Bachelorarbeit am Institut für Informatik der Freien Universität Berlin

Human-Centered Computing (HCC)

Privacy Visualizations: Introducing an interactive visualization of privacy indicators based on Exodus Privacy to F-Droid

Antonios Hazim

Betreuer: Peter Sörries

Erstgutachterin: Prof. Dr. Claudia Müller-Birn

Zweitgutachter: Prof. Dr.-Ing. Volker Roth

Berlin, 12.01.2023

Abstract

Context	Privacy
Subject area	Interactive Privacy Visualizations
Domain or application area	Application Stores
Field of Research	HCI

F-Droid [4] is a software store for open source applications for Android and is considered the most important distribution channel for free open source software for Android. This work attempts to tackle two problems of the *F-Droid* app store in terms of visualizing privacy-related information of the apps in a human-centered design fashion. The first problem is that *F-Droid*'s privacy metadata is limited to *Anti-Features* [1], which lack depth and use technical terms that are of no use to the majority of users. The second problem is to present the privacy parameters of these apps in an informative and interactive way, moving away from the current less user-friendly text formats.

To this end, *Exodus Privacy* [3], an extensive community-driven open database of privacy-related data (e.g., trackers) of Android apps, was used to complement *F-Droid*'s data. Building on existing research in the privacy visualization field (e.g., "*DCI*" [44], CLEVER°Franke's *Privacy Label* [21], or Stöver et al.'s research on privacy indicators [41]), several privacy visualization prototypes were developed. Subsequently, the prototypes were discussed in expert interviews, and the preferred prototype was then refined and implemented in *Neo Store* [31] , a modern *F-Droid* client. The study was wrapped up with a community survey to evaluate the developed design and identify potential future work.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Context & Motivation

As smartphones became an indispensable part of everyone's life, centralized data collection and tracking evolved into a highly profitable business. While the use of electronic devices increased exponentially and with it the interest in human-computer interaction (HCI), awareness of such business models and, more broadly, *digital literacy*, i.e. the ability to navigate, evaluate and communicate with digital platforms, still lagged behind. As a result, regulators have been forced to adopt regulatory measures (e.g., GDPR [5]) to curb this trend. These regulations, criticized by digital rights activists as insufficient to protect privacy because they leave room for deceptive designs with ever-changing dark patterns[1] [24], have also failed to make a clear contribution to users' digital literacy [16, 46, 45].

As a result, new approaches have been developed that focus exclusively on making data collection practices and privacy violations, such as the misuse of user data by third parties, more transparent for users. One of these areas is the field of *privacy visualizations*, which mainly addresses two questions:

- What privacy-related information should be provided to the user?
- What is the effective representation of privacy-related information?

App stores (e.g., Apple's AppStore and Google's Play Store) became the main source of the software most people use on their devices, whether on a desktop or a smartphone, which makes them an ideal starting point to optimize privacy visualization. However, this has only recently become the case, as Apple's "Privacy Labels" [11] and Google's "Data Safety Secion" [23] can be seen as the first steps in enforcing more transparency in the data collected by the applications distributed on their respective platforms, albeit not wholeheartedly, as they clearly continue to prioritize their business model interests (which is a matter of course for profit-oriented companies). Google and Apple's efforts were preceded by some initiatives such as *Exodus Privacy*, which has created an extensive database of privacy-related metadata on Android applications. Such *Open-Knowledge*[2] privacy approaches have been welcomed and

[1]Aka deceptive user interfaces, are design patterns that aim to manipulate users into taking actions that are beneficial to the service provider, or to prevent them from taking actions that are not beneficial to the service provider, contrary to their own interests. This is mainly related to the different incentives of companies and users [37, 17].

[2]Open-Knowledge is a term defined by the Open Knowledge Foundation that builds on the principle of open data and opens up applications in various fields, e.g. open-government, open-source [9].

adopted by many privacy advocates (e.g., iRights e.V.'s Mobilsicher project in Germany [2]), but have not evolved beyond a data source from which users can manually retrieve information, which is overwhelming for most users anyway. In the realm of application stores, the open-knowledge approach has also established itself as a direct alternative to the aforementioned commercial stores, namely with the open-source app store *F-Droid* [4].

1.1.1 Motivation

F-Droid alone does not provide enough data to create useful privacy visualizations, so the *F-Droid* metadata on apps must be supplemented by other sources. It's also to be noted that the fact that such a comprehensive source of privacy-related information as *Exodus Privacy* exists and is not being used to directly empower users is clearly a missed opportunity. At the same time, providing such information in direct textual form may not provide much benefit to most users or help raise their privacy awareness. Therefore, privacy-related information is presented based on the relevant design approaches that have been explored by researchers focusing on privacy visualizations (cf. section 2.1).

However, while institutions in the data collection industry (e.g., Google) have fortunately been unable to break into the *FOSS* space, few efforts have been made in the *FOSS* community to raise user awareness of privacy issues or to improve the usability of the tools already available. As privacy has essentially evolved in recent decades from an individual choice to a dense web of actors and their goals, it is becoming increasingly clear that empowering each individual user should be the main concern in improving people's privacy [13].

The main motivation behind this work is to address one of the main problems of the *F-Droid* app store system in terms of privacy-related information, namely the fact that privacy-related information is not communicated to users in a well-understood form and thus the user empowerment aspect is absent. This work goes a step further and provides privacy-related information as an interactive privacy visualization design that aims to improve comprehensibility for all users and provide them with control options. The developed design is then implemented in *Neo Store*, a modern *F-Droid* client I have developed.

1.1.2 Related Terminology

To better understand the context, it is helpful to know some of the key terms. These can be divided into community-specific and privacy-related terms. These terms are explained below, starting with terms primarily used by the open source community and then moving on to the key terms used in the research field of privacy.

The *Free Open Source Software (FOSS)* community is a relatively well-informed community around open source projects that advocates for general digital rights and strongly emphasizes digital freedom rights. The *FOSS* community is perceived by the public as a niche community with only the technically savvy. As a long-time member of the Android *FOSS* community, I am aware of the real and mostly obvious knowledge gap that exists between the few well-informed *FOSS* advocates and the many uninformed users.

F-Droid [4] is a store software for Android open-source applications (including backend and frontend) and is considered the main distribution mean for free open-source software (*"FOSS"*) for Android devices. *F-Droid*'s repository structure also enables easy hosting of decentralized repositories.

Since *F-Droid* was developed with a privacy-conscious mentality, *Anti-Feature* [1] labels were developed very early on, which are categorical labels describing possible undesirable app characteristics for users. The current eleven *Anti-Features* of *F-Droid* mainly refer to *"non-free"* attributes of the software, such as the inclusion of trackers, the presence of advertising, or the use of a *"non-free"* services [3].

While the above *Anti-Feature* labels may be of interest to savvy "Techies" and some *FOSS* advocates, they are useless to ordinary users since they utilize technical jargon that is incomprehensible to most users. Therefore, this work attempts to provide an informative representation of privacy-related information and translate these technical terms into understandable forms. This attempt would benefit the majority of users and raise awareness of privacy-related practices of apps.

Users in this context, contrary to stereotypical perceptions, consist of a minority of *FOSS* advocates or "power users" who are already strongly privacy-conscious, and an overwhelming majority of average users who use the various *F-Droid* repositories as a complementary means of installing and updating Android apps. For example, users who install the premium versions of Simple Mobile Tools [39] free of charge (as opposed to the Google Play Store).

In order to implement an informative presentation of privacy-related information, an additional source of information is needed, since *F-Droid* does not provide enough data itself, and preferably one that follows the principles of open knowledge, i.e. is publicly available or provided under an open license [9]. This is where *Exodus Privacy* [3] comes in, *Exodus Privacy* is a popular community-driven and open database that contains, among other things, a list of trackers in Android apps.

Privacy visualizations being the user-facing side of the coin; the developer-facing side is the *Privacy by Design* [4] guidelines, which dictate how developers should deal with privacy of users and the extent to which it should be taken into account in product design [12]. Yet no universally accepted standard for either *privacy visualizations* or *Privacy by Design* does exist, resulting in developers being confronted with divergent and conflicting situations regarding policies, while users being overwhelmed, for example, by the different privacy

[3] *"free"* is here the discursive term in the open source context or, in other words, free as in "free speech, not free beer" [40], as commonly noted by open source advocates.

[4] "This is an umbrella term for software development approaches that consider privacy issues in the early stages of design." [12].

visualization approaches. As a result, the challenge remains to enable user awareness through privacy-related information [12].

Privacy labels and *privacy indicators* are terms related to the implementation of privacy visualizations, but do not yet have a unified definition. In this work, the two terms are based on specific use cases from the literature and research in this area. The term *privacy labels* is based on the visual format of CLEVER°FRANKE [21], which, unlike *EU's nutrition labels*, is an assessment-oriented visualization and Fox et al.'s [20] *GDPR* label, which provides for a combination of icons and simple descriptions, *privacy label* thus covers both visual and descriptive visualization formats.

On the other hand, *privacy indicators* is aligned here with *Data Controller Indicators (DCI)*, i.e., the use of visualizations that indicate the presence or absence of certain attributes [44].Although not yet universally defined, these and other formats have brought some structure to the confusing field of privacy visualizations.

1.2 Research Goals & Questions

Given the status quo described above and the fact that I am involved in the development of *Neo Store*, an *F-Droid* client, I was motivated to develop and implement an interactive privacy visualization design that combines data from the two open knowledge systems and raises users' privacy awareness. The main goal of this work is to review privacy visualization research and work with the community on developing an interactive privacy visualization design.

Three steps were formulated to achieve this goal:

- Identify the data needed to provide comprehensive privacy information to users.
- Identify the most promising efforts in relevant research on privacy visualization in terms of interactive design for apps in app stores.
- Implement an interactive privacy visualization design in collaboration with the community.

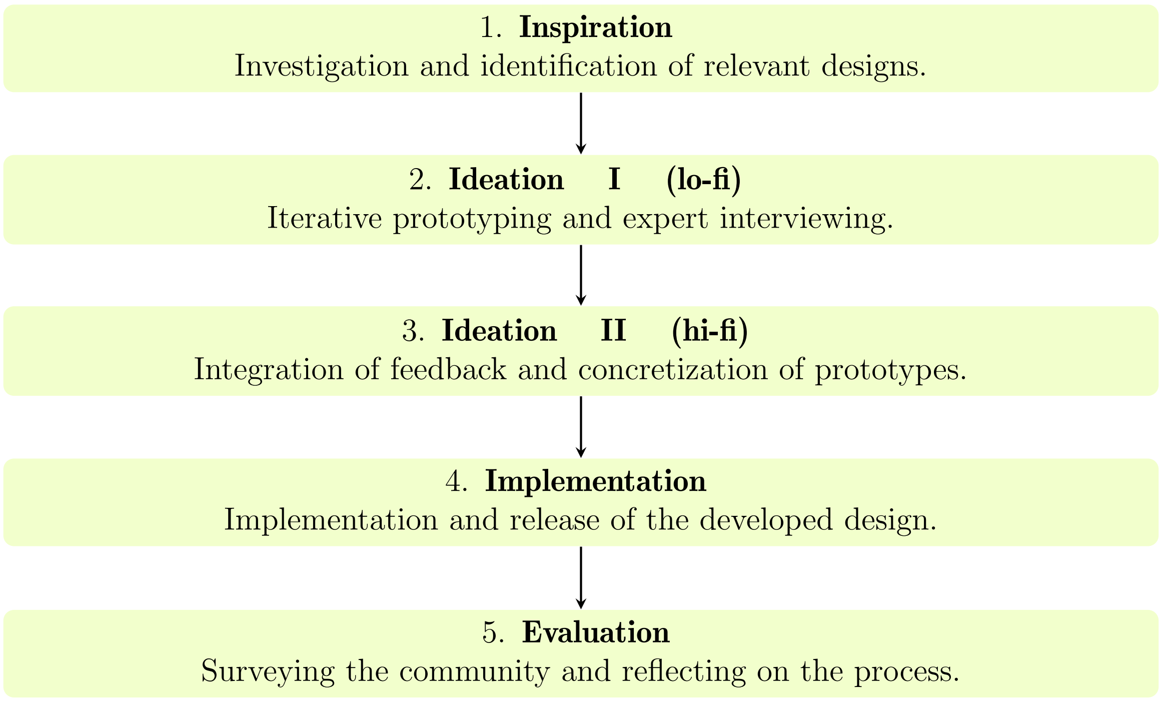
As Barth et al. [12] and Edwards and Abel [19] point out, privacy visualizations are not a novelty when it comes to empowering users through understandable forms of presentation, yet interactivity is a feature that is unfortunately not common but has already been brought up. The potential interaction here would focus on giving users control, rather than allowing them to provide feedback, for example. Whether such features and how control can be implemented depends entirely on access to specific APIs, which should be addressed in the design and implementation phases (cf. chapter 4).

While there are no standardized procedures yet in the privacy visualization research field [12], app stores have made great efforts in recent years to establish some standardized models that are adopted by most mainstream stores. On the other hand, centralized app stores (e.g., Play Store, Apple Store) still reflect the policies and practices of a data collection industry that places less emphasis on privacy. These practices, which do not go beyond what is legally mandated, make it difficult to implement privacy-friendly systems. However, in the open source space, where such institutions do not play a role, there have been few efforts to raise user awareness of privacy through visual design.

With regard to the goal (cf. section 1.2) of implementing an interactive privacy visualization for app stores that helps raise users' privacy awareness but does not get in their way, two questions need to be answered:

- What privacy visualization designs already exist and what are the advantages of one over the other?
- Which of the identified privacy visualization systems would the project's community prefer, and what improvements could be made?

1.3 Research Methodology & Structure


1. **Inspiration**
Investigation and identification of relevant designs.

2. **Ideation I (lo-fi)**
Iterative prototyping and expert interviewing.

3. **Ideation II (hi-fi)**
Integration of feedback and concretization of prototypes.

4. **Implementation**
Implementation and release of the developed design.

5. **Evaluation**
Surveying the community and reflecting on the process.

1. Inspiration

This phase focused on exploring the field, gathering knowledge about current issues, and being open to new possibilities [27]. Specifically, design considerations from relevant research were explored, data provided by *Exodus Privacy* and *F-Droid* were analyzed to identify a usable set of privacy parameters for creating privacy visualizations, and a set of specific design concepts were outlined.

Based on these findings, a theoretical framework was formulated and initial prototypes were developed in the next phase. For example, the framework builds on the fact that Stöver et al, Clever°Franke, and Bock et al. have each proposed different designs for privacy visualizations with different emphases [21, 41, 15], while Van Kleek et al. have shown that presenting users with different information leads to, among other things, more confidence in their decisions [44].

In this phase, Zotero [5] is used to collect, document, and annotate the examined literature.

2. Ideation I (lo-fi)

This phase focuses on channeling the knowledge acquired in the previous phase to identify opportunities, develop many ideas for possible solutions, and design initial prototypes [27]. This is the first of two design iterations aimed at creating initial prototypes based on the theoretical designs identified in the previous phase. These prototypes are then presented to a group of community experts for evaluation and discussion of possible improvements.

Penpot [6] is used here to create the first prototype sketches.

3. Ideation II (hi-fi)

The second design iteration is mainly concerned with further refining the prototypes, applying possible improvements and finalizing the design. Based on the expert interviews, a prototype was selected, which was further developed and finalized in consultation with the experts.

Penpot [6] is used here to further develop the prototypes and finalize the design concept.

4. Implementation

Bringing the interactive design of the privacy visualization to life and making it available to the community is the essence of this phase [27]. The implementation phase begins with defining the appropriate architecture for the developed interactive privacy visualization system and its subsequent implementation. After the implementation in *Neo Store*, a new version of the app was released featuring the new interactive privacy visualization design.

