Design Architecture for Spotify Social Features

Rohan Somji

Abstract

"Dear 3,749 people who streamed 'It's the End of the World as We Know It' the day of the Brexit vote, hang in there." In 2016, flexing its 'user data' muscles, Spotify demonstrated in its ads that it now has the capacity to extract information from social affairs and use it to peer into personal moments of users and how these affairs influence their music choices.

My aim in this paper is to unearth the layers of design architecture involved for enabling the social capabilities a user has on Spotify. Along the way we will see how it relies on older concepts and systems that were here long before Spotify itself. We will discover that the design at the bottom level of its architecture (its backend, the system, the cloud services, streaming, data storage, etc) is the same kind as the one used by many social media companies. Spotify steers itself in a non-social media direction under the influence of the socio-cultural landscape, to maintain itself as a solely music & podcast streaming platform, using design at the top level of its architecture (affordances, constraints, conventions, icons, features, accessibility).

Introduction

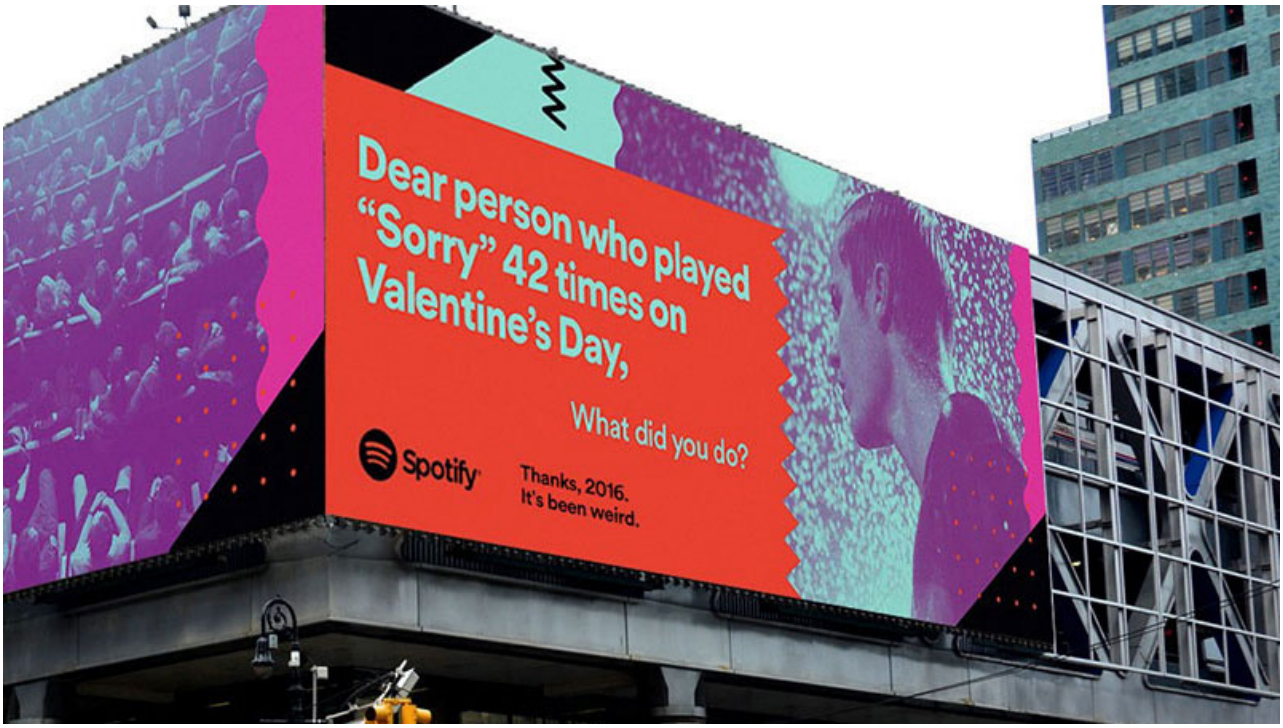
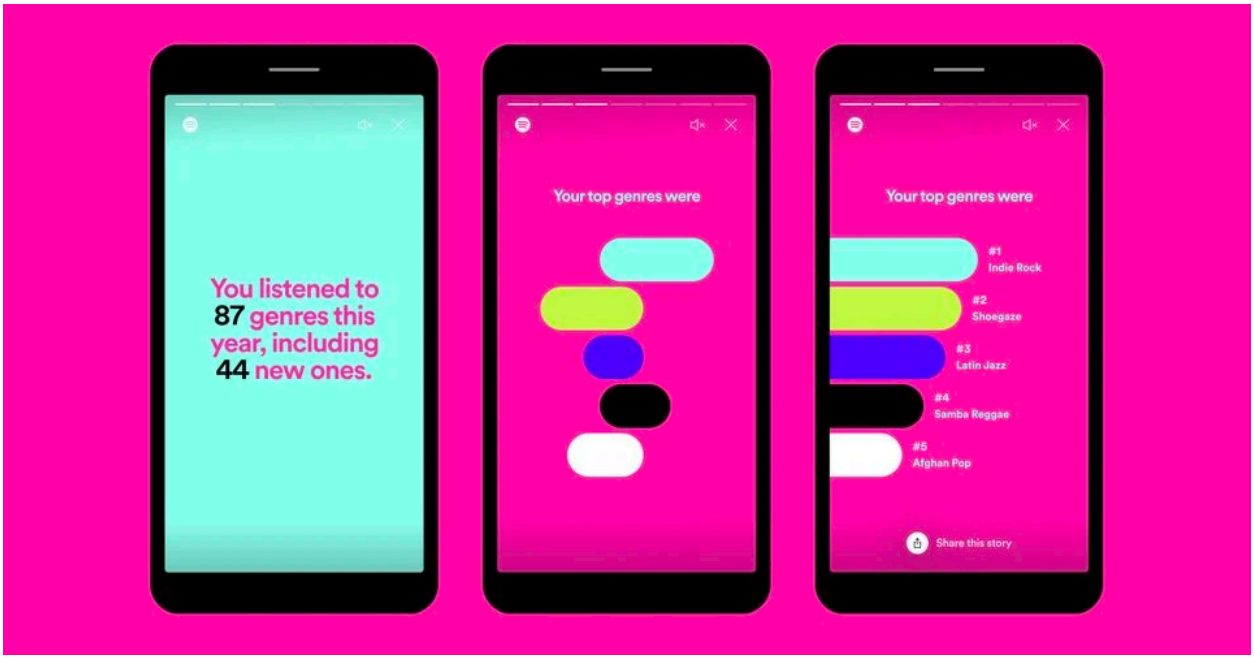


