Tales From the Past: Adapting App Repositories to App Store Dynamics

Michael Stach[1*], Marc Schickler[2], Manfred Reichert[2], Rüdiger Pryss[1]
[1]Institute of Clinical Epidemiology and Biometry, University of Würzburg, Würzburg, Germany
[2]Institute of Databases and Information Systems, Ulm University, Ulm, Germany
michael.stach@uni-wuerzburg.de, marc.schickler@uni-ulm.de, manfred.reichert@uni-ulm.de, ruediger.pryss@uni-wuerzburg.de

***Abstract*—The pervasiveness of smartphones in daily life has enabled the transformation of healthcare services into digital services, often offered as mobile apps to make them more accessible to the general public. The steady increase of apps in the app stores, in turn, led to the creation of app repositories to help end users, physicians, therapists, and healthcare providers find high-quality apps based on expert app ratings. However, the validity of these ratings is linked to a specific app version. In this paper, we aim to describe the problems of current app repositories and present an approach to address these shortcomings. More specifically, our approach is able to periodically monitor the two major proprietary app stores and react to upcoming changes. We furthermore present a mechanism to extract additional information from Android apps and to automate the latter procedure. Finally, our paper aims to stimulate discussion on what additional tools app researchers need to better study app quality and execution.**

***Index Terms*—mHealth, App Repository, Rating, App Store, Play Store, Android**

I. Introduction

The transformation of healthcare, and with it the provision and integration of new digital services, has advanced rapidly in recent years. In order to make health services accessible to the largest possible amount of the population, these services are being developed in particular as mobile applications (apps) for smartphones. Due to the high prevalence of smartphones and the integration into people's daily lives, apps are increasingly used to collecting data for research or patient monitoring, to providing educational content for patients, and have also been used for therapeutic applications.

The increasing amount of apps in the various app stores, however, complicates the selection and evaluation of adequate healthcare services for end users as well as physicians, therapists, and healthcare providers. To address this ever-increasing problem, a variety of review platforms for healthcare apps have emerged in recent years (e.g., [1], [2]). There also are efforts by large associations to develop their own standards for mobile health app repositories and mHealth data representations [3]. Most app repositories provide independent evaluations or ratings of the content quality of apps prepared by researchers and other experts. For the ratings, standardized instruments such as *ENLIGHT* [4] or *MARS* [5], which haven been translated in various languages [6] and adapted to variations for different target groups [7], are often used.

While these standardized instruments have proven themselves in the evaluation of apps, they are only valid for a specific version. Since software is an evolutionary construct, a rating is only meaningful for a certain period of time. Consequently, app repositories must take the life cycle of the rated apps into account in order to offer relevant recommendations over a longer period of time.

To illustrate this, we tested the availability of the publicly listed apps in our own app repository *Mobile Health App Database* (MHAD) [2]. To be more precise, we requested the app stores (i.e., either Google Play Store or Apple App Store) for 1083 apps that also possessed a store link and checked the returning HTTP status codes. We then excluded all apps without information about the time of the rating (n=325), to further check if recent ratings are also affected. The result showed that 41% of all apps were no longer available and thus neither the app rating could be verified nor other information about the app could be retrieved (see Figure 1). It also showed that this phenomenon does not only affect older ratings.

Looking at existing platforms and including the experience we made with MHAD [8]–[13], the current platforms are inadequately tailored to the highly volatile app market. To adequately support patients, researchers and healthcare providers, future platforms must offer solutions that address the following three shortcomings:

1) App store dynamics are not adequately reflected or processed and made available to researchers and raters. Changes in the app store (i.e., modification or deletion of apps and changes in app permissions or other metadata) must be monitored and properly reacted to.
2) The absence of historical app data disconnects the temporal relationship between app ratings and the app's lifecycle. The state of all data related to an app (i.e., metadata, App Store data, and installation package) should be archived and made accessible.
3) App metadata is not entirely made available through the app stores, resulting in incomplete or missing data. This data should be made accessible for researchers and future developments.