The system is implemented using the Android Studio [7] development environment and the code is written entirely in Kotlin [8]. While Jetpack Compose [9]

[5]An open source tool for managing bibliographies. https://www.zotero.org/

[6]An open Source design and prototyping platform. https://penpot.app

[7]Android's official development environment. https://developer.android.com/studio

[8]A modern multiplatform programming language. https://kotlinlang.org

[9]Android's declarative UI development toolkit. https://developer.android.com/jetpack/compose

library is used to define the user interface. Whereas the project is mainly hosted on GitHub [10].

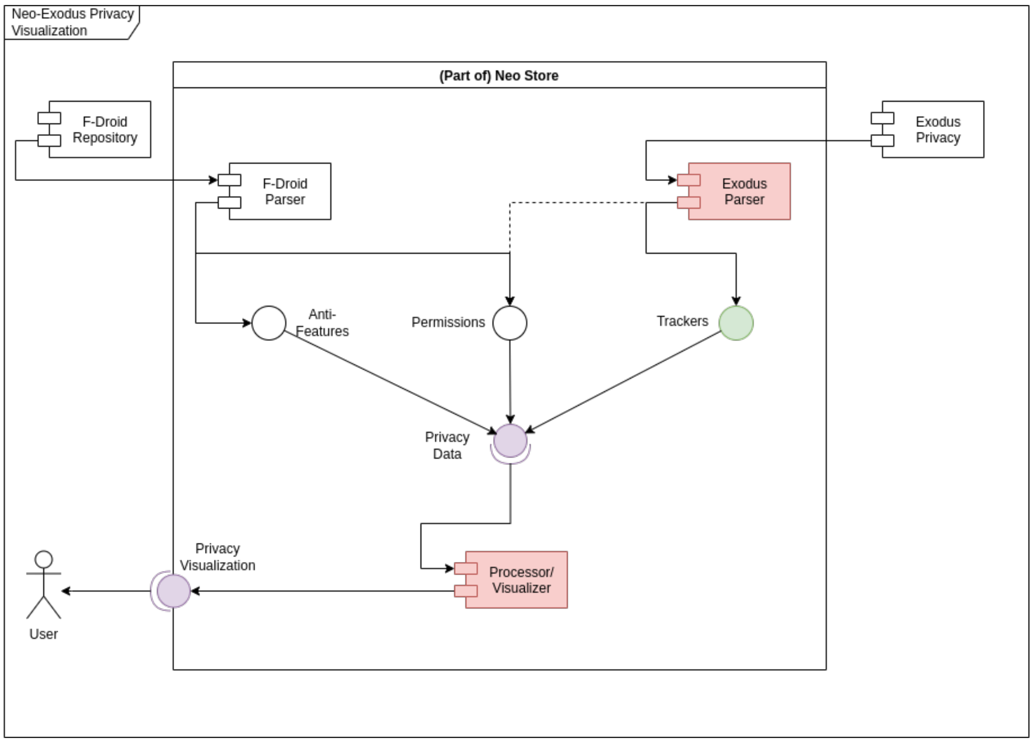


Figure 1.1: A first draft of the technical system (colored = to be implemented)

5. Evaluation

In this phase, the implemented design is evaluated by the community, and based on the community's feedback, further options are then discussed. The conclusion of the project includes a reflection and general evaluation of the overall development process, highlighting limitations and recommendations.

Cryptpad [11] Formular is used to create the survey and gather the responses.

[10] A popular Git hosting platform. https://github.com/

[11] An open source end-to-end encrypted collaboration suite. https://cryptpad.fr/

2 Inspiration

2.1 Related Work

As Barth et al. [12] point out in their systematic literature review of the research field around *privacy visualizations* and *Privacy by Design* policies, that there are still no defined standards and the available variants show a phase of divergence. Nevertheless, the systematic literature review also reveals a kind of systematic categorization of the content to be visualized. This basic categorization helps to identify the focus of each proposed implementation.

In the categorization, six basic categories were identified based on the questions:

- **What** data is being collected
- **How** is the data handled?
- **Why** (for what purpose) is the data collected?
- **How long** is the data retained?
- **Who** has access to the data?
- **Where** is the data saved?

In some of the discussed systems, there are also some smaller, more specific subcategories or special cases, which are still well defined in these systems and in some cases are hybrids of the enlisted categories.

While Pollach [33] questions the ethical justifiability of data collection without the user's informed consent. Martin and Shilton [30] and newly Reinhardt et al. [35] go a step further and conclude in their works that a clear positive correlation exists between meeting users' privacy expectations and the trustworthiness of mobile services. On this basis, and due to the fact that even interactive permission systems such as those found on the modern smartphones don't provide sufficient understanding of the privacy risks associated with the use of an application [18, 14], Barth et al. [12] conclude that privacy visualizations should be required by law because they are the best tool for achieving informed user consent, enabling wider adoption, e.g., ESRB's Game Ratings [26]. At the same time, they point out that only a couple of privacy visualizations are currently (still) being developed.

Most of the privacy visualizations discussed by Barth et al. [12] focus on general online browsing in the form of websites or services and do not explicitly apply to the use case in an application store. Nevertheless, these systems generally provide well reasoned guidelines for privacy visualization design. Few of them are even directly applicable to our use case for example, Van Kleek et al.'s [44] *"Data Controller Indicators (DCI)"* concluded that presenting users different privacy-related information leads to greater confidence in their decision-making. While CLEVER°FRANKE's [21] "Privacy Label" takes a universal approach to streamlining privacy visualizations so that they can be adopted for the analog world.

Further, Stöver et al. [41] examined various implementations of privacy indicators displayed in a user interface when a new mobile application is to be installed. The authors divided their study results into three criteria that characterize good privacy visualizations: (1) comprehensibility, (2) clear interpretation and (3) conveyance of further information. The key findings here are that users preferred quickly comprehensible visualizations over elaborate ones and visually rich (e.g., colorful) over monotonous ones.

On the other hand, Habib et al. [25] evaluated privacy icons and toggle buttons and found that most icons proposed to communicate privacy decisions are poorly interpreted and need to be paired with a contextual description to successfully signal privacy-related decisions. The main insight here is that while some stylized toggle switches are better interpreted in terms of user privacy choices than other formats, their function is almost always misunderstood as a one-click switch rather than a gateway to more information and control.

In the area of research with smartphone users, Frik et al. [22] showed that most users are unaware of existing privacy settings, while existing privacy interfaces do not provide users with enough information and control. On the other hand, with the goal of improving user experience in privacy control, Alsoubai et al. [10] identified different user types in terms of privacy strategies when interacting with apps and attempted to categorize the most important permissions into four internally correlated categories.

The implications of the above research, as well as the fact that most of the designs developed are either untested or poorly tested [36, 35], provide a strong argument for a real human-centered development of comprehensive privacy visualizations in relevant contexts (e.g., mobile applications, application store, analog-world privacy visualization) that underlie the importance of this work in terms of the design and development process.

In summary, many efforts in the research and practice have been made in various fields in recent years, most of which have invested in the development of optimized online privacy visualizations, while others have focused exclusively on specific application areas. On the one hand based on Barth et al.'s systematic literature review it's possible to map generic privacy-related questions to each of the suggested systems helping to identify relevant attributes for privacy visualizations [12]. Building on this, it has been made clear that perceived transparency regarding privacy concerns increases user trust and improves the user experience [35, 30]. On the other hand it should also be made clear that privacy visualizations are not a novelty in the field of empowering laypeople through visualizations. Based on previous experiences (e.g., EU/US energy labels [38, 29]), there is a clear tendency to make privacy visualizations legally binding when wider adoption is sought, thus providing a clear incentive for informed consent [12, 19, 36]. However, in support of the relevance of this work and its methodological approach, it was also noted that most of the existing designs have not been tested in real applications [36, 35], which makes the advantage of this work being developed and tested in a real application even more evident.

2.2 Conceptual Models

After a close look at the existing literature three central facts where made clear:

- Visual presentations are more effective than literal in empowering users in concern of privacy decision making.
- Privacy visualizations should have a balance between communicating as much information as possible and being easily comprehensible.
- Most of the designs developed earlier had little to no testing in real contexts.

Building on this, this work aims to develop, in a human-centered manner, an interactive privacy visualization design that is comprehensive and simple enough for its users, embedded in a real-world application.

On the technical side, *F-Droid* by itself does not provide sufficient data to create visualizations that can contribute much to users' informed decisions. Therefore, *F-Droid*'s metadata about applications must be supplemented by other sources. Fortunately, *Exodus Privacy* [3] provides the necessary data for more comprehensive privacy visualizations. In addition, the data from *Exodus Privacy* has already been used in several projects, most notably Oxford's TrackerControl [28] and "AppChecker" of the German project "Mobilsicher" [2]. These implementations provide a source of inspiration for what data might be meaningful to users and what quality can be achieved with the data provided.

As Barth et al. [12] pointed out, each of the existing privacy visualization designs contains a relatively small subset of privacy attributes that it communicates, so it remains simple while emphasizing certain aspects of privacy. The application of this conceptual model and the involvement of the user community as co-developers form the core of the development process in this work and allow the above facts to be addressed.

2.3 Design Requirements

Based on the insights of related work, privacy visualizations have one primary function, namely to communicate privacy-related information to users in a better and more concise way [12]. This function is supported primarily by the fact that visualizations are generally better communication tools than written formats, as is most evident when comparing a street sign to its written description, or in the context of privacy, when comparing, for example, the lengthy and difficult-to-understand online privacy policies to one of the alternative designs discussed by Reinhardt et al. [35].

Another feature suggested in the literature for privacy visualizations is providing a gateway to richer information and more control over privacy [35]. This provides a double benefit, as it can help increase users' perceived privacy control [42, 43] and improve the visibility of privacy settings that are typically less visible [22].

To sum up the above requirements: (1) The system consists of two distinct and connected layers, each with a specific function. (2) The first visual layer should provide as much privacy-related information as possible without compromising the comprehensibility of the visualization. (3) The second informative layer is intended to provide more comprehensive information and give users more control over their privacy.

The system will be based on these guidelines and evaluated at the end through an open, formal community survey. Requirement (1) is part of the conceptual process and should therefore be satisfied as soon as the system is completed. On the other hand, requirement (2) is validated by comparing the implemented design with prototypes based on other designs. While requirement (3) is satisfied if the second level receives overall good feedback given its content and control capabilities.

3 Ideation

3.1 Low-Fidelity Prototype

After gaining insights into the current state of research, its applications and potential leverage points during the inspiration phase, it was time to translate the accumulated findings into the first abstract prototypes. One of the goals of this study is to compare the different visualization formats proposed in the literature and to investigate the preferences regarding a complementary informative layer that offers more control and knowledge to the users. Low-fidelity prototyping will address two questions:

- Which privacy visualization designs are preferred by users? What are they doing well? And what could be improved?
- What content and tools could be presented in the second, informative layer?

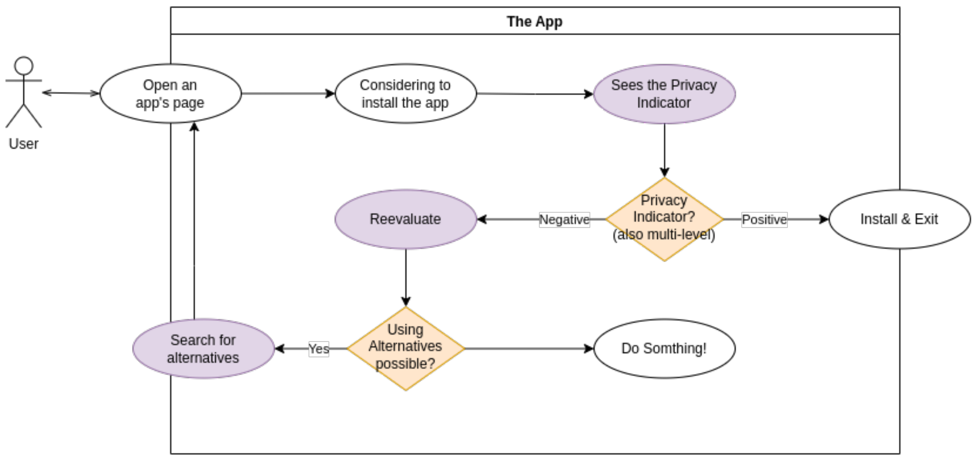

The App
User
Open an app's page
Considering to install the app
Sees the Privacy Indicator
Privacy Indicator? (also multi-level)
Negative
Positive
Reevaluate
Install & Exit
Using Alternatives possible?
Yes
Search for alternatives
Do Somthing!

Figure 3.1: A simplified user journey in the system (colored = additions to the existing system)

3.1.1 Bundle Ideas

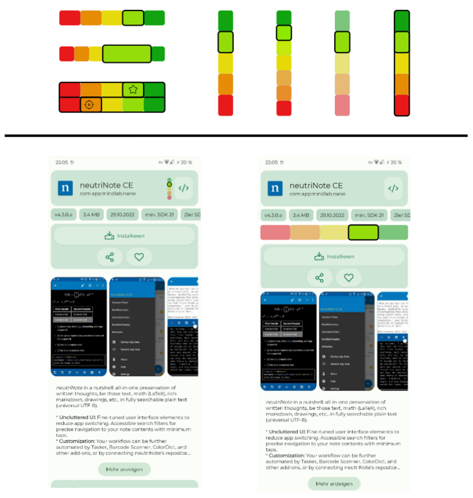
Comparing the design proposals examined in the inspiration phase, four categories of privacy visualization formats were identified:

- ***Meter Icon***: A simple icon that displays a privacy score on a colored scale. Based primarily on the proposal by Stöver et al. [41].
- ***State(ful) Icons***: A set of icons, each representing a particular property of the system. This can be implemented in one of two ways:
  - ***Stateful Icons***: A stable number of icons are represented, each of which represents a particular property of the system and has at least two possible states depending, for example, on whether the property is present or not. An example of this is Mozilla's *Privacy Icons* [34].
  - ***State Icons***: A set of icons, each indicating the presence of a particular property of the system. Of these icons, only those associated with existing system properties are displayed. One of the most recent academic efforts in this category is Rossi et al.'s DaPIS [36].
- ***Visual Label***: a modular format that translates curated information based on predefined questions into a visual form. The privacy label proposed by Clever°Franke is an example of such a format [21].
- ***Descriptive Label***: a concise text format supplemented by icons to facilitate navigation to privacy-related information. One of the most popular formats for visualizing privacy, as both Google's Data Safety Section and Apple's Privacy Labels are based on this design [11, 23].

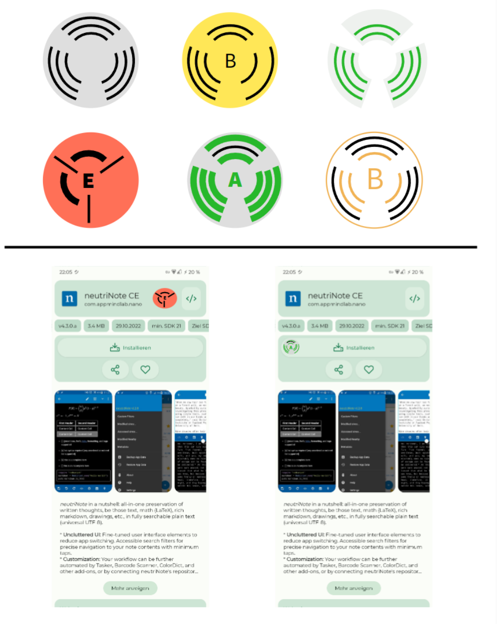
Each of these designs takes a different approach to conveying privacy-related information. While *Meter Icon*, *State(ful) Icons*, and *Visual Label* are appropriate for the first, visual layer, *Descriptive Label* can address the second, informative layer. Along these categories, I have begun prototyping three different designs in Penpot [6] for the visual layer and developing a fourth prototype *Descriptive Label* scheme based on the information collected about users' perceptions of privacy and the information provided by *Exodus Privacy* and F-Droid (see Figure 3.2).