Fig: Spotify Ad

As one of the top music and podcast streaming services available, their user base has grown substantially over the last few years owing to the increase in available music as well as increase in services. As the user base grew so did a sense of community and a need for social bonding over music. This involves features to share on social networking platforms, features to see what your friends are listening to, collaborating on creating playlists, following each other's playlists,

etc. The data infrastructure underneath the audio and streaming infrastructure is becoming more relevant each and every day as user data grows and ML techniques are applied.

This year they are making it even more personal with their “2020 wrapped” providing more granular details on both music and podcast listening habits: how many total minutes users listened to podcasts, their top podcast with its listen-count, which songs were on repeat, which song you discovered before they went hit and more.



One of the benefits of using Spotify has always been its mobility. Smartphones made it easy to keep music portable by connecting the user to a central database of music collection available via streaming services on the internet. This means the user listens to music in a number of locations and during a wide range of activities. So do their friends on Spotify. Currently Spotify offers

accessibility to adding friends via facebook, following a friend's playlists, and being notified of a friend's listening activity on the desktop app.

How does all this data get tied up together to provide a highly structured user dataset, intelligible insights and accurate recommendations? What kind of design architecture allows sensible data to emerge from this constant in-flux and out-flux of data arriving from multiple separate interactions of users? What do the design decisions tell us about how Spotify wishes to operate?

Data Infrastructure

Spotify shut down the last of its US data centers in 2016 freeing itself of on-premise infrastructure and migrating onto Google's Cloud Computing Platform . The data centers existed to "send out music files and fetch back user data" (Eriksson et al,2019, 44). Now streaming and storage of their user data and music files is in Google's Cloud Storage. The cloud computing services provide the advantage of Google tools like BigQuery cloud data warehouse, Pub/Sub for messaging and DataFlow for batch and streaming processing. In the fourth quarter of 2019, Spotify reported 271 million monthly users and 124 million Premium subscribers, and all this data is stored in Google's Cloud Platform (Spotify Case Study (n.d.), Google Cloud Customer).

Clicking the play button, tapping on a playlist, using any affordance provided by Spotify results in an "event". Whenever a user performs an action in the Spotify client—such as listening to a song or searching for an artist— a small piece of information, called an event, is sent to their servers (Maravić, 2016). Spotify calls the process "Event delivery", which makes sure that all

events get transported safely from users to a central processing system, managed on Google Cloud (Maravić, 2016). Cloud Pub/Sub is the transport mechanism for all the events.

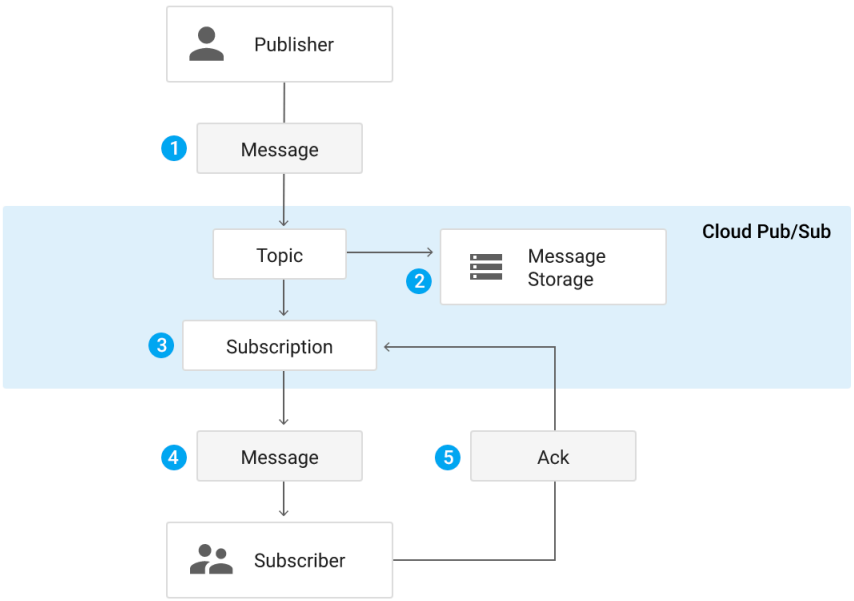


Fig: (What Is Pub/Sub? | Cloud Pub/Sub Documentation)

A pub/sub is a message-oriented middleware system where publishers send messages via a portal that categorizes published messages into classes on one end. The subscribers on the other end choose which messages to receive (What Is Pub/Sub? | Cloud Pub/Sub Documentation). Both do this without each other's knowledge. Here, messages can range from commands to payment information to subscriptions to premium accounts.

