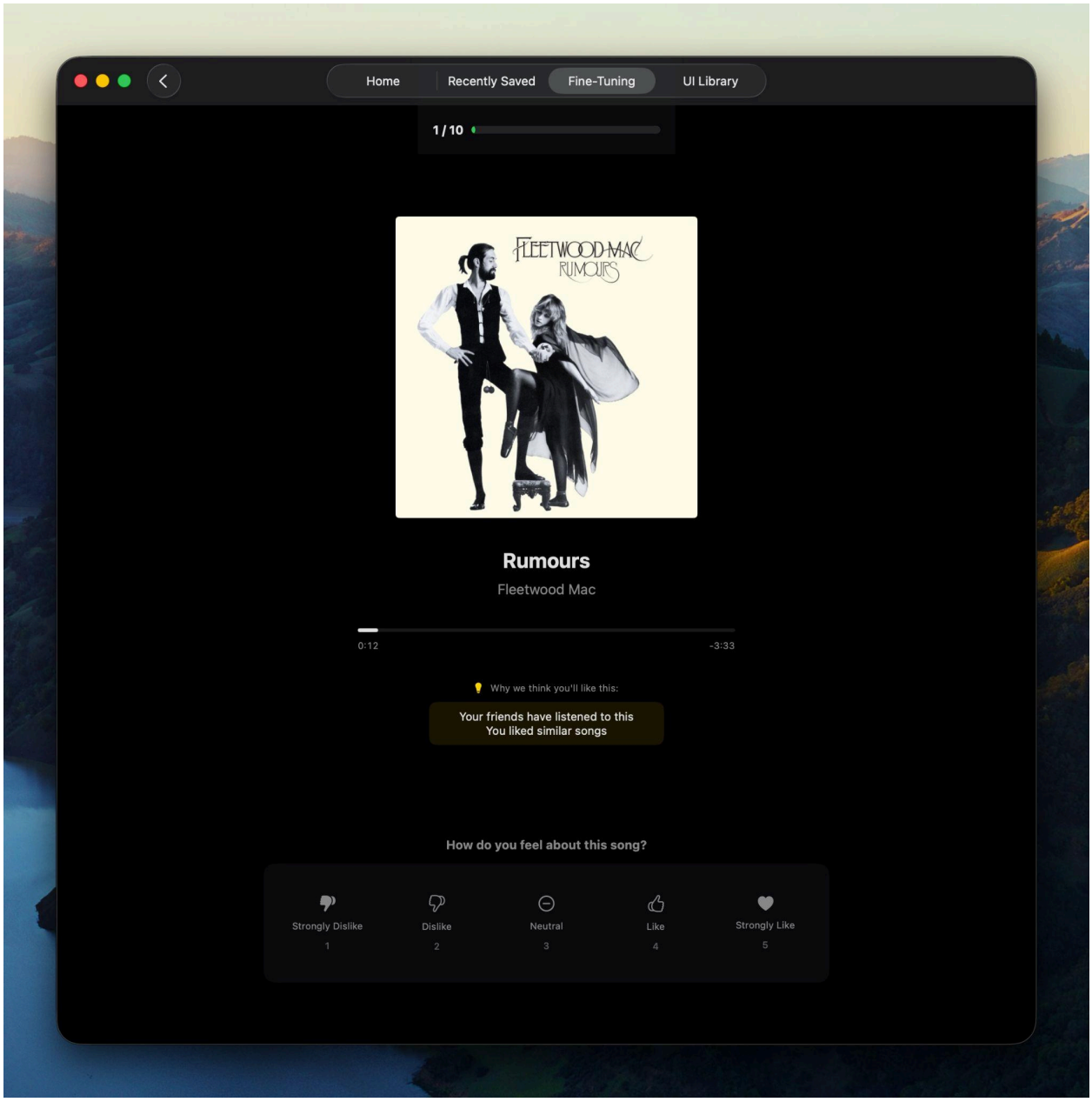


Figure 5.6: New Recommendation Fine-Tuning UI, showing the process of explicitly training the recommendation algorithm.