Each of the designs builds on a more specific privacy concept or assessment approach than the original designs. *Visual Label* reduced the number of questions proposed by Clever°Franke from fifteen to nine, with three categories containing three questions each:

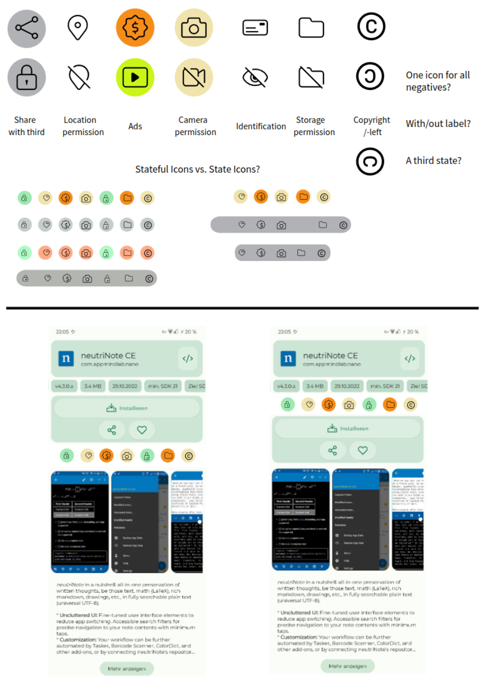
- Permissions:
  - The app doesn't ask for permissions for physical data?
  - The app doesn't ask for permissions for identification data?
  - The app doesn't ask for internet access?
- Trackers:
  - The app doesn't have MAGMA (Meta, Apple, Google, Microsoft, Amazon) trackers?



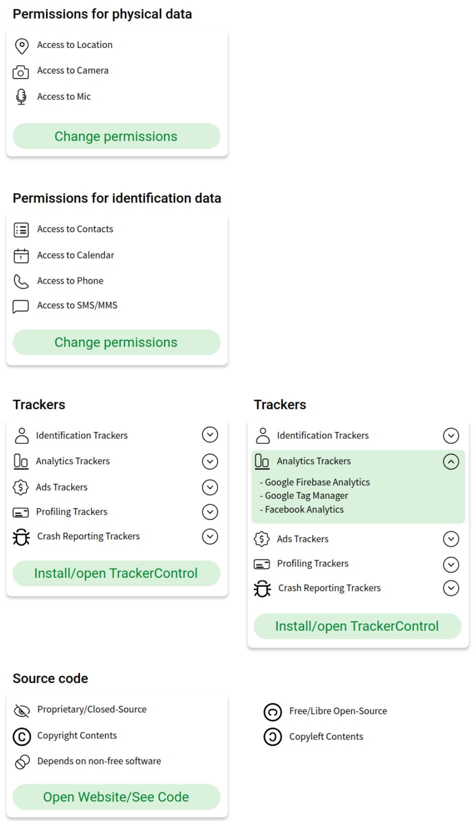
(a) Meter Icon



(b) Visual Label



(c) State(ful) Icon



Permissions for physical data

- Access to Location
- Access to Camera
- Access to Mic

Change permissions

Permissions for identification data

- Access to Contacts
- Access to Calendar
- Access to Phone
- Access to SMS/MMS

Change permissions

Trackers

- Identification Trackers
- Analytics Trackers
- Ads Trackers
- Profiling Trackers
- Crash Reporting Trackers

Install/open TrackerControl

Trackers

- Identification Trackers
- Analytics Trackers
  - Google Firebase Analytics
  - Google Tag Manager
  - Facebook Analytics
- Ads Trackers
- Profiling Trackers
- Crash Reporting Trackers

Install/open TrackerControl

Source code

- Proprietary/Closed-Source
- Copyright Contents
- Depends on non-free software

Open Website/See Code

- Free/Libre Open-Source
- Copyleft Contents

(d) Descriptive Label

Figure 3.2: The prototype boards used in the expert interviews. Each shows a number of different design and positioning options for a given design. These were discussed along with the possibility of suggesting other options.

Negative influence	Tracker groups	Minus points for trackers	Minus points for permissions
Low	Crash reporting	5	3
Medium	Analytics and Identification	10	7
High	Ads, Profiling and Location	20	15

Table 3.1: Categorization of anti-privacy attributes

  - The app doesn't have trackers based in non-free countries e.g. Yandex from Russia?
  - The app doesn't have Ads, Profiling or Location trackers?

- Source Code/License:
  - The app source code is publicly accessible to everyone?
  - The app doesn't include non-free assets?
  - The app doesn't depend on non-free software?

On the other hand, *Meter Icon* is based on a point system that starts with 100 points and decreases with each anti-privacy characteristic. In this context, the anti-privacy features are divided into three categories with respectively low, medium and high negative impact, which are explained in more detail in the Table 3.1. That said, the prototype of *Meter Icon* is essentially based on the results of Stöver et al. [41] and slightly inspired by the visual design of PrivacyRating.info [8].

The *State(ful) Icons* prototype combines two of the most widely tested designs in the research field. The two approaches, *Stateful Icons* and *State Icons*, as defined above, have much in common formally, but also follow different principles, which is why they were compared in the prototype board. The design provides a set of seven fields: sharing with third party, location permission, advertising trackers, camera permission, profiling or identification trackers, storage permission, and (being not) open source. Each field has at least two states and is represented by a single icon.

The last prototype, the *Descriptive Label*, builds heavily on the PrivacyLabel.org, Google, and Apple designs [11, 23, 7], separating the various specific privacy concerns into blocks with visible titles. The prototype also adds an action button that points to possible optimizations, such as using TrackerControl to block trackers, or leading to the appropriate control instance, such as the app's permissions page in System Preferences.

3.1.2 Get Feedback

As described in IDEO's Field Guide to Human-Centered Design, it can be very useful in the ideation phase to obtain feedback, which can come from a variety of sources [27]. One of these sources is expert semi-structured interviews, which were chosen for this first iteration because, unlike shorter, more concise formats, they provide qualitative feedback and allow for further questions to delve into specific details and explore new possibilities and aspects directly. This fits well with the goal of discovering new opportunities to improve the design while comparing existing designs to find the most promising.

For the semi-structured expert interviews, an abstract set of questions was created that focused on the experts' perceptions of each design and provided them an opportunity to suggest alternative options not considered in the prototyping process and to relate the privacy visualization system to users' perceived needs (Appendix C: Semi-structured Interview Questions).

To recruit interviewees, I announced my work on the study to the community via the project's Telegram channel and invited people to participate by describing the study and the function of the interviews. Eighteen people from the community of approximately 700 registered their interest in being interviewed, of which I had to write to nine to reach an acceptable number of four interviews, as some declined a real-time interview due to privacy concerns and some withdrew because they were not comfortable with any of the languages I offered for the interview (English, German, and Arabic). The selected interviewees received a "study consent" form prior to the interview, in which the aim and methodology of the study as well as their role were made transparent (Appendix D: Study Informed Consent).

Interviews took place digitally via Jitsi or Webex, whichever the interviewee preferred. In addition to showing great interest in the study, interviewees brought up other issues around *FOSS* development, the privacy-conscious community, and policies related to privacy visualization. For example, some pointed to their frustration that privacy regulations are still lacking or nonexistent, and therefore such bottom-up approaches are necessary to improve the situation - even if they are initially targeted at a smaller group of people. This is consistent with Edwards and Abel's finding that market-based approaches have failed because they have not reached a critical mass of users, while the legal incentives for industry are still lacking [19].

Another point that the experts made is that unfortunately only the larger, more established *FOSS* projects use open design practices, although it would be more effective for the smaller community projects to use participatory design practices as applied in this study. Given this, and how enriching the interviews were, I have indicated the willingness to promote such practices in the various *FOSS* projects in which I am involved [1].

[1]As a co-founder of Neo Collective, I'm involved in a couple of other projects and also maintain a few smaller projects. [6].

In the interviews, all experts made it quite clear that the *Visual Label* based on the Clever°Franke approach is too complex and not very user-friendly. This complexity results primarily from the complex form of presentation, which is far less accessible in small form factors such as on smartphone screens. Another factor was the extensive knowledge of the system required to even understand the visualization. On the other hand, some experts praised the data protection idea that this system represents.

On the other hand, *Meter Icon* was perceived as simple, as it largely hides the complexity of privacy-related terms and theoretically allows a quick decision on whether the user should seek further information (e.g., by going to the second layer) or whether a decision can be made without hesitation. As for the design, the experts had some clear preferences:

- Two clearly separated bars covering mainly two categories: Permissions and Trackers.
- Alpha masking of the unselected color fields.
- Positioning above the install button

In addition, the experts suggested placing the icons next to the bars instead of inside them.

State(ful) Icons, according to the experts, have a higher density of information in the same space and are also still relatively easy to understand, but require a somewhat better understanding of these privacy systems in order to use them optimally. This made it very difficult for most experts to decide on a preference and did not result in a clear decision, with two experts choosing each of the two designs and the remaining two experts not choosing any.

Choosing a preference between *Stateful* and *State Icons* also proved difficult, but in the end it was *Stateful Icons* that had more and better arguments. For *State Icons*, for example, formal static placement was preferred because it speeds up scanning the icons for specific information. For *Stateful Icons* icons, it was suggested to use staggered colors to allow faster grasping, and to cluster them so that one can see that they belong together. Other aspects noted by the experts include the need for a legend for the icons, which could be added to the second layer of the system, and the possibility of choosing quantity or severity as the main influencing factor for the color of the icons. One of the most interesting suggestions from one of the experts was to add a settings icon directly to the toolbar that would lead to the second level. Lastly, as for the position of the bar, there was no real preference.

When discussing the second layer, the experts made it clear that its addition is essential and were convinced by the amount of information provided. In these discussions, the second layer was given a new name due to its monitoring functionality, namely, the *"Privacy Panel"*. One of the big pluses of this design over others is that it embeds information about *Anti-Features* in a more understandable form. Two interesting suggestions from the experts were the addition of the status of each permission (whether it was granted or not) and the possibility to copy the signature code of the trackers, since some users may already use solutions other than *TrackerControl* that use the VPN permission, and therefore with the signature it is possible to block the trackers in the particular service.

In conclusion, in the low-fidelity prototyping phase, three different designs for the first layer were designed based on three different designs from the literature, and an initial prototype for the second layer (also known as the privacy panel) was designed based on the design for the *Descriptive Label.* These different designs were then presented and discussed with four experts from the community. These rejected the *Visual Label* design as too complicated and requiring explanation, but were not very decisive in their choice between *Meter Icon* and *Stateful Icons.* This indecisiveness led to the need for another iteration of feedback based on more concretized prototypes. While the *"Privacy Panel"* seemed almost ready for conceptualization, with only a few planned improvements suggested by the experts.

3.2 High-Fidelity Prototype

Building on the experts' feedback on the initial prototypes, the next steps involved integrating them into a second iteration of the prototypes, resulting in more concrete examples that were then sent to the experts for a final round of feedback to clarify their preferences. Finally, the feedback was integrated into the preferred design and the design phase was completed by creating a comprehensive concept that brought us closer to implementation.

3.2.1 Integrate Feedback & Iterate

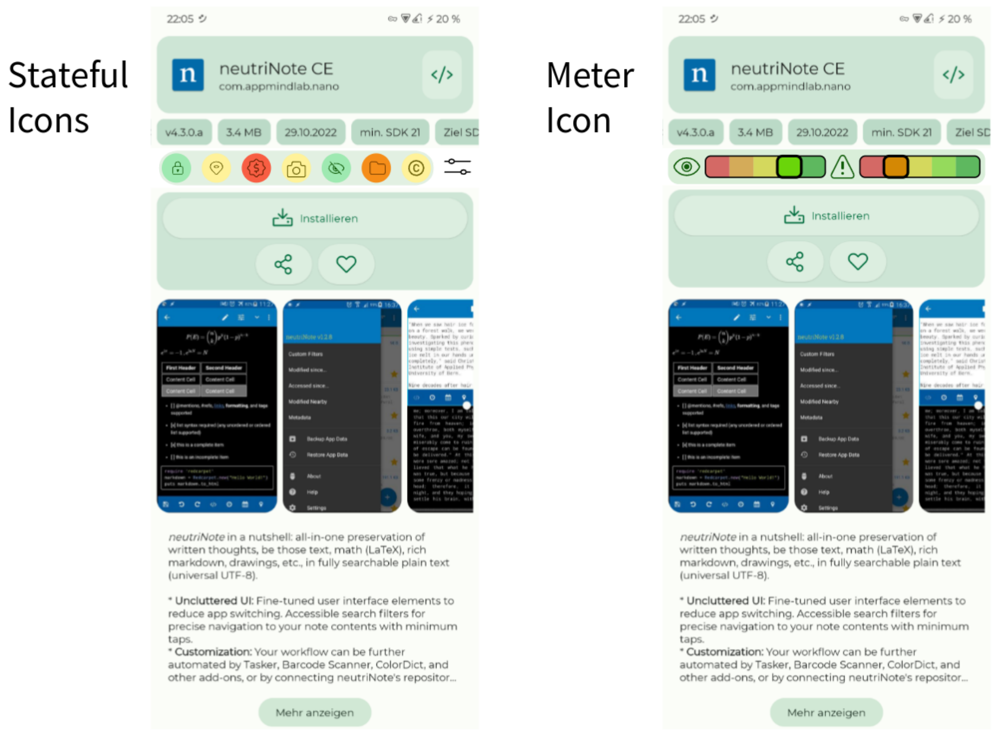
Based on the experts' preferences and suggestions, four design decisions were made for the *Meter Icon*:

- Two clearly separated bars, mainly covering the two categories of permissions and trackers.
- Alpha masking of unselected color fields.
- Positioning above the install button.
- Positioning of the icon next to the bar instead of in it.

On the other hand, four other decisions were made related to the *Stateful Icons*:

- Colored background of the icons.
- Cluster the icons with a surface background.

- Add an access icon next to the toolbar for the second layer.
- No labeling is required.



neutriNote in a nutshell: all-in-one preservation of written thoughts, be those text, math (LaTeX), rich markdown, drawings, etc., in fully searchable plain text (universal UTF-8).

* **Uncluttered UI:** Fine-tuned user interface elements to reduce app switching. Accessible search filters for precise navigation to your note contents with minimum taps.
* **Customization:** Your workflow can be further automated by Tasker, Barcode Scanner, ColorDict, and other add-ons, or by connecting neutriNote's repositor...

neutriNote in a nutshell: all-in-one preservation of written thoughts, be those text, math (LaTeX), rich markdown, drawings, etc., in fully searchable plain text (universal UTF-8).

* **Uncluttered UI:** Fine-tuned user interface elements to reduce app switching. Accessible search filters for precise navigation to your note contents with minimum taps.
* **Customization:** Your workflow can be further automated by Tasker, Barcode Scanner, ColorDict, and other add-ons, or by connecting neutriNote's repositor...