The pub/sub system as part of the internet "uses the bundles of data packed in smaller units" (Irvine, (n.d.), pp. 6). The packet itself has no information but much like carrier waves in radio

signals, it carries data that can be called the “payload”. This packet has other information in bits that determines its path in the network and the payload has the message stored in it which is received, and converted for the GUI on the other end (White, 258-259).

Connecting and accessing databases

User details, such as username, country, and email, are stored in a user database. Every time a user logs in, that database is queried (Vesterlund, 2015). Spotify uses Apache’s Cassandra and Hadoop to store user profile attributes and metadata about entities like playlists, artists, etc. (Mishra & Brown, 2015). These software libraries (by Apache) are frameworks that allow for the distributed processing of large data sets across clusters of computers using simple programming models. Similarly, it uses these software libraries for the following (Mishra & Brown, 2015):

Log collection like “completion of a song or delivery of an ad impression,” – Kafka
Real-time event processing like “clicks” “search” – Storm
To remove duplicate events, clean up data to “generate metadata like genre, tempo,” – Crunch
Store user profile attributes and metadata like playlists, artists, etc. – Hadoop and Cassandra
It’s the Storm pipelines that fetch the metadata back, group it per user and determine user level attributes to represent a user’s profile which is then stored in a Cassandra Cluster. Spotify calls this the User Profile Store (UPS).

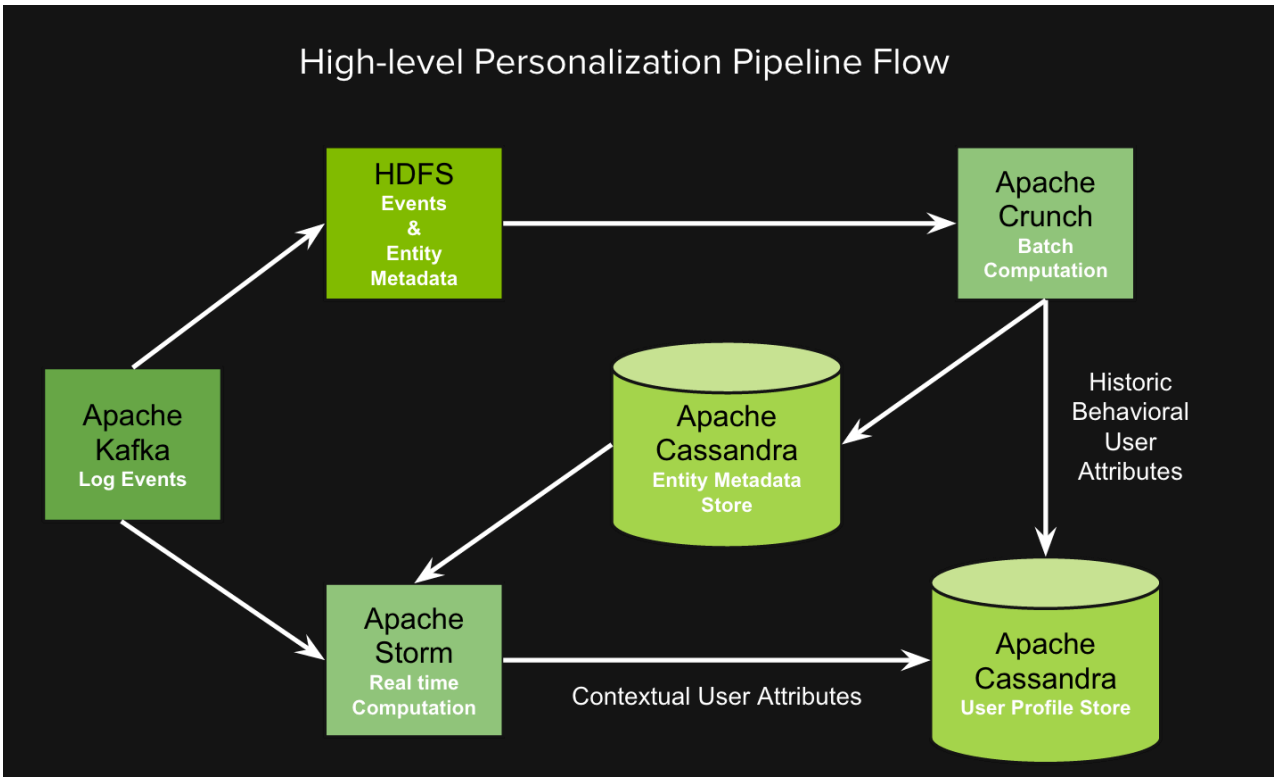


Fig: (Mishra & Brown, 2015)

When a user is listening in real-time, Apache Storm is at work. A large volume of data moves across all the libraries, while being worked on by Spotify's recommender algorithm, to generate "Recently Played" songs, curated playlists like "Discover Weekly", "Shows to try", "Recommended Radio", etc.