In this paper, we describe the current challenges of mobile health app platforms using the Google Play Store as an example and want to emphasize the necessity of app store

979-8-3503-2759-5/23/$31.00 ©2023 IEEE
DOI 10.1109/CSCE60160.2023.00225

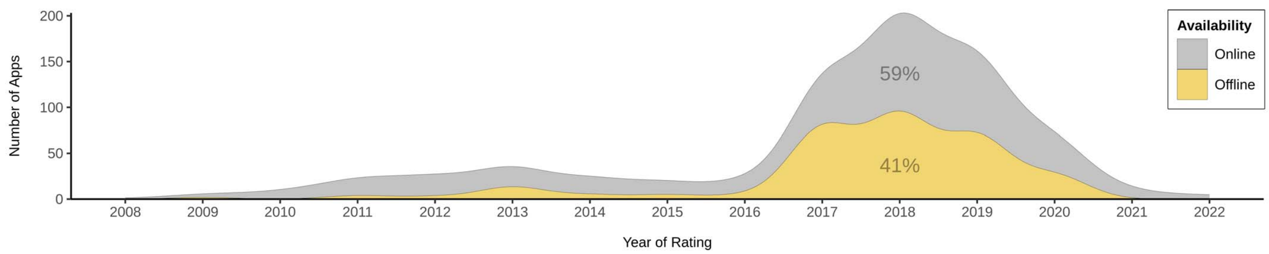

Fig. 1. Availability of apps with a valid link to either the Google Play Store or the Apple App Store in the Mobile Health App Database (n=758). The year refers to the time of the rating. Availability was checked using an HTTP request to the corresponding app store.

observation mechanisms to enable app repositories to react to changes in the app store. Furthermore, we present an approach to extract metadata from apps rather than from app stores and discuss the benefits of including such data in app repositories. Finally, we want to enhance the current state of app repositories with our contribution and start a discussion on future developments in mHealth app research.

II. Related Work

The evaluation of the quality of mHealth apps has been the content of various research papers for several years. Problems of the quality of mHealth apps have already been discussed in [14]. In terms of distribution, [14] and [15] describe the problem of large app stores allowing anyone to develop an app and listing it in the "medical" or "health and wellness" category. This implies two problems: On the one hand, the durability of the app is not necessarily given, and on the other hand, it requires expert reviews to find an effective health app in the multitude of apps. How such a review process can be designed was described in [16]. These methods were later challenged in [15], as the commonly used tools are said to be inadequate for assessing the quality of mHealth apps. In addition, the authors emphasized that systematic reviews can only be replicated for a short period of time, and this is, according to [15], also a problem with the current research approach.

Regarding the analysis of app store data, [17] describe the many methods used for feature analysis in their extensive survey. Among the methods used were the analysis of manifest files, which allowed app information such as permissions to be obtained. [18] also describes how the authors used Android applications to classify the latter based on the contents of their manifest file. An extensive overview and description of the contents of the manifest file is given in [19].

In addition, [20] compared the app permissions, which are also included in the manifest file, with the privacy policies of these apps. It was found that, especially in the Google Play Store, more permissions were requested than stated in the privacy policies. This was the case even though Google itself issues guidelines on the use of system permissions for its app store [21]. This worrying trend again shows the need for tools that can transparently show users and raters the actual permissions of an application.

In their attempt to measure the dynamics of the both large app stores, the authors of [22] also showed that privacy policies in particular are subject to frequent changes in the App Store. The authors of [23] attempted to quantify these frequent changes by calculating the half-life for apps. They also concluded that the mHealth environment is extremely volatile, and this fact, in turn, makes it difficult for consumers and clinicians to find suitable apps.

III. Technical Challenges

The reliable provision of app data is a challenging task. Especially in app research or for retrospective evaluation of app ratings, it is important that all necessary data is provided. If the app rating is up-to-date, plenty of information can be extracted from the app store. However, if this is no longer the case, it is difficult to obtain historical (meta) data for an app release. Metadata about apps are, for example, the version number of the app, app category, average star rating, app description, but also technical metadata about the app itself such as the Android version required to install the app, permissions or broadcast access. However, the technical metadata cannot be retrieved from the app stores without additional effort, since most proprietary services do not offer a publicly accessible interface.