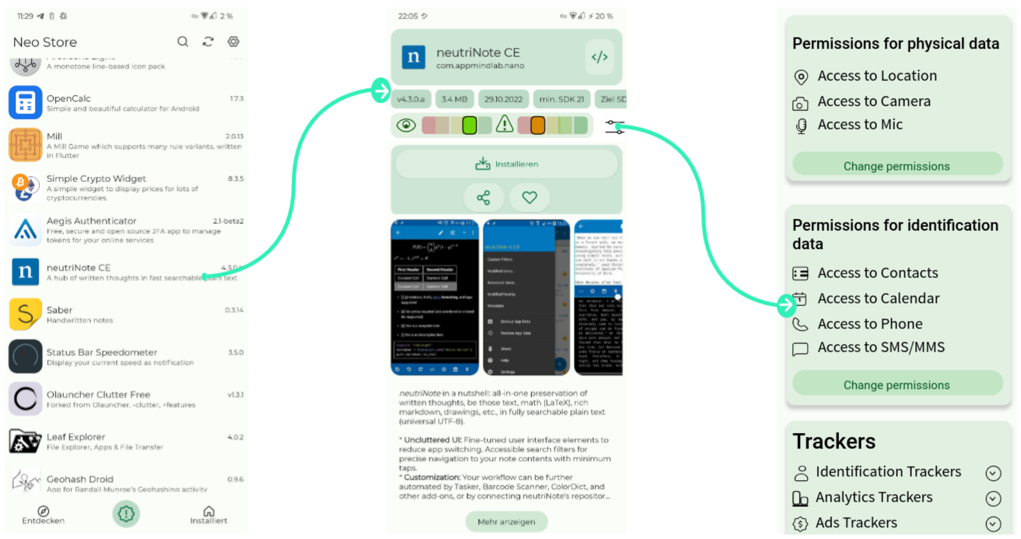
Figure 3.3: The last prototypes of the *Meter Icon* and *Stateful Icons* sent to the experts for feedback.

The result was two design prototypes of the system (see Figure 3.3), which were then sent to the experts for further feedback and to establish a clear preference if possible. This provided more concrete arguments for/against each of the two designs and some further suggestions for improvement. Exceptionally, two experts were now in favor of the *Meter Icon*, arguing for better accessibility and less cognitive load, while the other two experts opted for the *Stateful Icons*. One of the experts who voted for the *Stateful Icons* was the only expert who signaled flexibility in the choice, indicating how he could be won over to the other design. This ended up being one of the deciding factors in choosing *Meter Icon* over *Stateful Icons*. Other arguments in favor of the *Meter Icon* were the clear tendency in the literature to favor a balance between simplicity and information, with enough scientists favoring simplicity over “a little extra information” [19, 12, 35], and the fact that the *Privacy Panel* already provides richer information to adept users.

Five improvements were made based on feedback from experts:

- Increase of contrast between selected and deselected colors.

- Reduction of the use of frames.
- Converting the source code button into a source code information indicator.
- Adding a tooltip to describe the meaning of the color.
- Adding a control icon to the side of the bar.


11:29
Neo Store
A monotone line-based icon pack
OpenCalc 1.7.3
Simple and beautiful calculator for Android
Mill 2.0.13
A Mill Game which supports many rule variants, written in Flutter
Simple Crypto Widget 8.3.5
A simple widget to display prices for lots of cryptocurrencies
Aegis Authenticator 2.1-beta2
Free, secure and open source 2FA app to manage tokens for your online services
neutriNote CE 4.3.0
A hub of written thoughts in fast searchable plain text
Saber 0.3.14
Handwritten notes
Status Bar Speedometer 3.5.0
Display your current speed as notification
Olauncher Clutter Free v1.3.1
Forked from Olauncher. -clutter, +features
Leaf Explorer 4.0.2
File Explorer, Apps & File Transfer
Geohash Droid 0.9.6
App for Randall Munroe's Geohashing activity
Entdecken
Installiert
22:05
20 %
neutriNote CE
com.appmindlab.nano
v4.3.0.a
3.4 MB
29.10.2022
min. SDK 21
Ziel SD
Installieren
Mehr anzeigen
Permissions for physical data
Access to Location
Access to Camera
Access to Mic
Change permissions
Permissions for identification data
Access to Contacts
Access to Calendar
Access to Phone
Access to SMS/MMS
Change permissions
Trackers
Identification Trackers
Analytics Trackers
Ads Trackers

neutriNote in a nutshell: all-in-one preservation of written thoughts, be those text, math (LaTeX), rich markdown, drawings, etc., in fully searchable plain text (universal UTF-8).

* **Uncluttered UI:** Fine-tuned user interface elements to reduce app switching. Accessible search filters for precise navigation to your note contents with minimum taps.
* **Customization:** Your workflow can be further automated by Tasker, Barcode Scanner, ColorDict, and other add-ons, or by connecting neutriNote's repositor...

Figure 3.4: The final use flow sketch of the interactive privacy visualization system.

So, the prototyping process was completed with a defined design and a clear set of features based on decisions supported by a mix of literature, expert arguments, and preferences. The next step was to define an implementation concept for the design. The aim was to achieve a clear logical structure of the dual system, encompassing both layers.

3.2.2 Create a Concept

After completing the visual design, the next step was to create a comprehensive logical concept of the system, based on which the system would be implemented. To begin, an initial breakdown of the system into a data-oriented framework consisting of:

- the available data sources,
- the required data-holding objects and
- the visual design.

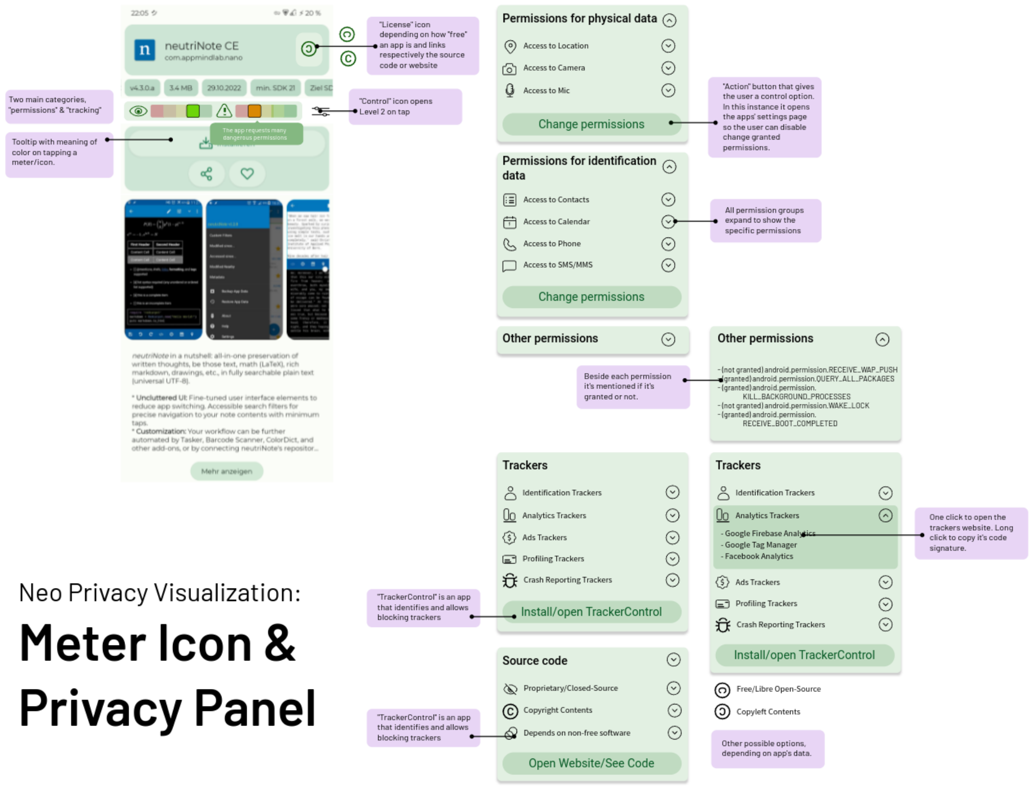

22:05
20 %
neutriNote CE
com.appmindlab.nano
"License" icon depending on how "free" an app is and links respectively the source code or website
v4.3.0.a
3.4 MB
29.10.2022
min. SDK 21
Ziel SD
Two main categories, "permissions" & "tracking"
"Control" icon opens Level 2 on tap
The app requests many dangerous permissions
Tooltip with meaning of color on tapping a meter/icon.
neutriNote in a nutshell: all-in-one preservation of written thoughts, be those text, math (LaTeX), rich markdown, drawings, etc., in fully searchable plain text (universal UTF-8).
* Uncluttered UI: Fine-tuned user interface elements to reduce app switching. Accessible search filters for precise navigation to your note contents with minimum taps.
* Customization: Your workflow can be further automated by Tasker, Barcode Scanner, ColorDict, and other add-ons, or by connecting neutriNote's repositor...
Mehr anzeigen
Permissions for physical data
Access to Location
Access to Camera
Access to Mic
Change permissions
"Action" button that gives the user a control option. In this instance it opens the apps' settings page so the user can disable change granted permissions.
Permissions for identification data
Access to Contacts
Access to Calendar
Access to Phone
Access to SMS/MMS
Change permissions
All permission groups expand to show the specific permissions
Other permissions
Beside each permission it's mentioned if it's granted or not.
Other permissions
-(not granted) android.permission.RECEIVE_WAP_PUSH
-(granted) android.permission.QUERY_ALL_PACKAGES
-(granted) android.permission.KILL_BACKGROUND_PROCESSES
-(not granted) android.permission.WAKE_LOCK
-(granted) android.permission.RECEIVE_BOOT_COMPLETED
Trackers
Identification Trackers
Analytics Trackers
Ads Trackers
Profiling Trackers
Crash Reporting Trackers
Install/open TrackerControl
"TrackerControl" is an app that identifies and allows blocking trackers
Trackers
Identification Trackers
Analytics Trackers
- Google Firebase Analytics
- Google Tag Manager
- Facebook Analytics
One click to open the trackers website. Long click to copy it's code signature.
Ads Trackers
Profiling Trackers
Crash Reporting Trackers
Install/open TrackerControl
Neo Privacy Visualization:
Meter Icon & Privacy Panel
Source code
Proprietary/Closed-Source
Copyright Contents
Depends on non-free software
Open Website/See Code
"TrackerControl" is an app that identifies and allows blocking trackers
Free/Libre Open-Source
Copyleft Contents
Other possible options, depending on app's data.

Figure 3.5: The final design prototype based on *Meter Icon* and *Descriptive Label*

This made the dependence of the system on the three data sources clear as well as the need for two data objects in the form of “Privacy Data” and “Privacy Note”. “Privacy Data” contains the sum of the required data and can be visualized directly in the “Privacy Panel”. While “Privacy Note” is a processed form of “Privacy Data” that can be mapped directly to the *Meter Icon* Design (see Figure 3.6).

Building on this, I dove deeper into the details by adding the appropriate fields to the “Privacy Data” and “Privacy Note” objects, drawing the relationships between the sources, the data objects, and the visual design, and adding indicators of where the data processing takes place, thus formulating a more concrete concept. This concept (see Figure 3.7) meets the main requirements of the design and covers the basic logical scheme of the system, so the architecture would only be an extension of the level of detail.

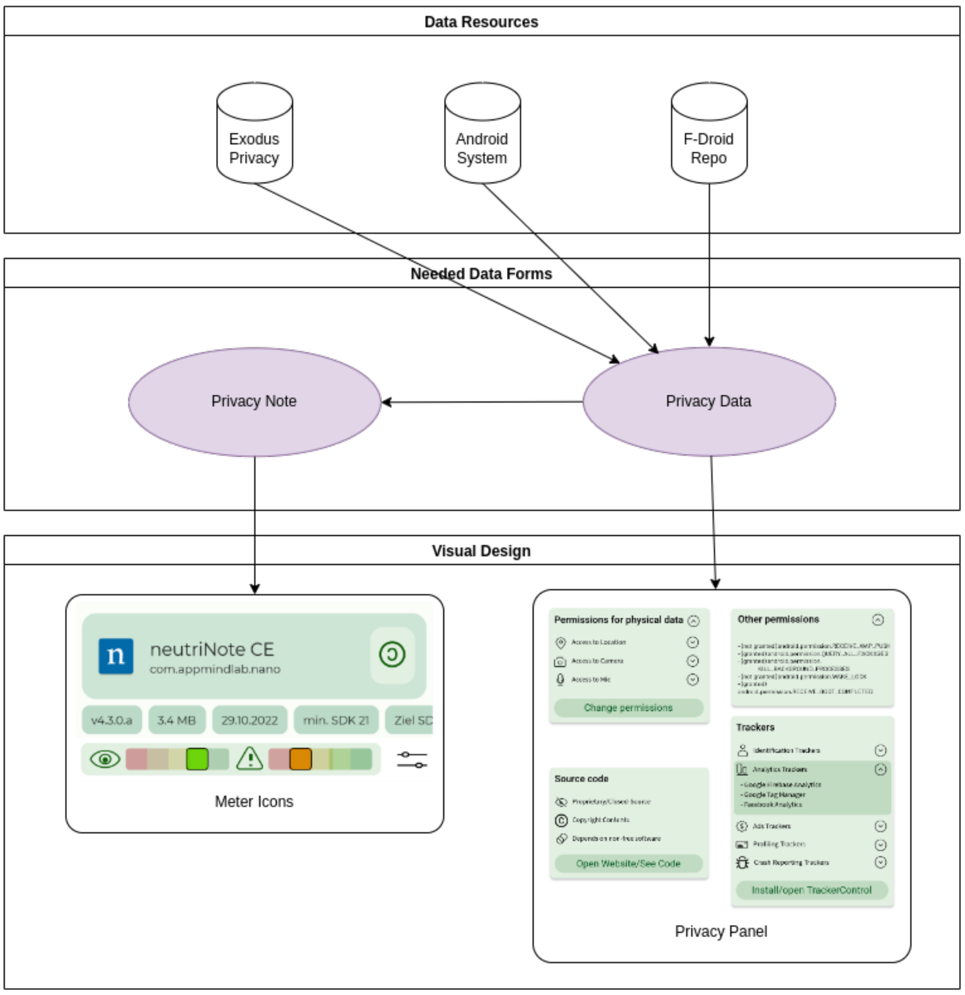


Figure 3.6: The initial framework of the system

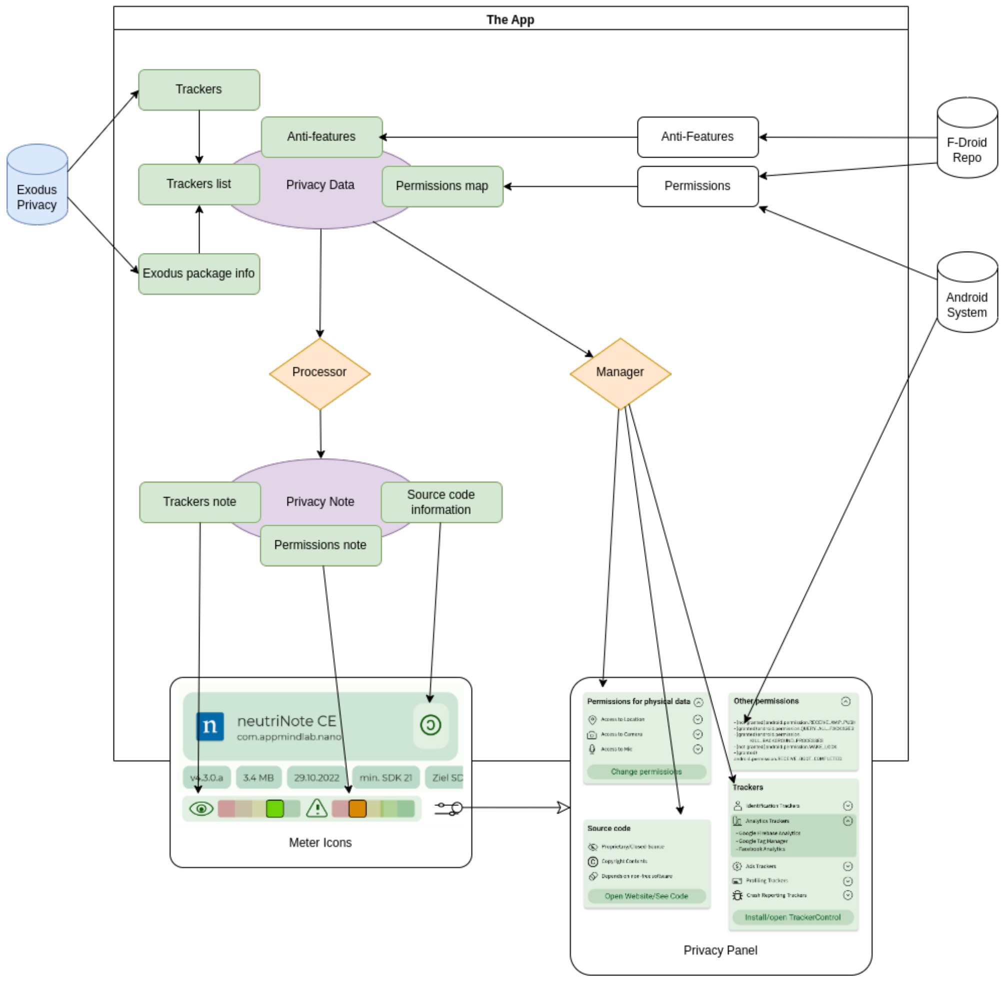


Figure 3.7: The final concept diagram of the system

4 Implementation

4.1 System Architecture

The next step was to concretize the architecture of the system based on the concept. This was mainly done by extending the concept diagram to a component diagram, still distinguishing between sources, classes, fields, and components. I went through the data flow backwards, from UI elements to data sources, which allowed me to identify these structural requirements:

- The permissions and trackers notes in the *Meter Icon* can be integers between 0-100. The source type is a separate class that contains three Boolean fields: open, free, and independent.
- Privacy Note's Trackers Note, Permissions Note and Source Type depend on Privacy Data's trackers list, permissions map and anti-features list respectively.
- *Privacy Panel* on the other hand uses Privacy Data directly. It just needs to group the data for each card (especially permissions for physical data, identification data and others based on the groups).
- Permissions are assigned to custom permission groups, which is handled in `Utils.kt`'s `getPermissionGroup()`, since Android's grouping of permissions is based on technical terms and doesn't have much regard for comprehensibility.
- To simplify the calls in the respective services, the Exodus API should be packaged according to the repository pattern. For this purpose, `Exodus Repository` should be created, which implements the interface of the API and processes the calls internally.
- The *Exodus Privacy* objects should be stored in the database to avoid repeated unnecessary downloads. Two separate DAOs are needed so that the tables are accessible on the view models side and the services side.
- Package Manager is Android's data source about an app's permissions and whether they are granted.

With these additions, the system takes on a more concrete form with a clear architecture, based on which the design can now be implemented. In the next and final step of the development, the system will be implemented in *Neo Store* and a first version will be released.

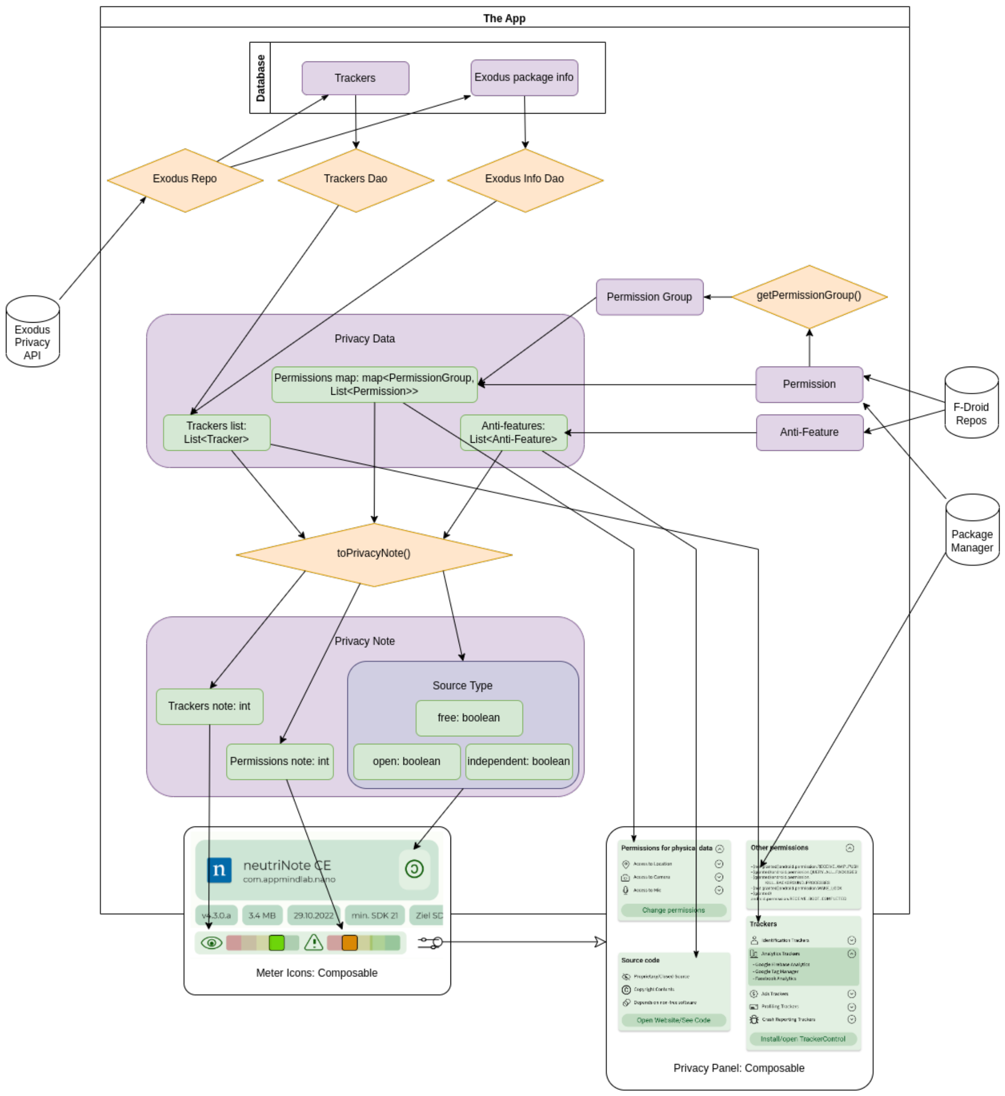


Figure 4.1: The designed architecture of the interactive privacy visualization system.

4.2 Technical Implementation

With the implementation I started from the bottom of the architecture (the top of the diagram), basically the back-end, following the tree till reaching the front-end. The system was implemented on a separate branch called `neo_privacy_visualization` to facilitate identification of relevant code additions and thus reproducibility.

4.2.1 Back-End

For the backend implementation, I considered the *Exodus Privacy* API, the entities needed in the database on one side and permissions, permission groups and anti-features on the other. Already in the inspiration phase, I took a look at *Exodus Privacy*'s API and realized that I needed a key, which I requested from their team and received within a week. To easily manage the API calls (short for *Exodus Privacy* API in this section), I added the library *Retrofit* [1], which was mainly used to define an internal API interface for the GET calls, while Retrofit's converter, *Moshi* [2], was added to enable automatic parsing of the JSON objects based on the corresponding Kotlin classes.

The API features a GET call to retrieve the list of trackers [3], `trackers`, which contained the full information about each tracker with a numeric identifier.

```
{
  "trackers": {
    "69": {
      "name": "Facebook Places",
      "network_signature": "\\.facebook\\.com",
      "code_signature": "com.facebook.places",
      "creation_date": "2017-12-05",
      "website": "https://developers.facebook.com/docs/android",
      "description": "",
      "categories": ["Analytics", "Ads"],
      "documentation": ["https://developers.facebook.com/docs/graph-api/
          reference/page/locations/"]
    },
    ...
  }
}
```

Based on this, two classes were created to allow *Retrofit* to automatically map the JSON objects:

```
open class TrackerData(
    open val name: String = String(),
    open val network_signature: String = String(),
    open val code_signature: String = String(),
    open val creation_date: String = String(),
    open val website: String = String(),
    open val description: String = String(),
    open val categories: List<String> = emptyList(),
    open val documentation: List<String> = emptyList(),
)

data class Trackers(
```

[1]Retrofit is a type-safe HTTP client for Android built on top of OkHttp that simplifies REST calls.

[2]Moshi is a modern JSON serializer with support for Android, Java and Kotlin.

[3]Exodus Privacy API: https://github.com/Exodus-Privacy/exodus/blob/v1/doc/api.md

```
    val trackers: Map<String, TrackerData> = emptyMap()
)
```

Then a *Room*[4] entity class was defined that contains the tracker identifier with the tracker data:

```
@Entity
data class Tracker(
    @PrimaryKey
    val key: Int = 0,
    override val name: String = String(),
    override val network_signature: String = String(),
    override val code_signature: String = String(),
    override val creation_date: String = String(),
    override val website: String = String(),
    override val description: String = String(),
    override val categories: List<String> = emptyList(),
    override val documentation: List<String> = emptyList(),
) : TrackerData(
    name,
    network_signature,
    code_signature,
    creation_date,
    website,
    description,
    categories,
    documentation,
)
```

On the other hand, the API has two different calls for app details:

- a simple one that packages the apps into one object based on the package name and includes a field with a list of reports, each representing a version of the app

- and a detailed one that responds with a list of objects, each of which represents a detailed report about each app version, including other information such as permissions, hashes, and so on.

Since the simple call provides a better flat-hierarchical response and the additional information provided by the detailed reports either has no use to the system or is already provided from other sources (e.g., permissions from the F-Droid repository), I opted for the simple call `GET /api/search/$packageName`. The corresponding object and *Room* entity classes were not very complicated, since the response is a single object and only the package name is added to the entity compared to the response object:

[4]A SQLite abstraction layer for Android to simplify creation of and access to SQLite databases.

```
@Entity
data class ExodusInfo(
    @PrimaryKey
    val packageName: String = "",
    override val handle: String = String(),
    override val app_name: String = String(),
    override val uaid: String = String(),
    override val version_name: String = String(),
    override val version_code: String = String(),
    override val source: String = String(),
    override val icon_hash: String = String(),
    override val apk_hash: String = String(),
    override val created: String = String(),
    override val updated: String = String(),
    override val report: Int = 0,
    override val creator: String = String(),
    override val downloads: String = String(),
    override val trackers: List<Int> = emptyList(),
    override val permissions: List<String> = emptyList()
) : ExodusData(
    handle, app_name, uaid, version_name, version_code, source,
    icon_hash, apk_hash, created, updated, report, creator, downloads,
        trackers, permissions,
)

open class ExodusData(
    open val handle: String = String(),
    open val app_name: String = String(),
    open val uaid: String = String(),
    open val version_name: String = String(),
    open val version_code: String = String(),
    open val source: String = String(),
    open val icon_hash: String = String(),
    open val apk_hash: String = String(),
    open val created: String = String(),
    open val updated: String = String(),
    open val report: Int = 0,
    open val creator: String = String(),
    open val downloads: String = String(),
    open val trackers: List<Int> = emptyList(),
    open val permissions: List<String> = emptyList()
) {
    fun toExodusInfo(packageName: String) = ExodusInfo(
        packageName, handle, app_name, uaid, version_name, version_code,
            source,
        icon_hash, apk_hash, created, updated, report, creator, downloads
            , trackers, permissions
    )
}
```

Also, a converter for `List<Int>` has been added to the converter class, `Converters.kt`, to allow automatic serialization of the tracker list. Using the database

Room requires a DAO [5] for each entity, which I defined building on the predefined `BaseDao<T>`:

```
interface BaseDao<T> {
    @Insert
    fun insert(vararg product: T)

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insertReplace(vararg product: T)

    @Update(onConflict = OnConflictStrategy.REPLACE)
    fun update(vararg obj: T): Int

    @Delete
    fun delete(obj: T)
}

@Dao
interface TrackerDao : BaseDao<Tracker> {
    @get:Query("SELECT * FROM `tracker`")
    val all: List<Tracker>

    @get:Query("SELECT * FROM `tracker`")
    val allFlow: Flow<List<Tracker>>

    @Query("SELECT * FROM `tracker` WHERE key = :key")
    fun get(key: Int): Tracker?

    @Query("SELECT * FROM `tracker` WHERE key = :key")
    fun getFlow(key: Int): Flow<Tracker?>
}

@Dao
interface ExodusInfoDao : BaseDao<ExodusInfo> {

    @Query("SELECT * FROM `exodusinfo` WHERE packageName = :packageName
        ")
    fun get(packageName: String): List<ExodusInfo>

    @Query("SELECT * FROM `exodusinfo` WHERE packageName = :packageName
        ")
    fun getFlow(packageName: String): Flow<List<ExodusInfo>>

}
```

Then I added the DAOs and an `AutoMigration` [6] object to the database instance:

[5] Short for "Data Access Object": a pattern that provides an access abstraction layer for databases.

[6] Auto-Migration is a feature of Room to automatically handle the migration of the database from one version to another, such as adding the new tables.

```kotlin
@Database(
    entities = [
        ...
        ExodusInfo::class,
        Tracker::class,
    ],
    version = 13,
    ...
    autoMigrations = [
    ...
      AutoMigration(
        from = 12,
        to = 13,
      )
    ]
)
@TypeConverters(Converters::class)
abstract class DatabaseX : RoomDatabase() {
    ...
    abstract val exodusInfoDao: ExodusInfoDao
    abstract val trackerDao: TrackerDao
    ...
}
```

With that, the database was ready to store the retrieved data to be used in the data flows to the frontend. On the other hand, the client of the API was still not ready, so the next steps were to define an interface (`IExodusAPI`), create a client module (`ExodusModule`) and create a repository instance (`RExodusAPI`). All of these objects were then created using *Dagger* [7] *Hilt* [8].

Using the existing API repository, it was possible to add internal retrieval functions to the synchronization service, `SyncService`. Finally, a public *Binder* function was added to retrieve Exodus data of a specific package, accessing the function through the *Binder* object, while a special request key (`EXODUS_TRACKERS_SYNC`) was defined for retrieving trackers task and the function was called in `handleNextTask()`, adding it to the manual synchronization process:

```kotlin
@AndroidEntryPoint
class SyncService : ConnectionService<SyncService.Binder>() {
    ...
    @Inject
    lateinit var repoExodusAPI: RExodusAPI

inner class Binder : android.os.Binder() {
        ...
        fun fetchExodusInfo(packageName: String) = fetchExodusData(
            packageName)
```

[7]Dagger is a fully static, compile-time dependency injection network for Kotlin, Java, and Android.

[8]Hilt is a Dagger overlay library for Android to simplify the use of Dagger and reduce boilerplate code.

```

        fun sync(request: SyncRequest) {
            scope.launch {
                val ids = db.repositoryDao.all.filter { it.enabled }.map
                    { it.id }.toList() + EXODUS_TRACKERS_SYNC
                sync(ids, request)
            }
        }

        private fun handleNextTask(hasUpdates: Boolean) {
            ...
            if (repository != null && repository.enabled && task.
                repositoryId != EXODUS_TRACKERS_SYNC) {
                ...
            } else if (task.repositoryId == EXODUS_TRACKERS_SYNC) {
                fetchTrackers() // the internal function
                handleNextTask(hasUpdates)
            } else {
                handleNextTask(hasUpdates)
            }
            ...
        }
    }
    ...
}
```

The Exodus data retrieval is only needed when the user calls the app's page. I was limited to this implementation because the API does not provide a GET call for a list of packet data and is limited to 30 calls in one second. Also, it is not efficient to store the Exodus data from thousands of apps, only a handful of which the user may be accessing, until the data is updated and therefore should be replaced. Once all that is done, the database flows can now be added to the respective view model and UI page:

```
class AppViewModelX(val db: DatabaseX, val packageName: String,
    developer: String) : ViewModel() {
    ...
    @OptIn(ExperimentalCoroutinesApi::class)
    val exodusInfo = db.exodusInfoDao.getFlow(packageName)
        .mapLatest { it.maxByOrNull(ExodusInfo::version_code) ?:
            ExodusInfo() }

    val trackers = exodusInfo.combine(db.trackerDao.allFlow) { a, b ->
        b.filter { it.key in a.trackers }
    }
    ...
}

class AppSheetX() : FullscreenBottomSheetDialogFragment(), Callbacks {
    ...
    @Composable
    fun AppSheet() {
        ...
        val exodusInfo by viewModel.exodusInfo.collectAsState(ExodusInfo
            ())
        val trackers by viewModel.trackers.collectAsState(emptyList())
        ...
    }
}
```

4.2.2 Privacy Processor

The translation of the data collected in the back-end into a form usable by the front-end can be considered part of the back-end, but is discussed separately from the other back-end components in this paper in the spirit of separating static and dynamic logic.