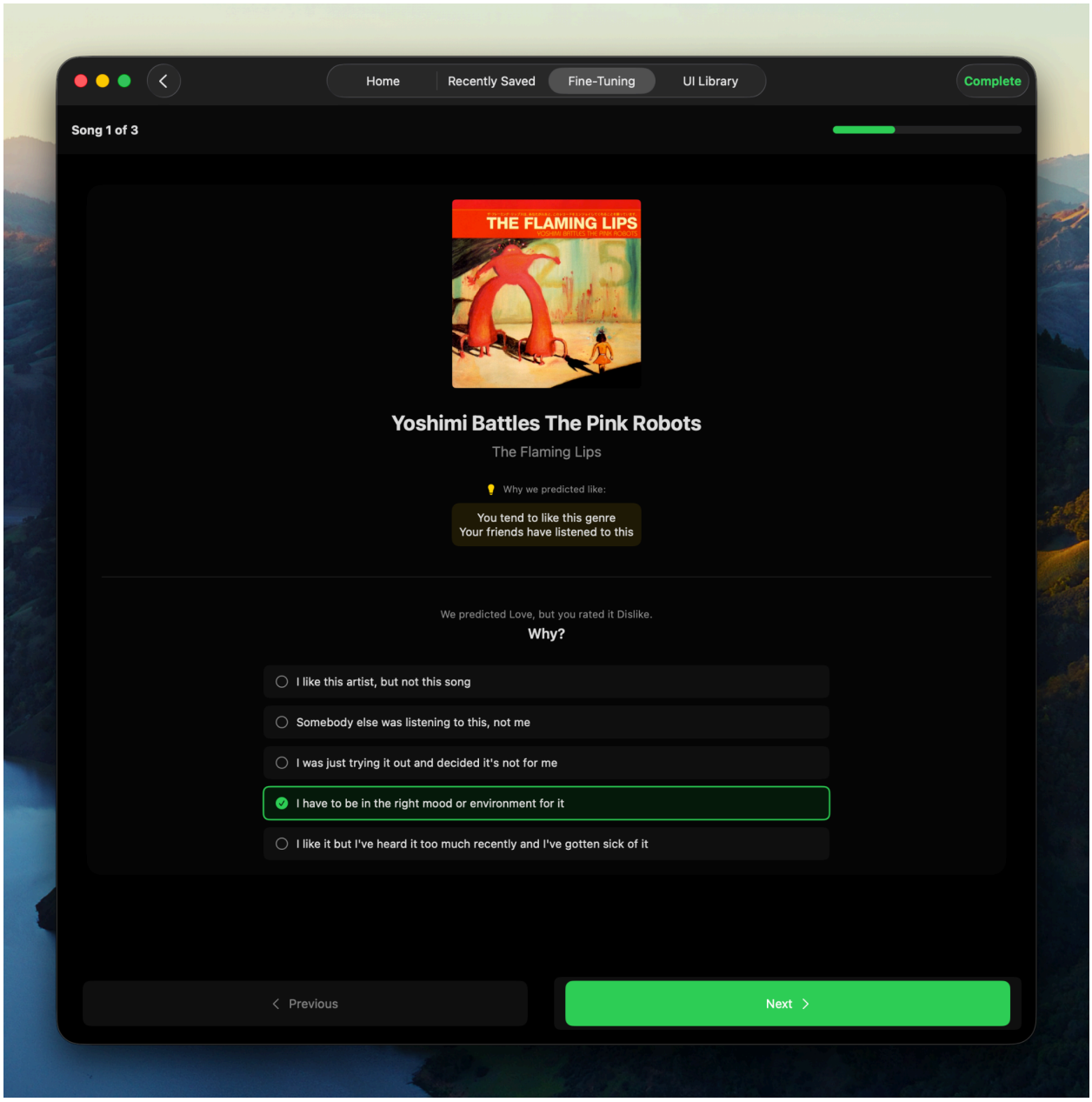


Figure 5.7: New Recommendation Fine-Tuning UI, showing the optional additional feedback after rating songs. This is provided just for the songs that the algorithm misidentified (a disliked song presumed liked, or vice-versa).