Unlike the Apple App Store, the Google Play Store does not offer an official interface, so information must be extracted from the website through a process known as scraping. Scraping means that a program takes elements from a website (e.g., the textual app description, the app logo as an image, the app category or suggested similar apps) based on predefined rules. Since this process is extremely prone to errors, many technical challenges are related to the scraping mechanism.

As Apple's App Store offers several interfaces for reliably retrieving app information, the following sections will focus on the Google Play Store. The latter is known as a black box that is constantly changing [24]. A/B tests, for example, are held to improve the user experience, which can lead to different search results: Apps are displayed in a different order or the results are limited to 30 apps that Google has selected to show when searching for specific terms. These constant changes complicate the extraction of information, as the extraction rules have to be constantly adapted and the program code changed. This issue, in turn, complicates the provision of a reliable and available search feature, since the execution of new queries is not possible during the downtime. Furthermore, too many requests to the store websites can lead to an IP-based ban. These limitations of automated accessibility require the creation of an in-between solution to ensure accessibility to information without depending on the availability in the app stores.

Another problem is becoming visible due to current changes to the Google Play Store: Metadata is only displayed in a limited way (i.e., shortened or potentially not at all). Thus, any scraper-based solutions will only be able to offer data of poorer quality. Such a change was implemented in 2022, when Google included its so-called Data Safety feature [25], which on the one hand required developers to provide a summary of the collected and shared data, but on the other hand did not verify this information and delegate the responsibility for validity to the developers [26]. Shortly thereafter, Google removed the self-generated app permission information from the Play Store, so that no reliable information about interface access was available anymore. Though Google rolled back this change due to public pressure not long after, this, however, again illustrates that changes to the app stores can limit data availability at any time.

Interestingly, the availability of app permission metadata is an important issue in that the Google Play Store, for example, replaces the actual permission identifiers with more easily accessible descriptions. Thus, providing app repositories with the complete list requires a new different approach to obtain these data. To fill this technological gap, we propose to use the Android Package file (APK)[1] of an app to retrieve the required permissions in the technical description documents of an app. To be more precise, in the case of Android, the information is stored in the so-called *AndroidManifest* XML file. This file contains, besides the required permissions, other information that is of interest to app research: Since many studies in the field of mHealth are designed with the ”bring you own device” approach, it must be ensured that applications also work on different devices. Therefore, app repositories should offer the possibility to show the users only apps for their devices. The information needed to accomplish this can also be found in the *AndroidManifest*. Furthermore, the *AndroidManifest* also includes information about retrieving shared device information (i.e., broadcasts), for example, to react to device reboots.

However, both app stores do not offer the option of downloading an app directly from the website. For Apple, this is understandable, as it currently does not allow sideloading (i.e., installing apps outside the official app store) [27]. Android, however, has already supported sideloading for years and only allows downloading apps via the store app on the smartphone, or via Google's interfaces that require authentication. In order to provide app repositories with additional information, this interface must be made accessible programmatically and expanded in a scaling manner.

[1]APK is the file format that Android uses to distribute and install apps.

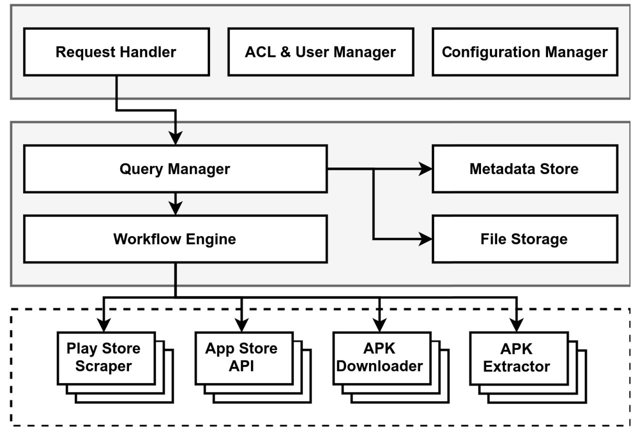

Fig. 2. Conceptual overview of the proposed approach to combine app store metadata with app metadata extracted from the apps' APK file.