The deployment of these databases along with cloud-based pub/sub system collectively called Event Delivery system are used to generate and move critical data. These include EndSong Event (an event emitted when a Spotify user is done listening to a track), which is used to pay royalties to labels and artists, calculate Daily Active Users (DAU), Monthly Active Users

(MAU); and the UserCreate Event (an event indicating a new Spotify user account was created) (Janota & Stephenson, 2019).

Connecting users to each other

Users can integrate their Spotify profiles with their Facebook account, and, be able to find all their Facebook friends who are also on Spotify i.e. within the pub/sub system we mentioned before, these friends are referred to as "topics" that can be "subscribed" (followed). You can subscribe even without integrating your Facebook account by searching for one another, i.e. querying a database (Setty et. al., 2013, pp. 2). Similarly, publicly available user created playlists can be searched for. Users could also mark these as "collaborative" allowing others to edit, giving them writing permissions.

In the desktop version, we see a pane on the right side of the screen called "Friend Activity" that lists the songs they have been listening to.

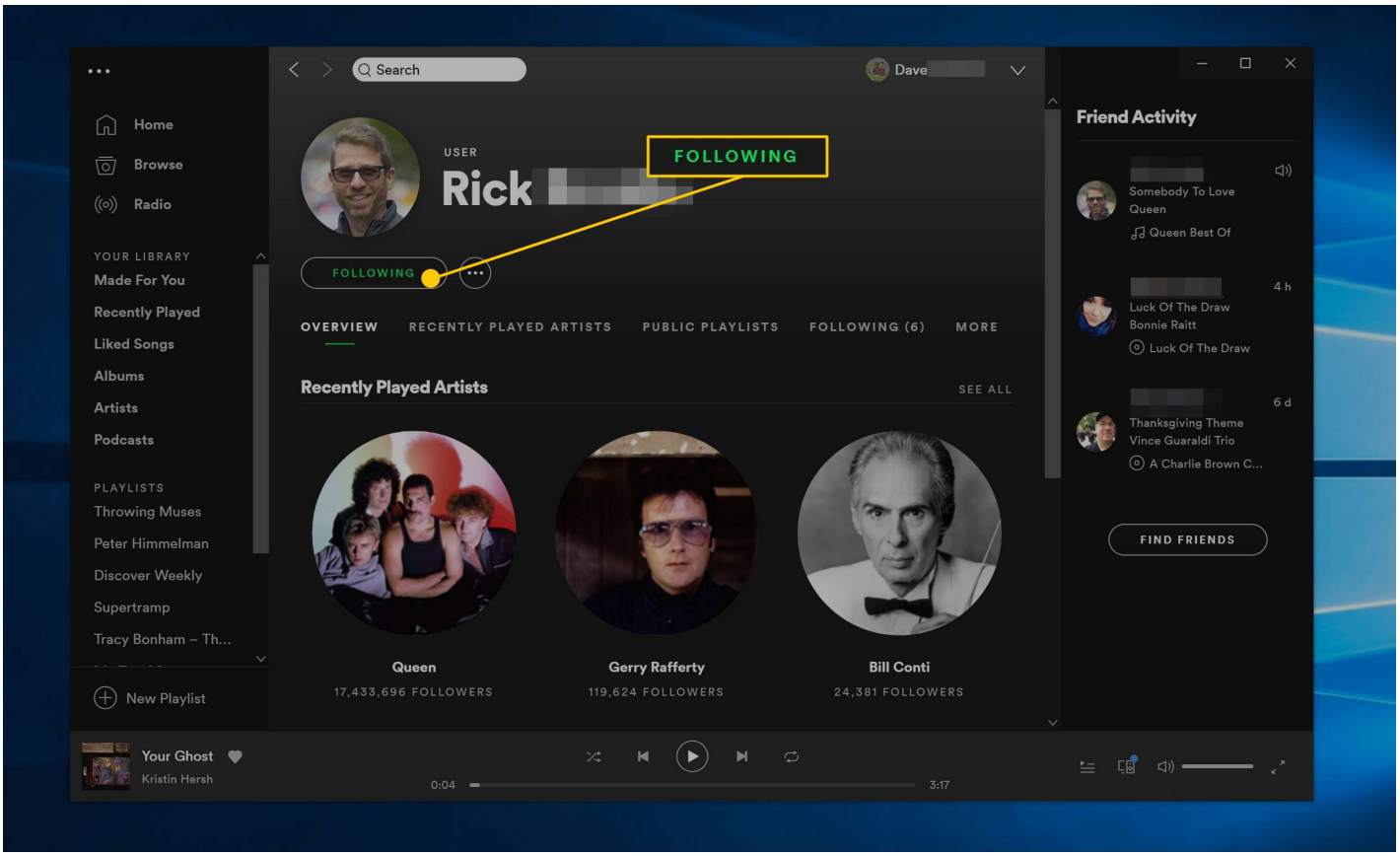


Fig: (Johnson, 2019)

This means that real-time user metadata is being shared across the platform using the pub/sub cloud systems through Apache Storm and Casandra that is then presented on the Graphical User Interface of computers. It might be difficult to understand why Spotify has not yet integrated this feature into its smartphone app. One possible reason why they have not launched this feature on smartphones might be because UX researchers and engineers are still in the testing phases to figure out the best affordances and user flows to optimise smartphone specific user experience with the “Friend Activity” feature while still maintaining Spotify as a primarily music and podcast streaming platform as opposed to a social media platform.