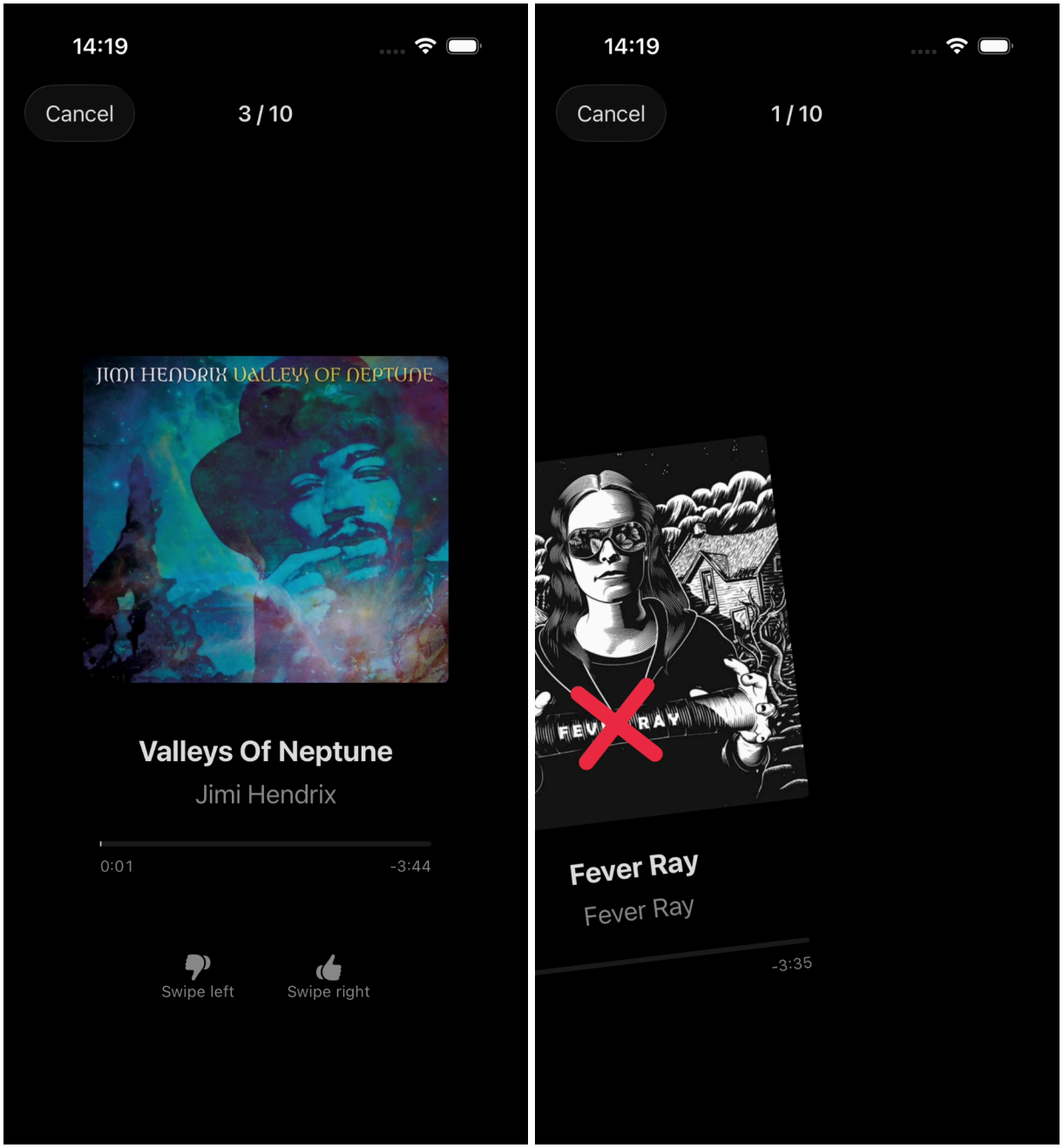


Figure 5.8: New Recommendation Fine-Tuning UI, iPhone/iOS version with Tinder-inspired swipe UX. (Left) initial view, (Right) swipe-gesture to classify a track in progress.