Based on the architecture, *PrivacyData* and *PrivacyNote* are the essential objects in the system that contain the usable data, and were therefore initially defined as the simple data classes that they are. The only exception was *PrivacyNote*'s *SourceType*, which had its own data class with three Boolean values for each of the properties:

- is it open-source?
- is it free (as in freedom)?
- is it independent from non-free software?

The permissions field of *PrivacyData* is a map from user-defined groups to the respective set of permissions requested by the application. So, to generate the map from the permissions list held by the *Release* object:

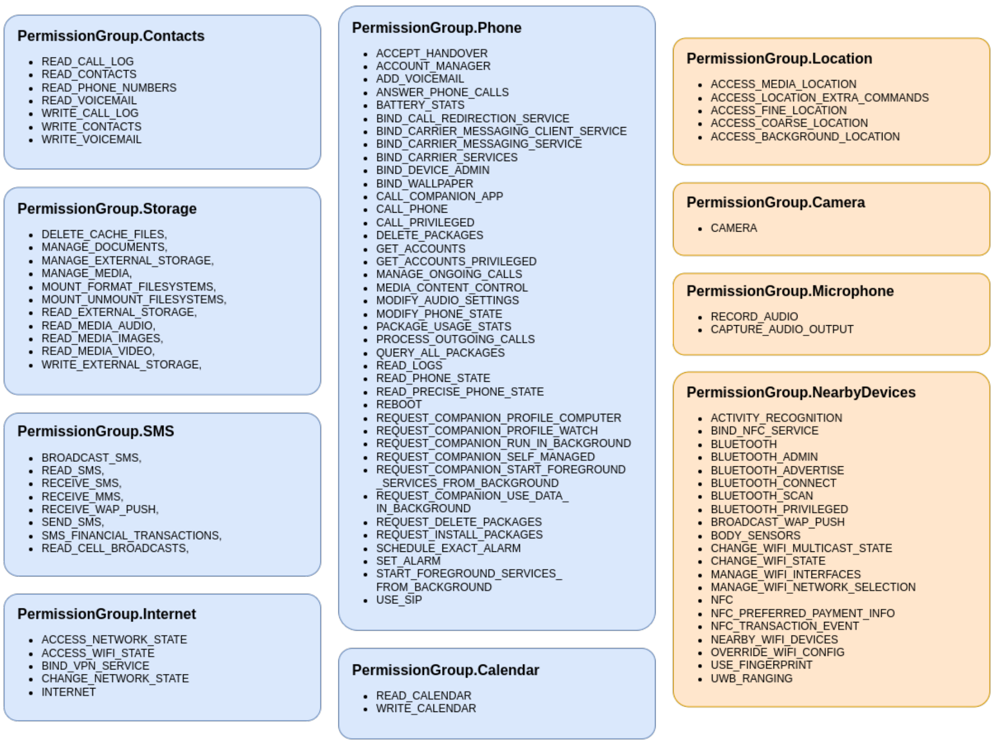
1. Eight groups of permissions were created based on the Alsoubi et al. model [10], plus two groups:
   - *Internet*: undeniably the most important non-runtime permission (Android still grants it automatically to any app that requests it, despite its importance for real user control),
   - and *Nearby Devices*: a relatively new set of permissions that has grown in recent years as the number of interconnected devices (e.g., Internet of things) has increased.

   This grouping includes only those permissions identified as relevant or high risk, all other permissions are included in *Other*

2. Based on the Alsoubi et al. model, the permission groups were also divided into those related to the physical world and those related to identification data. *Internet* was considered a permission group of identification data, while *Nearby Devices* a permission group of physical world data.

3. Then, the identified permissions were again classified into a series of groups according to their risk: high risk, medium risk, and low risk, to allow calculation of the negative impact of a permission based on more than just the permission group to which it belongs.

4. For the constructor of *PrivacyData*, *Release*'s function has been modified to use the new grouping logic when creating the permission map.



PermissionGroup.Contacts

- READ_CALL_LOG
- READ_CONTACTS
- READ_PHONE_NUMBERS
- READ_VOICEMAIL
- WRITE_CALL_LOG
- WRITE_CONTACTS
- WRITE_VOICEMAIL

PermissionGroup.Storage

- DELETE_CACHE_FILES,
- MANAGE_DOCUMENTS,
- MANAGE_EXTERNAL_STORAGE,
- MANAGE_MEDIA,
- MOUNT_FORMAT_FILESYSTEMS,
- MOUNT_UNMOUNT_FILESYSTEMS,
- READ_EXTERNAL_STORAGE,
- READ_MEDIA_AUDIO,
- READ_MEDIA_IMAGES,
- READ_MEDIA_VIDEO,
- WRITE_EXTERNAL_STORAGE,

PermissionGroup.SMS

- BROADCAST_SMS,
- READ_SMS,
- RECEIVE_SMS,
- RECEIVE_MMS,
- RECEIVE_WAP_PUSH,
- SEND_SMS,
- SMS_FINANCIAL_TRANSACTIONS,
- READ_CELL_BROADCASTS,

PermissionGroup.Internet

- ACCESS_NETWORK_STATE
- ACCESS_WIFI_STATE
- BIND_VPN_SERVICE
- CHANGE_NETWORK_STATE
- INTERNET

PermissionGroup.Phone

- ACCEPT_HANDOVER
- ACCOUNT_MANAGER
- ADD_VOICEMAIL
- ANSWER_PHONE_CALLS
- BATTERY_STATS
- BIND_CALL_REDIRECTION_SERVICE
- BIND_CARRIER_MESSAGING_CLIENT_SERVICE
- BIND_CARRIER_MESSAGING_SERVICE
- BIND_CARRIER_SERVICES
- BIND_DEVICE_ADMIN
- BIND_WALLPAPER
- CALL_COMPANION_APP
- CALL_PHONE
- CALL_PRIVILEGED
- DELETE_PACKAGES
- GET_ACCOUNTS
- GET_ACCOUNTS_PRIVILEGED
- MANAGE_ONGOING_CALLS
- MEDIA_CONTENT_CONTROL
- MODIFY_AUDIO_SETTINGS
- MODIFY_PHONE_STATE
- PACKAGE_USAGE_STATS
- PROCESS_OUTGOING_CALLS
- QUERY_ALL_PACKAGES
- READ_LOGS
- READ_PHONE_STATE
- READ_PRECISE_PHONE_STATE
- REBOOT
- REQUEST_COMPANION_PROFILE_COMPUTER
- REQUEST_COMPANION_PROFILE_WATCH
- REQUEST_COMPANION_RUN_IN_BACKGROUND
- REQUEST_COMPANION_SELF_MANAGED
- REQUEST_COMPANION_START_FOREGROUND _SERVICES_FROM_BACKGROUND
- REQUEST_COMPANION_USE_DATA_ IN_BACKGROUND
- REQUEST_DELETE_PACKAGES
- REQUEST_INSTALL_PACKAGES
- SCHEDULE_EXACT_ALARM
- SET_ALARM
- START_FOREGROUND_SERVICES_ FROM_BACKGROUND
- USE_SIP

PermissionGroup.Calendar

- READ_CALENDAR
- WRITE_CALENDAR

PermissionGroup.Location

- ACCESS_MEDIA_LOCATION
- ACCESS_LOCATION_EXTRA_COMMANDS
- ACCESS_FINE_LOCATION
- ACCESS_COARSE_LOCATION
- ACCESS_BACKGROUND_LOCATION

PermissionGroup.Camera

- CAMERA

PermissionGroup.Microphone

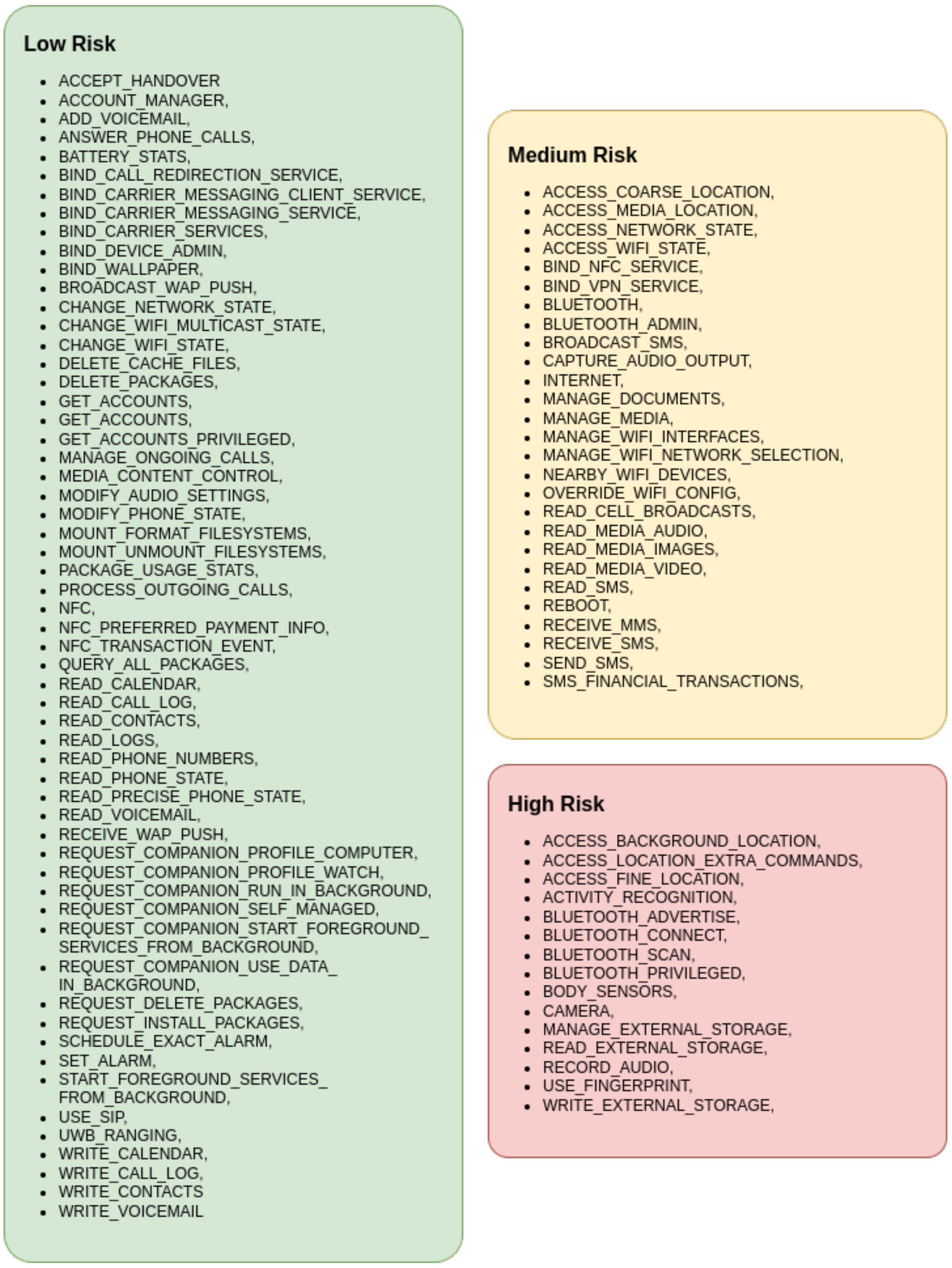
- RECORD_AUDIO
- CAPTURE_AUDIO_OUTPUT

PermissionGroup.NearbyDevices

- ACTIVITY_RECOGNITION
- BIND_NFC_SERVICE
- BLUETOOTH
- BLUETOOTH_ADMIN
- BLUETOOTH_ADVERTISE
- BLUETOOTH_CONNECT
- BLUETOOTH_SCAN
- BLUETOOTH_PRIVILEGED
- BROADCAST_WAP_PUSH
- BODY_SENSORS
- CHANGE_WIFI_MULTICAST_STATE
- CHANGE_WIFI_STATE
- MANAGE_WIFI_INTERFACES
- MANAGE_WIFI_NETWORK_SELECTION
- NFC
- NFC_PREFERRED_PAYMENT_INFO
- NFC_TRANSACTION_EVENT
- NEARBY_WIFI_DEVICES
- OVERRIDE_WIFI_CONFIG
- USE_FINGERPRINT
- UWB_RANGING

Figure 4.2: The defined permission groups (blue and orange for the permissions of identification data and physical data, respectively).

```kotlin
object PackageItemResolver {
    ...
    fun getPermissionGroup(permissionInfo: PermissionInfo):
        PermissionGroup {
        return if (Android.sdk(29)) {
            when (permissionInfo.name) {
                in CONTACTS_PERMISSIONS -> PermissionGroup.Contacts
                in CALENDAR_PERMISSIONS -> PermissionGroup.Calendar
                in SMS_PERMISSIONS -> PermissionGroup.SMS
                in STORAGE_PERMISSIONS -> PermissionGroup.Storage
                in PHONE_PERMISSIONS -> PermissionGroup.Phone
                in LOCATION_PERMISSIONS -> PermissionGroup.Location
                in MICROPHONE_PERMISSIONS -> PermissionGroup.Microphone
```



Low Risk

- ACCEPT_HANDOVER
- ACCOUNT_MANAGER,
- ADD_VOICEMAIL,
- ANSWER_PHONE_CALLS,
- BATTERY_STATS,
- BIND_CALL_REDIRECTION_SERVICE,
- BIND_CARRIER_MESSAGING_CLIENT_SERVICE,
- BIND_CARRIER_MESSAGING_SERVICE,
- BIND_CARRIER_SERVICES,
- BIND_DEVICE_ADMIN,
- BIND_WALLPAPER,
- BROADCAST_WAP_PUSH,
- CHANGE_NETWORK_STATE,
- CHANGE_WIFI_MULTICAST_STATE,
- CHANGE_WIFI_STATE,
- DELETE_CACHE_FILES,
- DELETE_PACKAGES,
- GET_ACCOUNTS,
- GET_ACCOUNTS,
- GET_ACCOUNTS_PRIVILEGED,
- MANAGE_ONGOING_CALLS,
- MEDIA_CONTENT_CONTROL,
- MODIFY_AUDIO_SETTINGS,
- MODIFY_PHONE_STATE,
- MOUNT_FORMAT_FILESYSTEMS,
- MOUNT_UNMOUNT_FILESYSTEMS,
- PACKAGE_USAGE_STATS,
- PROCESS_OUTGOING_CALLS,
- NFC,
- NFC_PREFERRED_PAYMENT_INFO,
- NFC_TRANSACTION_EVENT,
- QUERY_ALL_PACKAGES,
- READ_CALENDAR,
- READ_CALL_LOG,
- READ_CONTACTS,
- READ_LOGS,
- READ_PHONE_NUMBERS,
- READ_PHONE_STATE,
- READ_PRECISE_PHONE_STATE,
- READ_VOICEMAIL,
- RECEIVE_WAP_PUSH,
- REQUEST_COMPANION_PROFILE_COMPUTER,
- REQUEST_COMPANION_PROFILE_WATCH,
- REQUEST_COMPANION_RUN_IN_BACKGROUND,
- REQUEST_COMPANION_SELF_MANAGED,
- REQUEST_COMPANION_START_FOREGROUND_
  SERVICES_FROM_BACKGROUND,
- REQUEST_COMPANION_USE_DATA_
  IN_BACKGROUND,
- REQUEST_DELETE_PACKAGES,
- REQUEST_INSTALL_PACKAGES,
- SCHEDULE_EXACT_ALARM,
- SET_ALARM,
- START_FOREGROUND_SERVICES_
  FROM_BACKGROUND,
- USE_SIP,
- UWB_RANGING,
- WRITE_CALENDAR,
- WRITE_CALL_LOG,
- WRITE_CONTACTS
- WRITE_VOICEMAIL