IV. CONCEPT

To address the described challenges or, to be more precise, to track changes in the app stores and align to the app lifecycle, we combined the rich data retrieved with the scraper with a novel approach to additionally store app packages as well as their extracted metadata (see Figure 2). To accomplish this, we have decomposed the procedure into individual units of work. This increases interchangeability in case of app store changes and simplifies scaling. Furthermore, decomposition can be used to prevent IP-based blocking by allowing requests to be sent from different servers. This technique can also be used to bypass country-specific app store restrictions. A trade-off of decomposition is the additional effort that must be spent on orchestration. Since the individual steps can take several seconds, the communication of the *Request Handler* with the requesting parties is implemented asynchronously. This adds additional overhead to manage the status of a request. Our concept proposes the use of a *Workflow Engine* to take care of these tasks. This technology also provides additional quality assurance mechanisms such as retry on failure, exception handling, and state monitoring with an audit function. We have chosen *Zeebe* as the *Workflow Engine*, but these tasks can also be handled by other workflow engines or technologies. Furthermore, the modeled workflows can be easily extended, for example, to notify applications, researchers, or evaluators when changes occur.

App store data (i.e., app descriptions, user star ratings, user reviews or app version) is collected either using the Play Store scraper or App Store API worker. To be more precise, we reused the same tools that were developed for the MHAD platform, as they have proven themselves in the field. Furthermore, additional configurations are possible via the scraper, for example, to set the country of the store and the language.

We extract information from the *AndroidManifest* to obtain additional app data. However, the Google Play Store does not provide a public interface to retrieve this information. Since

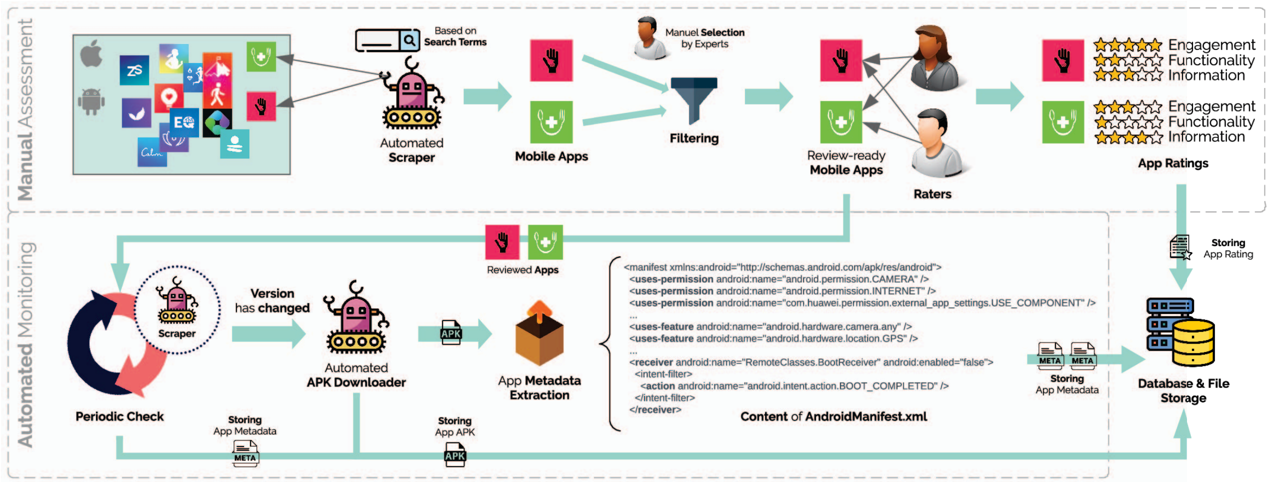

Fig. 3. The proposed adaptations can be easily integrated into existing procedures (e.g., assessment process of the Mobile Health App Database [2]).

the manual workaround via a smartphone does not scale, we developed an automatable *APK Downloader* based on the Java application *Raccoon*. To do this, the *APK Downloader* requires a Google account to request the non-public interfaces. Comparable to the process that runs on the Android smartphone, the *APK Downloader* associates a custom device configuration with the user account to download apps through the official interface. The advantage of this approach is that you can create different profiles and thus simulate (test) devices. Apps that are not supported by this device (e.g., because the Android version is too old) can consequently not be downloaded. The *Workflow Engine* can subsequently react to this error via exception handling.