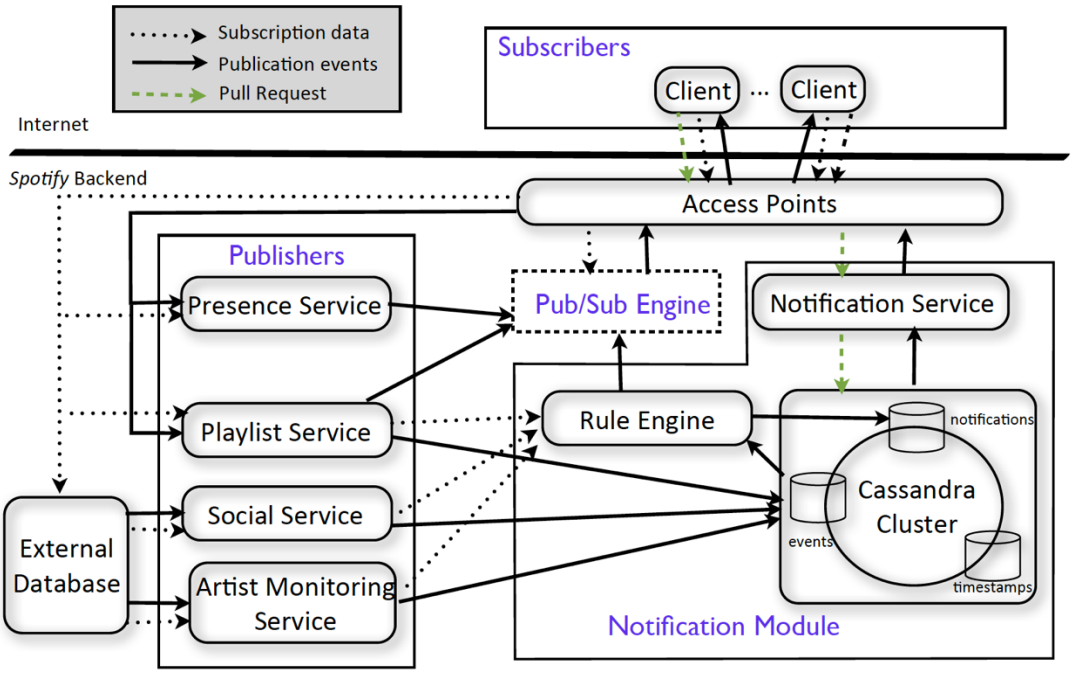


Fig: Architecture Supporting Social Interaction (Setty et. al., 2013, pp. 3)

The 'external database' mentioned in the figure is now part of Google's Cloud Storage. All of Spotify's backend services are operated via Google Cloud Services still using the Pub/Sub system.

In terms of database management and access, Spotify uses similar pub/sub systems as Facebook, Twitter and Google+ (Kermarrec and P. Triantafillou, 2013, pp. 16: 5). Yet Spotify is not a social media platform. Which means that it is only on the top level of the design architecture that it fashions itself as a cloud based music and podcast streaming service. At the bottom levels of the design architecture as well as the kind of user data it has, the way it manages it and the way it leverages the data, is it remarkably similar to how many social media companies design their systems.

Privacy

Spotify uses data to calculate royalties, run A/B tests, process payments, serve playlists and suggest new tracks to users. (Leenders, 2018) This data is protected by encrypting it with a single keychain. “Each user has their own set of keys that should be used for the encryption” which reduces the impact of any possible data leak since even hackers need decryption keys. Additionally it allows Spotify to control the lifecycle of data for individual users centrally.

Padlock is their key management service that manages keychains for all Spotify users. “This means, for example, every time a user looks at a playlist (even their own), the playlist service makes a call to Padlock to get the keychain of the playlist owner and then decrypts the playlist. Each service that calls Padlock gets its own set of keys” (Leenders, 2018). The keys have other applications as well. For example when a user opts out of targeted advertisement, access to this user’s personal data by the targeted advertisement can be blocked by removing the corresponding key so the advertiser can no longer identify the user as a target.

We can also deblackbox ‘friend activity’ and its family of features using our knowledge of Spotify’s Padlock system. The same encryption-decryption keys must be called upon when users follow each other or receive notifications about their friend’s activities. The user can opt out of displaying their data in ‘friend activity’ tabs any time by switching off ‘Listening activity’ which would block access to this user’s personal data related to the relevant ‘friend activity’ metadata by removing the corresponding key.