Medium Risk

- ACCESS_COARSE_LOCATION,
- ACCESS_MEDIA_LOCATION,
- ACCESS_NETWORK_STATE,
- ACCESS_WIFI_STATE,
- BIND_NFC_SERVICE,
- BIND_VPN_SERVICE,
- BLUETOOTH,
- BLUETOOTH_ADMIN,
- BROADCAST_SMS,
- CAPTURE_AUDIO_OUTPUT,
- INTERNET,
- MANAGE_DOCUMENTS,
- MANAGE_MEDIA,
- MANAGE_WIFI_INTERFACES,
- MANAGE_WIFI_NETWORK_SELECTION,
- NEARBY_WIFI_DEVICES,
- OVERRIDE_WIFI_CONFIG,
- READ_CELL_BROADCASTS,
- READ_MEDIA_AUDIO,
- READ_MEDIA_IMAGES,
- READ_MEDIA_VIDEO,
- READ_SMS,
- REBOOT,
- RECEIVE_MMS,
- RECEIVE_SMS,
- SEND_SMS,
- SMS_FINANCIAL_TRANSACTIONS,

High Risk

- ACCESS_BACKGROUND_LOCATION,
- ACCESS_LOCATION_EXTRA_COMMANDS,
- ACCESS_FINE_LOCATION,
- ACTIVITY_RECOGNITION,
- BLUETOOTH_ADVERTISE,
- BLUETOOTH_CONNECT,
- BLUETOOTH_SCAN,
- BLUETOOTH_PRIVILEGED,
- BODY_SENSORS,
- CAMERA,
- MANAGE_EXTERNAL_STORAGE,
- READ_EXTERNAL_STORAGE,
- RECORD_AUDIO,
- USE_FINGERPRINT,
- WRITE_EXTERNAL_STORAGE,

Figure 4.3: Risk permission groups are based on the extent of access to the data that users consider important.

```
                in CAMERA_PERMISSIONS -> PermissionGroup.Camera
                in NEARBY_DEVICES_PERMISSIONS -> PermissionGroup.
                    NearbyDevices
                in INTERNET_PERMISSIONS -> PermissionGroup.Internet
                else -> PermissionGroup.Other
```

```
            }
        } else {
            permissionInfo.group?.getPermissionGroup() ?: PermissionGroup
                .Other
        }
    }
}

fun Release.generatePermissionGroups(context: Context): Map<
    PermissionGroup, List<PermissionInfo>> {
    val packageManager = context.packageManager
    return permissions
        .asSequence().mapNotNull {
            try {
                packageManager.getPermissionInfo(it, 0)
            } catch (e: Exception) {
                null
            }
        }
        .groupBy(PackageItemResolver::getPermissionGroup)
}
```

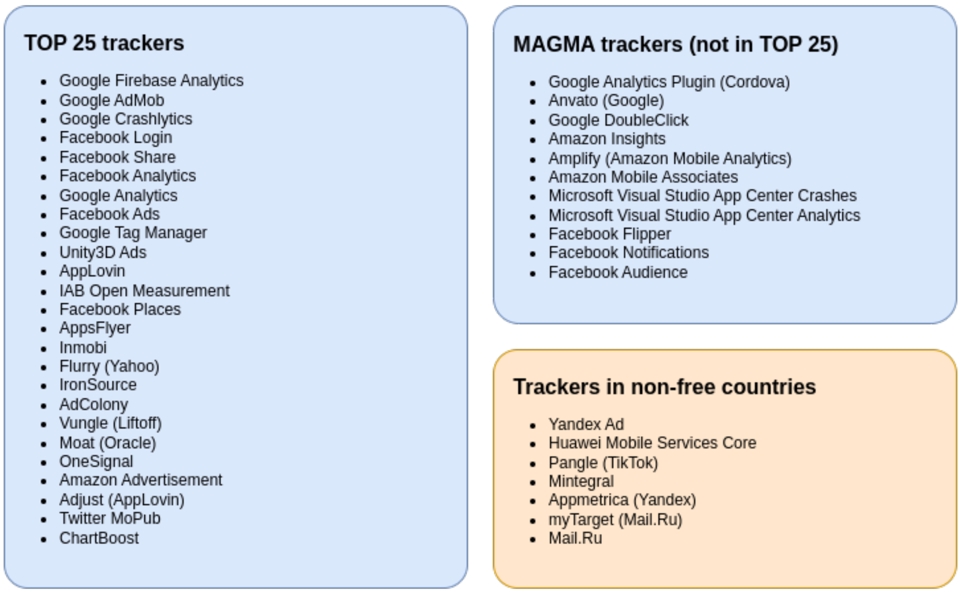
Since anti-features can be taken directly from the *Release* object, the tracker list was the last field of *PrivacyData* to be added. Just like with permissions, *Exodus Privacy*'s trackers were grouped into different types: Advertising, Profiling, Location, Analytics, Identification and Accident Reports. In addition, two special groups were created:

- The most widely used trackers, including the 25 most commonly used trackers based on *Exodus Privacy* statistics, as well as all trackers that belong to *MAGMA* corps [9].
- A group of all trackers located in non-free countries, such as China and Russia. This is especially (but not only) important for the citizens of these countries or for those who have contact with them.

After *permissions*, *antifeatures* and *trackers* were better defined and differentiated, it was possible to define a converter function that converts *PrivacyData* to *PrivacyNote*:

```
fun PrivacyData.toPrivacyNote(): PrivacyNote {
    val permissionsNote = 100 - permissions.values.flatten().sumOf {
        when (it.name) {
            in HIGH_RISK_PERMISSIONS -> 15
            in MEDIUM_RISK_PERMISSIONS -> 7
            in LOW_RISK_PERMISSIONS -> 3
```

[9]An acronym for Meta-Apple-Google-Microsoft-Amazon: A new acronym for the most valuable digital media and entertainment corporations. As pointed out by Sébastien Wilmet [47], this new acronym is meant to illustrate how dangerous these corporations are to competition and that governments must intervene.



TOP 25 trackers

- Google Firebase Analytics
- Google AdMob
- Google Crashlytics
- Facebook Login
- Facebook Share
- Facebook Analytics
- Google Analytics
- Facebook Ads
- Google Tag Manager
- Unity3D Ads
- AppLovin
- IAB Open Measurement
- Facebook Places
- AppsFlyer
- Inmobi
- Flurry (Yahoo)
- IronSource
- AdColony
- Vungle (Liftoff)
- Moat (Oracle)
- OneSignal
- Amazon Advertisement
- Adjust (AppLovin)
- Twitter MoPub
- ChartBoost

MAGMA trackers (not in TOP 25)

- Google Analytics Plugin (Cordova)
- Anvato (Google)
- Google DoubleClick
- Amazon Insights
- Amplify (Amazon Mobile Analytics)
- Amazon Mobile Associates
- Microsoft Visual Studio App Center Crashes
- Microsoft Visual Studio App Center Analytics
- Facebook Flipper
- Facebook Notifications
- Facebook Audience

Trackers in non-free countries

- Yandex Ad
- Huawei Mobile Services Core
- Pangle (TikTok)
- Mintegral
- Appmetrica (Yandex)
- myTarget (Mail.Ru)
- Mail.Ru

Figure 4.4: The special trackers sets.

```
            else -> 2
        }.toInt()
    }.coerceAtMost(100)
    val trackersNote = 100 - (trackers.sumOf {
        it.categories.sumOf(String::trackerCategoryNote) * it.key.
            trackerNoteMultiplicator
    }).coerceAtMost(100)
    val sourceType = SourceType(
        open = AntiFeature.NO_SOURCE_SINCE !in antiFeatures,
        free = !antiFeatures.any {
            it == AntiFeature.NON_FREE_NET ||
                    it == AntiFeature.NON_FREE_UPSTREAM
        },
        independent = !antiFeatures.any {
            it == AntiFeature.NON_FREE_DEP || it == AntiFeature.
                NON_FREE_ASSETS
        }
    )
    return PrivacyNote(
        permissionsNote,
        trackersNote,
        sourceType
    )
}

private val String.trackerCategoryNote: Int
    get() = when (this) {
        "Ads", "Profiling", "Location" -> 20
        "Analytics", "Identification" -> 10
        "Crash reporting" -> 5
        else -> 1
    }

private val Int.trackerNoteMultiplicator: Int
    get() = when (this) {
        in WIDESPREAD_TRACKERS,
        in NON_FREE_COUNTRIES_TRACKERS -> 2
        else -> 1
    }
```

4.2.3 Front-End

With the translation layer between the back-end and front-end implemented, the last part of the implementation left was writing the front-end. The entire UI is written using Jetpack Compose [10].

The two layers of the system have no common components. The first layer's UI, *Meter Icon* has two main components, the *Meter Icon Bar* and the *Source Code Icon*. The *Source Code Icon* was so far a simple icon in *Neo Store* that led to the source code of the app, and was redesigned to complement the privacy visualization design dynamically changing the icon based on *SourceType* of *PrivacyNote* so that it can indicate that the app is free, open-source, available-source or proprietary. While *Meter Icon Bar* UI can be divided into two blocks:

- Two blocks, each of which conveys the privacy score of trackers or permissions. Each of these blocks has an icon representing the subject and an indicator that shows the corresponding score by highlighting the corresponding color. Tapping the indicator brings up a tooltip with a description of the score, e.g. for trackers: red indicates "Is it an app or spyware?", while green indicates "Minimum to no trackers".

- An icon that opens the Privacy Panel, the second layer of the system, when clicked.

The *Privacy Panel*, the user interface based on the design of *Descriptive Label* consists of several expandable privacy cards, each focusing on a specific topic:

- Permissions for physical data
- Permissions for identification data
- Other permissions
- Trackers

[10]Jetpack Compose is a modern declarative UI toolkit for Android using Kotlin.

- Source code
- Other anti-features

The permission data cards show the permissions requested by the app grouped into expandable blocks of the respective permission group. Thus, it is a direct translation of the permission map that filters only the relevant permission groups for each privacy card. The tracker map is similar in structure and groups trackers based on their function: advertising, analytics, identification, location tracking, profiling, or crash reporting. As for the source code privacy map, it contains two or three blocks based on *SourceInfo*, depending on the *SourceType* of the *PrivacyNote*:

- sourceType.open: Shows an open source block if true, otherwise a proprietary block.
- sourceType.free: Shows a copyleft block if true, otherwise a copyright block.
- sourceType.independent: If it is false, it shows a dependency on non-free software block, otherwise it shows nothing.

All privacy cards are expanded by default, except for "Other permissions" and "Other anti-features". All individual elements (trackers, permissions and source information) contain a short description for better understanding, which is either dynamically extracted from the system or defined in the respective objects:

```kotlin
// In Privacy.kt
open class TrackersGroup(
    @StringRes labelId: Int,
    @StringRes descriptionId: Int,
    icon: ImageVector,
)

open class SourceInfo(
    @StringRes labelId: Int,
    @StringRes descriptionId: Int,
    icon: ImageVector,
)

// In Utils.kt
fun List<PermissionInfo>.getLabelsAndDescriptions(context: Context):
    List<String> {
    val localCache = PackageItemResolver.LocalCache()

    return map { permission ->
        val labelFromPackage =
            PackageItemResolver.loadLabel(context, localCache, permission
                )
        val label = labelFromPackage ?: run {
            val prefixes =
                listOf("android.permission.", "com.android.browser.
                    permission.")
            prefixes.find { permission.name.startsWith(it) }?.let { it ->
                val transform = permission.name.substring(it.length)
                if (transform.matches("[A-Z_]+".toRegex())) {
                    transform.split("_")
                        .joinToString(separator = " ") { it.lowercase(
                            Locale.US) }
                } else {
                    null
                }
            }
        }
        val description =
            PackageItemResolver.loadDescription(context, localCache,
                permission)
                ?.nullIfEmpty()?.let { if (it == permission.name) null
                    else it }

        if (description.isNullOrEmpty()) (label ?: permission.name).
            toString()
        else "${label ?: permission.name}: $description"
    }
}
```

Finally, each of the privacy cards displays an action button if the app under review is installed. For permissions, this would open the app's settings page so the user can revoke the desired permissions, while for trackers, there is an option to open (or install, if not already installed) TrackerControl [28] where the user can block specific trackers. Two features for the tracker elements have been added on the suggestion of an expert:

- one click opens the page of a tracker.
- one long click copies its signature.

Concluding, it has proven to be relatively straightforward in the implementation phase to concretize the design concept in an architecture diagram and then build the back-end along these lines, taking into account the classes that already exist. This is largely due to the fact that, as the *Neo Store* developer, I have a clear idea of what classes and functions already exist and have the freedom to rewrite the logic of existing functions, e.g. `generatePermissionGroups()`. Otherwise, a deeper analysis of *Neo Store*'s code base and a pre-coordination with the project lead would have been required before starting the implementation.

5 Evaluation

5.1 Participant Recruitment

After the release of app version 0.9.11 with the newly implemented interactive privacy visualization design, I called upon the community in the Telegram group to participate in an online survey to evaluate the design. The survey focuses on collecting qualitative feedback and therefore consists mainly of open-ended questions (Appendix E: Evaluation Survey Questions).

The turnout ended up being 16 participants (Appendix F: Evaluation Survey Result), far below my expectations considering that the group has over 700 accounts, but that could also be due to the qualitative nature of the survey (and thus the relatively lengthy time required) and lack an external incentive such as a reward. Other than that, some of the questions may have been linguistically misunderstood and therefore answered inappropriately.

Although the number of participants is not significantly representative, it is reasonable to assume that the result is representative given the consistency of opinions with those of the experts. However, it should also be pointed out that the community itself is not so generalizable, which is related to the fact that the group is mainly male (15/16), with IT background (11/16), and privacy-sensitive (15/16) [1].

5.2 Results

When asked how they interpreted the new visual element in the app, ten of the 16 participants interpreted it correctly, either explicitly as an indicator of privacy-friendliness or implicitly as an indicator of whether they should double-check an app before installing it. The other six had a mix of "*looks good*," "*needs a pre-indicator*"(P14) and "*not much unless there are little to no trackers and little to no permissions*"(P10). The last answer is related to the fact that the participant P10 was not aware of the *Privacy Panel*.