Then, in a second step, the downloaded APK is extracted and the *AndroidManifest* is parsed. An excerpt of the contents of an *AndroidManifest* is shown in Figure 3. It contains, for example, information about the requested permissions, the used features of the smartphone or which broadcasts the app wants to receive. Since the *AndroidManifest* is not stored in readable form in the APK file, we use a tool called *Android Asset Packaging Tool* (AAPT)[2], which is developed and published by Google, to output the *AndroidManifest* contents. Then the content is parsed, stored in a structured format, and given a correlation identifier to associate it with the APK file.

V. Discussion

With this paper, we would like to not only present our proposed approach to extending the current state of app repositories, but rather to begin a discussion of the problems of current approaches and the opportunities created by the extensions presented. Therefore, we presented an approach that describes how app search can be accomplished via scrapers and extended using metadata about the app itself. We used the *AndroidManifest* for the latter purpose, as it is currently the only reliable source of technical metadata. In the future, this information will be of increasing interest, since, for example, the display behavior of notifications is controlled via a permission request starting with Android 13 [28]. Future systems can use this information to better assess whether apps work reliably on users' end devices and what device features the apps use. Due to the many different Android distributions available, inconsistencies in the execution of apps will occur more frequently in the future and tools for verification - especially with regard to mHealth or apps that are used as scientific instruments - will become more and more important [29]. Apparently, the quality assurance guidelines published by Google [30] are not sufficient to ensure proper execution on different devices [31]. Moreover, the extracted information of the *AndroidManifest* can be used in the future for more sophisticated tools like an automatic reliability assessment to detect possible incompatibilities (i.e., simulate the execution on a specific device), if this information is made available.

As already described, another problem of current approaches is the lack of tracking of changes in the app stores. Our approach is designed as a workflow-based approach and thus allows us to schedule and execute processes multiple times. By repeatedly requesting the app stores, we can detect changes and thereby generate higher-value insights such as update frequency or correlation of download trends and ratings. Tracking changes also enables the creation of a historical dataset. Due to the lack of such an archive in the app stores, only the current state is available at any given time. Since apps are an evolutionary construct and are always changing, ratings are only valid for a certain release. In order to be able to retrospective evaluate these ratings, a historical data set that includes the date of the rating is needed. This is of scientific interest in that ratings are also often published in research papers and are difficult to verify without a historical data set [15].

In addition, we have designed our approach to be open so

[2]https://developer.android.com/tools/aapt2

that it can be easily integrated into existing rating processes. We have demonstrated an example integration of the approach into the MHAD rating process in Figure 3. Raters can, for example, send a list of rated apps to the system for monitoring and/or storing additional data. The latter will start a process for all apps in the list and query the app stores periodically. In case of a version change, all changes are saved and the app is reported to the *APK downloader*. The meta information is then unpacked and can be stored along with the app ratings. Even though this example simulates the best case, app store monitoring needs an approach that is robust to failure cases and flexible for extensions. For this reason, we decided to use a workflow engine, since we can easily configure both error handling and workflow extensions with the bundled modeler.