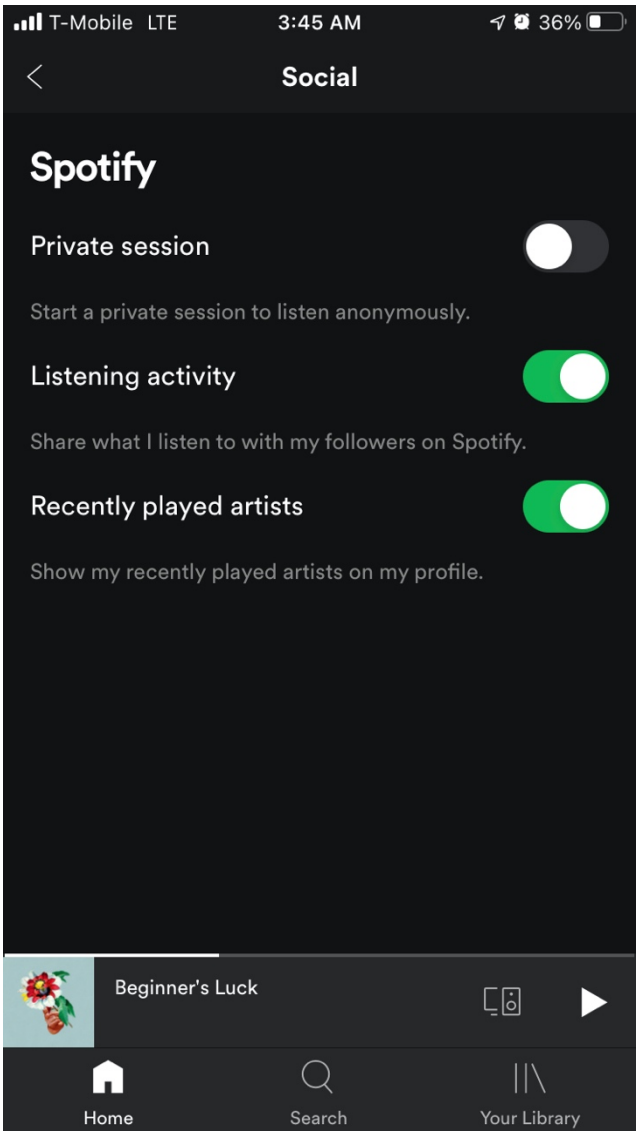
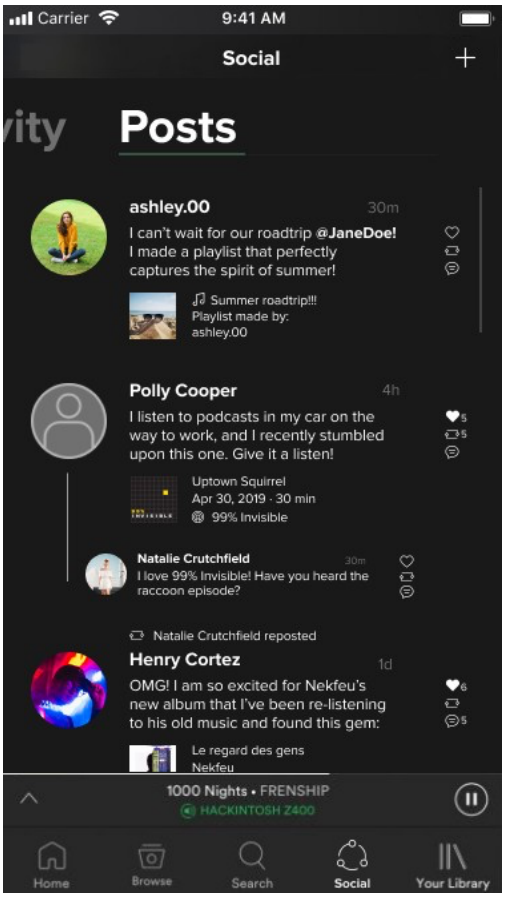


Fig: Social Privacy Settings

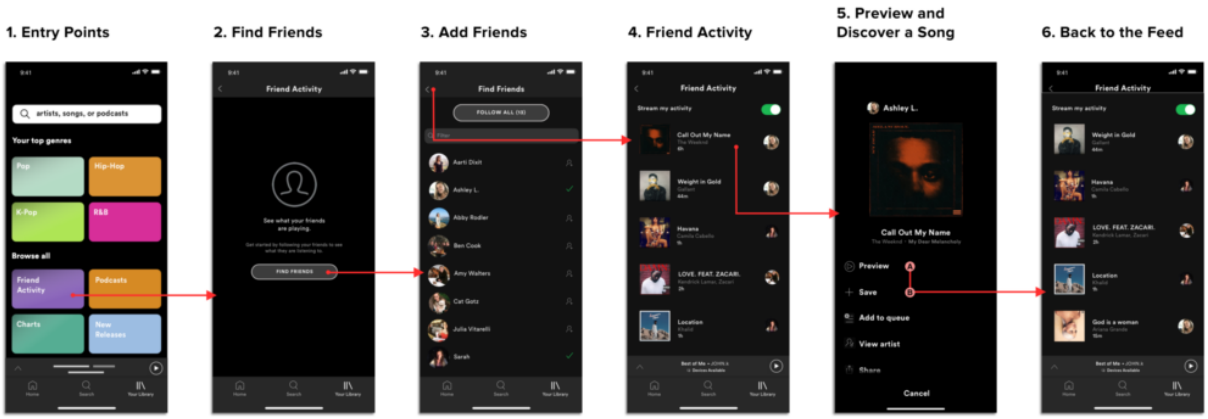
We can deblackbox ‘friend activity’ and its family of features using our knowledge of Spotify’s Padlock system. The same encryption-decryption keys must be called upon when users follow each other or receive notifications about their friend’s activities. The user can opt out of displaying their data in ‘friend activity’ tabs any time. All they have to is switch off ‘Listening activity’. This would mean that the corresponding key is removed, thus blocking access to this user’s metadata related to the relevant ‘friend activity’.