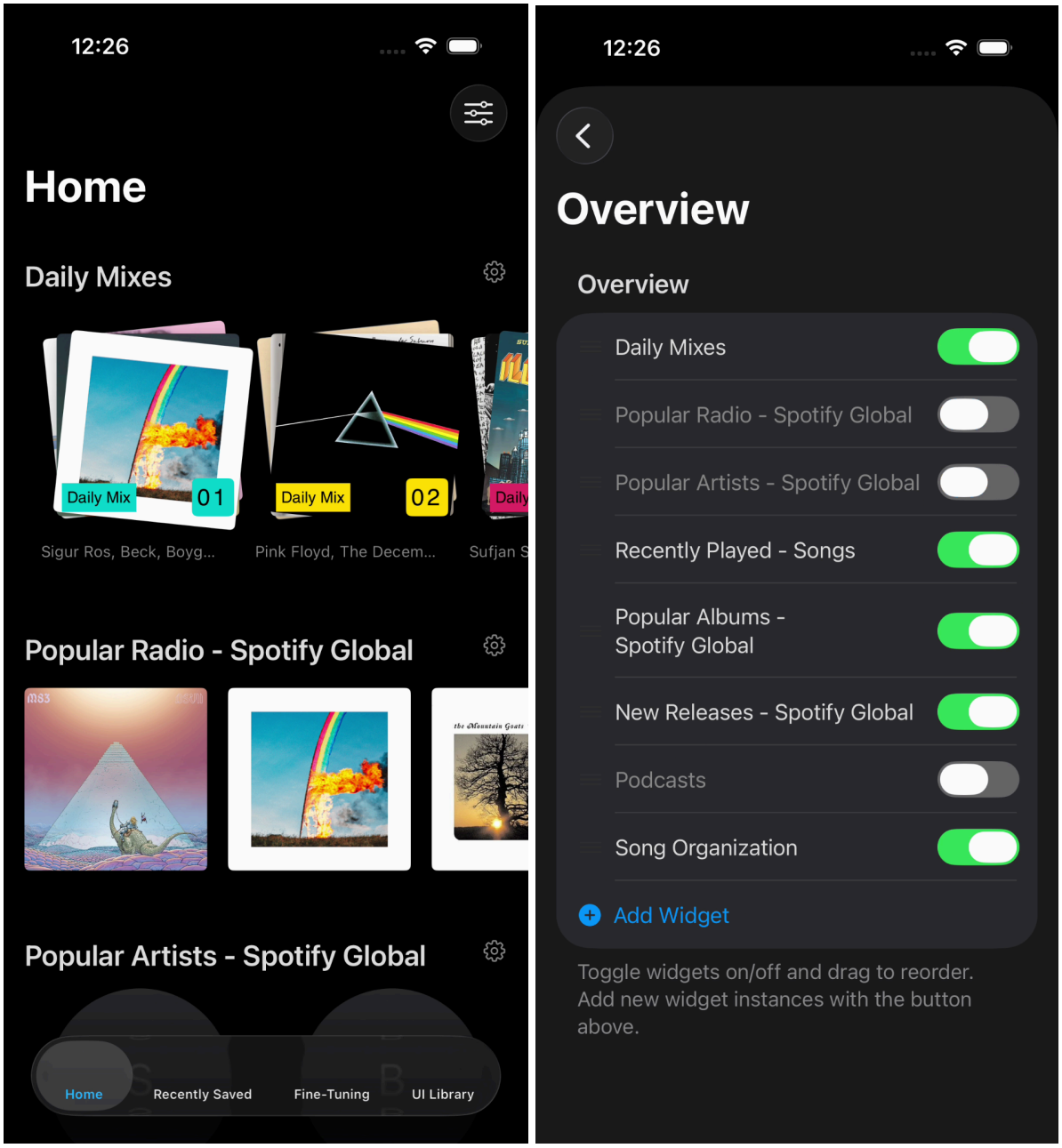


Figure 5.9: Redesigned customizable homepage, iPhone/iOS version. (Left) Homepage does not show the “Delete” button, as it’s easy to accidentally tap on a touch screen. (Right) Form inputs for customizing widgets are optimized for smartphone controls, using standard iOS components.

Conclusion

This project demonstrates how insights from contextual design can be translated into a cohesive, user-centered redesign that meaningfully improves the music discovery experience. By focusing on high-impact pain points, homepage clutter, limited algorithmic feedback, and overwhelming song organization, we developed a vision that prioritizes user agency while remaining grounded in feasible system behaviors. Our storyboards illustrate how users can move fluidly between discovery and familiarity, while our use cases formalize these interactions into scalable system-level requirements.

Together, the proposed homepage customization, recommendation fine-tuning, and recently saved playlist features support a more transparent, adaptable, and humane discovery workflow. Rather than forcing users to conform to rigid algorithmic assumptions, our design allows Spotify to reflect users' evolving tastes, listening contexts, and organizational needs.

Ultimately, this work redesign positions Spotify as a more responsive partner in music discovery, one that empowers users to explore, curate, and enjoy music on their own terms while reducing frustration and cognitive overload.