A. Future Work

Our concept and implemented prototype constitutes the basis of further work. On the one hand, based on MHAD [2], it would provide the basis for app repositories that not only offer a list of ratings, but can provide additional information and insights through stronger integration with app stores. Such a centralized platform could also be used to unify ratings from different tools and languages (e.g., ENLIGHT [4], MARS [5]–[7] or AQUA [32]). In the area of apps in health research, papers and projects could also be found automatically, for example, by searching and linking to interfaces of literature management services such as *SemanticScholar*[3]. Furthermore, integrating results from research projects on app compatibility [31] would help to make a compatibility statement for execution on proposed devices. This could also increase the acceptance of such services by the general public if users could directly check whether this app works as expected on their end device. In addition to the in-depth expert evaluation, it would be possible to integrate tools for evaluating apps that are tailored to end users in particular. Ultimately, such an information hub for healthcare apps could enrich research by gathering information to feed future AI-based systems to discover new ways to find high-quality apps based on their app store data. The latter is a major problem [33] [34] and could be improved by automated evaluations.

B. Limitations

Even though the approach presented in this paper can obtain data from the Apple App Store, an extraction through metadata similar to the *AndroidManifest* on Android is currently not possible. Furthermore, the approach to downloading Android apps relies on private Google accounts and can potentially be hampered by further security measures. In particular, if Google would require a mandatory connection to a real device. Also, Google requires a hardware configuration to download apps, which is irretrievably linked to the Google account. The compilation of appropriate configurations is complex, which is why the selection of hardware configurations is currently still limited. In addition, a legal classification of the data use is difficult, since the interpretation of a fair use of the data for research varies.

[3] https://www.semanticscholar.org/

VI. Conclusion

In this paper, we have described the shortcomings of current app repositories that we have identified by looking at existing platforms as well as through the experience of our own app repository.

On the one hand, current approaches are not sufficiently adapted to the lifecycle of apps. This means that changes in the app store (e.g., updating or deleting an app) are not adequately mapped. Furthermore, no historical data is provided by the proprietary app stores, which is why a retrospective review of ratings is not possible. In addition, not all app metadata is made available through the app stores, so approaches are needed to extract it.

Our presented approach aims to address these shortcomings by enabling periodic monitoring of the app store. We also presented an automated method for extracting app metadata from the app itself. The extracted information can be used to train future system to identify effective mHealth apps based on app data in the app store. Finally, this paper aims to stimulate discussion on whether app research needs new tools and whether relying on data from the major app store operators is future-proof.

References

[1] S. Lagan, P. Aquino, M. R. Emerson, K. Fortuna, R. Walker, and J. Torous, "Actionable health app evaluation: translating expert frameworks into objective metrics," *npj Digital Medicine*, vol. 3, no. 1, p. 100, Jul. 2020. [Online]. Available: https://doi.org/10.1038/s41746-020-00312-4

[2] M. Stach, R. Kraft, T. Probst, E.-M. Messner, Y. Terhorst, H. Baumeister, M. Schickler, M. Reichert, L. B. Sander, and R. Pryss, "Mobile Health App Database - A Repository for Quality Ratings of mHealth Apps," in *33rd IEEE CBMS International Symposium on Computer-Based Medical Systems (CBMS)*. Rochester, MN, USA: IEEE, Jul. 2020, pp. 427–432. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9183243/

[3] "IEEE standard for open mobile health data—representation of metadata, sleep, and physical activity measures," *IEEE Std 1752.1-2021*, pp. 1–24, Sep. 2021.

[4] Baumel, Amit and others, "Enlight: A Comprehensive Quality and Therapeutic Potential Evaluation Tool for Mobile and Web-Based eHealth Interventions," *J Med Internet Res*, vol. 19, no. 3, p. e82, Mar. 2017. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/28325712

[5] S. Stoyanov and others, "Mobile App Rating Scale: A New Tool for Assessing the Quality of Health Mobile Apps," *JMIR mHealth and uHealth*, vol. 3, no. 1, p. e27, 2015, 00755.