Social Savvy Affordances or the lack thereof

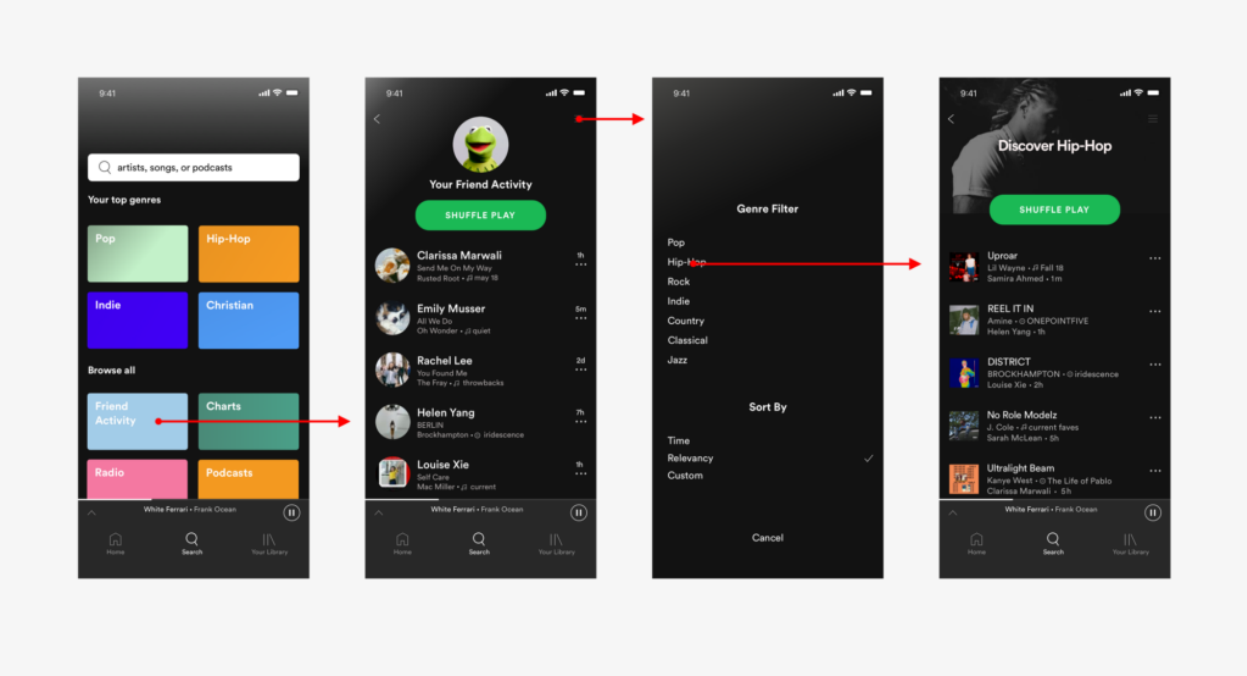
It seems that it is a strategic business/political decision then, that Spotify does not become a social media platform despite the kind of robust data infrastructure it has. With the heavy hitting that Big Tech is receiving in recent years, it seems reasonable to want to steer clear from the controversial limelight related to user data. Privacy, market power, free speech and censorship are key issues that plague social-media based platforms (Boskin, 2019). Spotify has avoided being politicized, while still proliferating its user database and metadata acquiring capacities. Despite a range of designs available out there on prototypr.io — from a 'strong' social features centric design (Jessica Man) to a 'weak' social features supplemented design (Cecilia Lu, Sanjana Seshadri), nothing even close has been integrated into Spotify.



(Fig: Man)



(Fig: Seshadri)



(Fig: Lu)

This means that in terms of UI design, Spotify may not be willing to create affordances that enable large scale social interactions internally within the app, steering clear of any Facebook wall-like features. Spotify may be trying to adhere to its privacy statements and avoid political controversies surrounding user data. Searching people on Spotify is not as straightforward as Facebook in terms of both accuracy as well as simplicity. You cannot comment, react or tag a song inside Spotify. The no of actions one can perform on a song one likes is limited. Sharing is not granted internally, nor are there many features to recommend others, notify or perform any social actions on these songs. This can partially be explained as a flexibility-usability tradeoff (Lidwell-Holden, pp. 86) and partially by Spotify's reluctance to be more social savvy, meaning the lack of affordances could actually be a strategic design constraint.

De-blackboxing the app reveals that though it has quite similar structures to social media apps, it is on the user-front that Spotify wishes to remain a streaming platform. It is here that we see how even technical decisions at the database structure level and UI design decisions depend on not just universal design principles for efficiency, but also socio-cultural landscapes.

Spotify allows "sharing" via third party apps like Facebook, Whatsapp, Tumblr, etc, using Web-based APIs but not internally with friends or followers. It does not access contacts on the mobile phone to connect to other possible Spotify users. Facebook data integration is the only way Spotify connects its users. It is understood that Spotify Wrapped was created to be shared on Instagram Stories. Meanwhile, Spotify acquires better AI systems to improve its recommendation service, continuing to focus on being streaming service platform (Novet, 2020). This again can be noticed in the proliferating affordances provided for personalized curated music such as discover weekly, made for you playlists, recommended radio playlists, etc.