[6] K. Yamamoto, M. Ito, M. Sakata, S. Koizumi, M. Hashisako, M. Sato, S. R. Stoyanov, and T. A. Furukawa, "Japanese Version of the Mobile App Rating Scale (MARS): Development and Validation," *JMIR Mhealth Uhealth*, vol. 10, no. 4, p. e33725, Apr. 2022. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/35197241

[7] S. R. Stoyanov, L. Hides, D. J. Kavanagh, and H. Wilson, "Development and Validation of the User Version of the Mobile Application Rating Scale (uMARS)," *JMIR Mhealth Uhealth*, vol. 4, no. 2, p. e72, Jun. 2016. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/27287964

[8] A. A. Portenhauser, Y. Terhorst, D. Schultchen, L. B. Sander, M. D. Denkinger, M. Stach, N. Waldherr, D. Dallmeier, H. Baumeister, and E.-M. Messner, "Mobile Apps for Older Adults: Systematic Search and Evaluation Within Online Stores," *JMIR Aging*, vol. 4, no. 1, p. e23313, Feb. 2021. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/33605884

[9] M. Muehlmann, A. Lienert, H. Muehlan, M. Stach, Y. Terhorst, and E.-M. Messner, “Digitale Sexualaufklärung: Verfügbarkeit und Evaluation mobiler deutschsprachiger Apps zur Förderung der sexuellen Gesundheit,” *Z Sex Forsch*, vol. 34, no. 04, pp. 197–207, Jul. 2021. [Online]. Available: https://doi.org/10.1055/a-1669-7626

[10] D. Schultchen, Y. Terhorst, T. Holderied, M. Stach, E.-M. Messner, H. Baumeister, and L. B. Sander, “Stay Present with Your Phone: A Systematic Review and Standardized Rating of Mindfulness Apps in European App Stores,” *International Journal of Behavioral Medicine*, vol. 28, no. 5, pp. 552–560, Oct. 2021. [Online]. Available: https://doi.org/10.1007/s12529-020-09944-y

[11] L. S. Steubl, J. Reimann, L. Simon, Y. Terhorst, M. Stach, H. Baumeister, L. B. Sander, and E.-M. Messner, “A systematic quality rating of available mobile health apps for borderline personality disorder,” *Borderline Personality Disorder and Emotion Dysregulation*, vol. 9, no. 1, p. 17, Jun. 2022. [Online]. Available: https://doi.org/10.1186/s40479-022-00186-w

[12] E.-M. Messner, N. Sturm, Y. Terhorst, L. B. Sander, D. Schultchen, A. Portenhauser, S. Schmidbaur, M. Stach, J. Klaus, H. Baumeister, and B. M. Walter, “Mobile Apps for the Management of Gastrointestinal Diseases: Systematic Search and Evaluation Within App Stores,” *J Med Internet Res*, vol. 24, no. 10, p. e37497, Oct. 2022. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/36197717

[13] L. Simon, J. Reimann, L. S. Steubl, M. Stach, K. Spiegelhalder, L. B. Sander, H. Baumeister, E.-M. Messner, and Y. Terhorst, “Help for insomnia from the app store? A standardized rating of mobile health applications claiming to target insomnia,” *Journal of Sleep Research*, vol. 32, no. 1, p. e13642, Feb. 2023. [Online]. Available: https://doi.org/10.1111/jsr.13642

[14] Q. Grundy, “A review of the quality and impact of mobile health apps,” *Annual Review of Public Health*, vol. 43, pp. 117–134, 2022, publisher: Annual Reviews.

[15] Q. H. Grundy, Z. Wang, and L. A. Bero, “Challenges in assessing mobile health app quality: a systematic review of prevalent and innovative methods,” *American journal of preventive medicine*, vol. 51, no. 6, pp. 1051–1059, 2016, publisher: Elsevier.

[16] E. D. Boudreaux, M. E. Waring, R. B. Hayes, R. S. Sadasivam, S. Mullen, and S. Pagoto, “Evaluating and selecting mobile health apps: strategies for healthcare providers and healthcare organizations,” *Translational behavioral medicine*, vol. 4, no. 4, pp. 363–371, 2014, publisher: Oxford University Press.

[17] W. Martin, F. Sarro, Y. Jia, Y. Zhang, and M. Harman, “A survey of app store analysis for software engineering,” *IEEE transactions on software engineering*, vol. 43, no. 9, pp. 817–847, 2016, publisher: IEEE.