Conclusion

Spotify derives its power from its database management systems. All the music and all the user data is stored on Google cloud using these systems. We saw how pub/sub system is utilised to operationalise streaming services on Spotify. We saw how various software libraries are used by Spotify to engage, extract and channel metadata generated by users as they interact on that app. The library softwares, cloud storage and the pub/sub system form the backbone on which Spotify functions. Accordingly, there are UI affordances and constraints that the app provides to the user that dictate how these systems will be used on the backend. And finally we saw how these UI

affordances and constraints are placed, depend not just on universal design principles but also on the overall strategy of the company i.e. just because Spotify has the user data, the systems and the capacity to provide certain services (social media related) does not mean that the top level design has to necessarily exploit bottom level design architecture just because the the bottom level design has the capacity for it.

References

Boskin, M. (2019, April 29). Big tech must get its house in order or risk stronger regulation | Michael Boskin. The Guardian. https://www.theguardian.com/business/2019/apr/29/big-tech-regulation-facebook-google-amazon

Greenberg, D., Kosinski, M., Stillwell, D., Monteiro, B., Levitin, D., & Rentfrow, P. (2016). The Song Is You: Preferences for Musical Attribute Dimensions Reflect Personality. Social Psychological and Personality Science, 7. https://doi.org/10.1177/1948550616641473

Jakobsen, A. Y. L. (2018). Eventization of listening: A qualitative study of the importance of events for users of the streaming service Spotify. 125.

Janota, B., & Stephenson, R. (2019, November 12). Spotify's Event Delivery – Life in the Cloud.

Spotify Engineering. https://engineering.atspotify.com/2019/11/12/spotifys-event-delivery-life-in-the-cloud/

Johnson, D. (2019). How to Find Friends on Spotify. Lifewire. https://www.lifewire.com/how-to-add-friends-on-spotify-4692334

Kermarrec, A.-M., & Triantafillou, P. (2013). XL peer-to-peer pub/sub systems. ACM Computing Surveys, 46(2), 16:1–16:45. https://doi.org/10.1145/2543581.2543583

Leenders, B. (2018, September 18). Scalable User Privacy. Spotify Engineering. https://engineering.atspotify.com/2018/09/18/scalable-user-privacy/

Lidwell, W., Holden, K., & Butler, J. (2010). Universal Principles of Design, Revised and Updated: 125 Ways to Enhance Usability, Influence Perception, Increase Appeal, Make Better Design Decisions. Rockport Publishers.

Lu, C. (2018, December 16). Spotify Mobile Case Study: Integrating Friend Activity. Medium. https://blog.prototypr.io/spotify-mobile-case-study-integrating-friend-activity-30de0cf12ee

Man, J. (2019, August 10). Re-imagining Spotify as a Social Media Platform. Medium. https://blog.prototypr.io/re-imagining-spotify-as-a-social-media-platform-2a646e60ab5f

Maravić, I. (2016, February 25). Spotify's Event Delivery – The Road to the Cloud (Part I). Spotify Engineering. https://engineering.atspotify.com/2016/02/25/spotifys-event-delivery-the-road-to-the-cloud-part-i/

Martin Irvine, The Internet: Design Principles and Extensible Futures

Mishra, & Brown. (2015, January 9). Personalization at Spotify using Cassandra. Spotify Engineering. https://engineering.atspotify.com/2015/01/09/personalization-at-spotify-using-cassandra/

Novet, J. (2017, May 18). Spotify just bought an AI startup to help it stay ahead of Apple Music. CNBC. https://www.cnbc.com/2017/05/18/spotify-buys-niland-french-ai-music-startup.html

Nudd, T. (2009). Spotify Crunches User Data in Fun Ways for This New Global Outdoor Ad Campaign. https://www.adweek.com/creativity/spotify-crunches-user-data-fun-ways-new-global-outdoor-ad-campaign-174826/

Reach for the Top: How Spotify Built Shortcuts in Just Six Months. (2020, April 15). Spotify Engineering. https://engineering.atspotify.com/2020/04/15/reach-for-the-top-how-spotify-built-shortcuts-in-just-six-months/

Ron White, "How the Internet Works." Excerpt from How Computers Work. 10th ed. Que Publishing, 2015.

Seshadri, S. (n.d.). Spotify. SANJANA SESHADRI. Retrieved December 8, 2020, from https://www.sanjanaseshadri.com/work/spotify-friend-activity

Setty, V., Kreitz, G., Vitenberg, R., van Steen, M., Urdaneta, G., & Gimåker, S. (2013). The hidden pub/sub of spotify: (Industry article). Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems – DEBS '13, 231. https://doi.org/10.1145/2488222.2488273

Spotify Case Study. (n.d.). Google Cloud. Retrieved December 9, 2020, from https://cloud.google.com/customers/spotify

Vesterlund, M. (2015, June 23). Switching user database on a running system. Spotify Engineering. https://engineering.atspotify.com/2015/06/23/user-database-switch/

What Is Pub/Sub? | Cloud Pub/Sub Documentation. (n.d.). Google Cloud. Retrieved December 7, 2020, from https://cloud.google.com/pubsub/docs/overview