[18] A. Shabtai, Y. Fledel, and Y. Elovici, “Automated static code analysis for classifying android applications using machine learning,” in *2010 international conference on computational intelligence and security*. IEEE, 2010, pp. 329–333.

[19] A. K. Jha, S. Lee, and W. J. Lee, “Developer mistakes in writing android manifests: An empirical study of configuration errors,” in *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*. IEEE, 2017, pp. 25–36.

[20] B. Brumen, A. Zajc, and L. Bošnjak, “Permissions vs. Privacy Policies of Apps in Google Play Store and Apple App Store,” in *Information Modelling and Knowledge Bases XXXIV*. IOS Press, 2023, pp. 258–275.

[21] Google, “Permissions and APIs that Access Sensitive Information - Play Console Help,” May 2023. [Online]. Available: https://support.google.com/googleplay/android-developer/answer/9888170?visit_id=638211237760159048-3750078916&rd=1

[22] S. E. Stoeckl, E. Torres-Hernandez, E. Camacho, and J. Torous, “Assessing the Dynamics of the Mental Health Apple and Android App Marketplaces,” *Journal of Technology in Behavioral Science*, Jan. 2023. [Online]. Available: https://doi.org/10.1007/s41347-023-00300-x

[23] M. E. Larsen, J. Nicholas, and H. Christensen, “Quantifying App Store Dynamics: Longitudinal Tracking of Mental Health Apps,” *JMIR Mhealth Uhealth*, vol. 4, no. 3, p. e96, Aug. 2016. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/27507641

[24] T. Poell, D. Nieborg, and J. Van Dijck, “Platformisation,” *Internet Policy Review*, vol. 8, no. 4, pp. 1–13, 2019, publisher: Berlin: Alexander von Humboldt Institute for Internet and Society.

[25] S. Frey, “Get more information about your apps in Google Play,” Apr. 2022. [Online]. Available: https://blog.google/products/google-play/data-safety/

[26] Google, “Provide information for Google Play’s Data safety section - Play Console Help,” 2022. [Online]. Available: https://support.google.com/googleplay/android-developer/answer/10787469

[27] M. Gurman, “Will Apple Allow Users to Install Third-Party App Stores, Sideload in Europe? - Bloomberg,” Dec. 2022. [Online]. Available: https://www.bloomberg.com/news/articles/2022-12-13/will-apple-allow-users-to-install-third-party-app-stores-sideload-in-europe

[28] Android Developers, “Behavior changes: Apps targeting Android 13 or higher | Android Developers,” May 2023. [Online]. Available: https://developer.android.com/about/versions/13/behavior-changes-13

[29] A. González-Pérez, M. Matey-Sanz, C. Granell, and S. Casteleyn, “Using mobile devices as scientific measurement instruments: Reliable android task scheduling,” *Pervasive and Mobile Computing*, vol. 81, p. 101550, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574119222000074

[30] Android Developers, “Core app quality | App quality | Android Developers,” May 2023. [Online]. Available: https://developer.android.com/docs/quality-guidelines/core-app-quality

[31] M. Stach, F. Pflüger, M. Reichert, and R. Pryss, “LAMP: a monitoring framework for mHealth application research,” *Procedia Computer Science*, vol. 198, pp. 203–210, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050921024686

[32] T. O’Rourke, R. Pryss, W. Schlee, T. Probst, and others, “Development of a multidimensional app-quality assessment tool for health-related apps (AQUA),” *Digital Psychology*, vol. 1, no. 2, pp. 13–23, 2020.

[33] H. Wisniewski, G. Liu, P. Henson, A. Vaidyam, N. K. Hajratalli, J.-P. Onnela, and J. Torous, “Understanding the quality, effectiveness and attributes of top-rated smartphone health apps,” *Evidence-based mental health*, vol. 22, no. 1, pp. 4–9, 2019, 00014.

[34] M. E. Larsen, K. Huckvale, J. Nicholas, J. Torous, L. Birrell, E. Li, and B. Reda, “Using science to sell apps: evaluation of mental health app store quality claims,” *NPJ digital medicine*, vol. 2, no. 1, pp. 1–6, 2019, 00043.