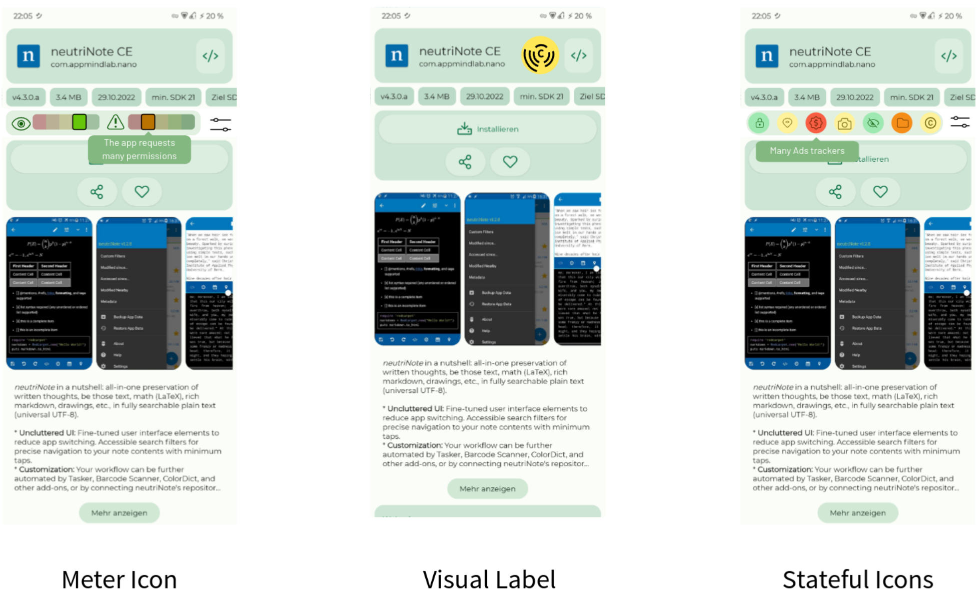
Answering the question about expectations for an on-click, seven participants expected more details about trackers and permissions and another three expected an explanation of the results shown. The remaining responses included "*I didn't think it was clickable*"(P02) and "*I would see some kind of pie chart*"(P03). All participants indicated that they had never seen a similar design or were unfamiliar with it, except for the participant P05 who associated the design with the textual listing of trackers in *Aurora Store*.

[1] The remaining person, didn't answer the question. Applies to gender and privacy sensitiveness.

Asked why they would say the design conveys privacy-related information well, ten pointed to the simplicity and intuitiveness of the design, while two gave no answer and the rest gave a mix of “I like it” and “for providing privacy information, especially…”. While when asked why they would say it is not good for conveying privacy-related data, five said it does, another five said the various icons are not that intuitive, while two wanted more sensitivity about what permissions are really needed by apps and what is excessive. However, asked if such a design could be useful in other app stores, 15 answered with a resounding yes, while one noted that it could be different in other app stores in terms of the information presented and the use of more panels.

In response to the question of which of the three designs examined in this study they preferred, nine chose *Meter Icon* and seven chose *Stateful Icons*, while *Visual Label* received none. Here are some quotes from the responses to put things in context:

- “*First of all: visual [l]abel is horrible to understand, for me personally a no go…*”(P03)
- “*Stateful Icons gives much info without the need to click and go a seperate page to the permissions it asking.*”(P16)
- “*In case of Stateful Icons design, it is not possible to show icon of every permission requested by the app. In Meter Icon design, it shows both level of tracking & permissions. And extensive details can be seen in Privacy Panel page.*”(P05)



neutriNote in a nutshell: all-in-one preservation of written thoughts, be those text, math (LaTeX), rich markdown, drawings, etc., in fully searchable plain text (universal UTF-8).

* **Uncluttered UI:** Fine-tuned user interface elements to reduce app switching. Accessible search filters for precise navigation to your note contents with minimum taps.
* **Customization:** Your workflow can be further automated by Tasker, Barcode Scanner, ColorDict, and other add-ons, or by connecting neutriNote's repositor...

Figure 5.1: The three final design prototypes as shown in the community survey.

Comparison along other variables mostly shows a similar result, for instance, the results for participants with and without a formal IT background are six and three for *Meter Icon* and five and two for *Stateful Icons*, respectively. However, when related to privacy awareness, it can be seen that users who are specifically concerned with large enterprises (e.g., *MAGMA*) have a clear bias towards *Stateful Icons* with five participants for *Stateful Icons* and one for *Meter Icon*, while *Meter Icon* is preferred by privacy advocates (five to two) and participants concerned with government surveillance (two to null).

In the discussion of the privacy panel, 13 participants felt it was at least somewhat good, while the remaining three were not sure or had no opinion. Arguments ranged from "*a detailed view*" to "*looking good*" to "*too much information*". Three users pointed out to problems finding and navigating the layout, while the participant who thought it had too much information wished it had more hints about what is important and what is not.

To sum up, the addition of the interactive privacy visualization is very much appreciated by most users, this is especially true for the *Privacy Panel*. As for the different possible designs, none of the participants considered *Visual Label* a good design, while the *Meter Icon* is preferred by a narrow margin over *Stateful Icons*, and there are good arguments in favor of each of them. In addition, some participants reported minor navigation issues and suggested generally feasible improvements.

6 Discussion

In this study, using IDEO's human-centered design guidelines [27], we examined several designs in the privacy visualization literature that appeared to be well-developed but were generally untested in practice [36, 35]. In a next step, four distinguishable designs were identified and framed: *Meter Icon*, *Stateful Icons*, *Visual Label*, and *Descriptive Label*. This was followed by a design process that included interviews with community experts and several prototyping iterations, based on which a layered design based on *Meter Icon* and *Descriptive Label* was implemented in *Neo Store*, a modern F-Droid client [31]. In the last instance, the implemented design was evaluated by the community through a survey and it is safe to say that the design is a welcome addition and the experts' design choice matches well with the community preferences.

6.1 Recommendations

Based on the study results and learnings during the process, a set of recommendations was formulated for practitioners interested in designing interactive privacy visualizations:

6.1.1 Two Layered Privacy Visualization Rocks

One of the main recommendations in the literature is to make privacy visualization a gateway to more information [41]. In the developed design, two different privacy visualization designs were used at two different interconnected levels, so that the simple privacy visualization of Meter Icon is a gateway to more information, whose complexity is reduced by utilizing Descriptive Label, an informative privacy visualization design.

The aforementioned combination was very welcomed in the community group and praised in the rating survey. The biggest highlight for users was the ability to easily see if there was a need to educate themselves more about the privacy-related features of the app, and if so, the privacy panel mostly provided all the information needed. This was also reflected in the updated User Journey, which better portrays the two moments of reflection (see Figure 6.1)

6.1.2 Intuitiveness is the Queen/King

Intuitiveness was (in)arguably the factor that gave *Meter Icon* the advantage over *Stateful Icons*. This is consistent with the various recommendations in the literature [41, 21, 35, 19] and was the main argument made by survey

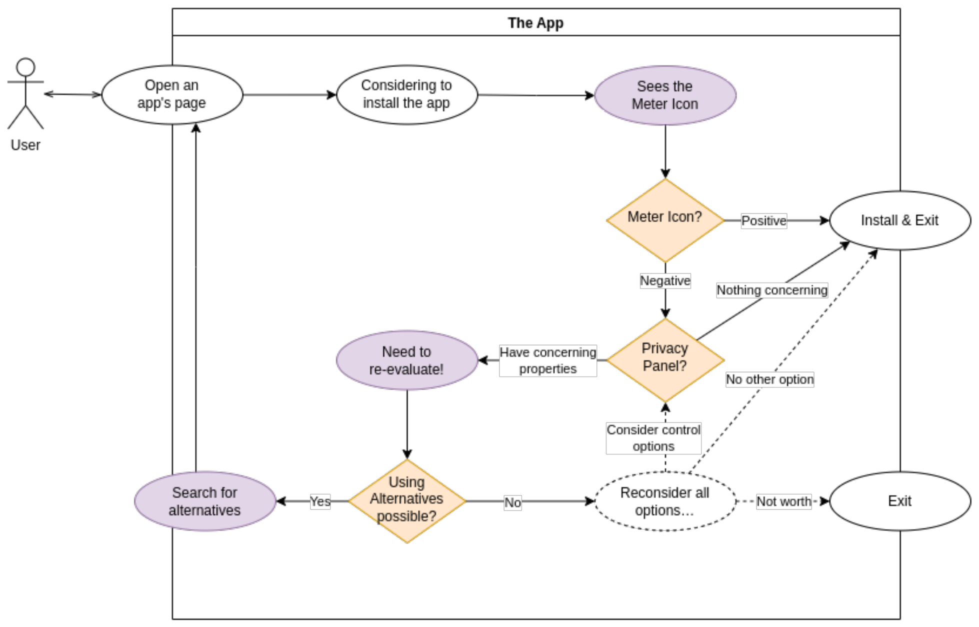


Figure 6.1: The final simplified user journey (colored = additions to the existing system)

participants when asked which design was preferable or good in communicating privacy-related data.

While the quantitative difference in votes is not significant, the arguments made by participants in favor of *Meter Icon*'s simplicity cover various aspects of design (e.g., accessibility, comprehensibility), which is also consistent with the NN Group's UX slogans "Keep it Simple," "Less is More," and "Brevity = Brilliance"[32]. However, compared to an argument for *Stateful Icons* that merely refers to showing more information at a glance, this benefit was disputed by some participants, who pointed out that it is impossible to show all relevant elements, as this depends heavily on personal preferences (Appendix F: Evaluation Survey Result).

6.2 Limitations

Although this work provides a practical example of human-centered design and implementation of an interactive privacy visualization, there are some limitations worth mentioning. One of these that may be clear for software developers is the lack of unit tests, which is owed to the focus of this work on the design process and general complexity of unit testing Android UIs, leaving it as a point to handle in future work.

Another limitation faced in the design was reaching certain demographic groups, especially genders other than (cis)men and experts from continents other than Europe. The first problem is related to the fact that men are clearly in the majority in the community. The second problem seems to be due to two factors. The first is that the majority of users are not native English speakers, and although I offered to conduct the interview in German or Arabic where desired, this did not convince more experts. The second factor is related to a portion of the users living in so-called non-free countries (e.g., China), who may be more reluctant to speak out because their safety is tied to their complete anonymity.

Apart from this latter limitation, the expert group had quite diverse backgrounds and expertise. For example, Omar is politically active in addition to his social work, Luka works in telecommunications infrastructure and is active in several *FOSS* communities, Piere is an executive in an international hardware company, and Warren works in media management [1].

Limitations of the draft include that the control options are limited to what the operating system allows for a non-privileged application and what limitations the Android permission system itself has. Another limitation of the design, also mentioned in the survey, was that the permissions assessment does not consider what permissions an app really needs for its functions and what are excessive permissions. Unfortunately, there were no resources or tools to improve this at the time of implementation.

A final general limitation of the study is time constraints, as I would have liked to have conducted more interviews to explore more aspects, given the community more time with the design before they had to respond to the survey, and engaged the community in another iteration concerning control options before implementation. In this regard, it is safe to say that it is good to devote enough time in the inspiration phase of an HCD process. However, given the time frame for the bachelor thesis, this was not possible.

[1]All names used here are pseudonymized names.

7 Conclusion & Future Work

Existing privacy visualization designs have varying degrees of detail and simplicity, and therefore work differently well or poorly for different user groups [12]. Unfortunately, most of the existing privacy visualization designs have not been tested in real-world applications, if they have been tested at all [36, 35]. In addition, most designs to date have been conceived as stand-alone systems rather than thought of in combinations. This study addresses both issues by applying a human-centered design process to develop a community-tested two-layer interactive privacy visualization design in *Neo Store*, an open-source app store client.

The study shows the importance of intuitiveness and simplicity for good privacy visualization designs and a general rejection of non-transparent or complex systems. This is most evident in the fact that experts and the community rejected the design of *Visual Label* as too complex, as well as the general dissatisfaction with traditional privacy policy visualizations. At the same time, we see only a small advantage for designs with a clear focus on simplicity (like *Meter Icon*) over others that try to balance simplicity with richness of content (like *Stateful Icons*).

The work focused mainly on the design process and the use of available resources to achieve a satisfactory result for users and open up more possibilities. That said, the evaluation by the community shows that there is a desire to obtain information about the relevance of each permission to the actual functionality of the apps and to learn which permissions are excessive. A possible solution could be to develop an artificial intelligence model that identifies these groups of permissions and provides the results via an open API, or such a system could also be integrated into the *Exodus Privacy* analysis workflow.

Another future possibility to use the result of this study is the adaptation of the developed design by the main project F-Droid to enable further development with a larger community and to push for a standardization of privacy visualization in the field of open source app stores. On the theoretical side, there is an opportunity to attempt the development of a *Privacy by Design* using a similar HCD process and lessons learned in this study. It has not been given special consideration in this study because it deals primarily with the views of developers in an area that clearly should be, but is not, regulated by policy.

Bibliography

[1] Anti-features | f-droid - free and open source android app repository. `https://f-droid.org/docs/Anti-Features/`.

[2] App-check. `https://appcheck.mobilsicher.de/`.

[3] Exodus privacy. `https://exodus-privacy.eu.org/en/`.

[4] F-droid. `https://f-droid.org/`.

[5] General data protection regulation (GDPR) compliance guidelines. `https://gdpr.eu/`.

[6] machiav3lli on github. `https://github.com/machiav3lli`.

[7] Privacy label. `https://www.privacylabel.org`.

[8] Privacy rating. `https://github.com/RNDRnl/privacy-rating`.

[9] The Open Definition. `https://opendefinition.org`.

[10] Alsoubai, A., Ghaiumy Anaraky, R., Li, Y., Page, X., Knijnenburg, B., and Wisniewski, P. J. Permission vs. app limiters: Profiling smartphone users to understand differing strategies for mobile privacy management. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, Association for Computing Machinery, pp. 1–18.

[11] Apple. App store - privacy nutrition labels. `https://www.apple.com/privacy/labels/`.

[12] Barth, S., Ionita, D., and Hartel, P. Understanding online privacy—a systematic review of privacy visualizations and privacy by design guidelines. *ACM Comput. Surv. 55*, 3 (feb 2022).

[13] Becker, M. Privacy in the digital age: comparing and contrasting individual versus social approaches towards privacy. 307–317.

[14] Benton, K., Camp, L. J., and Garg, V. Studying the effectiveness of android application permissions requests. In *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pp. 291–296.

[15] Bock, S., and Momen, N. Nudging the user with privacy indicator: A study on the app selection behavior of the user. In *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society*, no. 60. Association for Computing Machinery, pp. 1–12.

[16] Bodoni, S. Europe's data law is broken, departing privacy chief warns.

[17] Brignull, H. Dark patterns: Deception vs. honesty in UI design.

[18] Chen, Y., Zha, M., Zhang, N., Xu, D., Zhao, Q., Feng, X., Yuan, K., Suya, F., Tian, Y., Chen, K., Wang, X., and Zou, W. Demystifying hidden privacy settings in mobile apps. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 570–586. ISSN: 2375-1207.

[19] Edwards, L., and Abel, W. The use of privacy icons and standard contract terms for generating consumer trust and confidence in digital services.

[20] Fox, G., Lynn, T., and Rosati, P. Enhancing consumer perceptions of privacy and trust: a GDPR label perspective. 181–204. Publisher: Emerald Publishing Limited.

[21] Franke, G., Clever, T., van Dijk, W., Raider, J., and de Jong, R. PRIVACY LABEL — part i: The privacy illusion.

[22] Frik, A., Kim, J., Sanchez, J. R., and Ma, J. Users' expectations about and use of smartphone privacy and security settings. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, Association for Computing Machinery, pp. 1–24.

[23] Google. Play store - data safety section. `https://blog.google/products/google-play/data-safety/`.

[24] Gunawan, J., Pradeep, A., Choffnes, D., Hartzog, W., and Wilson, C. A comparative study of dark patterns across web and mobile modalities. 377:1–377:29.

[25] Habib, H., Zou, Y., Yao, Y., Acquisti, A., Cranor, L., Reidenberg, J., Sadeh, N., and Schaub, F. Toggles, dollar signs, and triangles: How to (in)effectively convey privacy choices with icons and link texts. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021), CHI '21, Association for Computing Machinery.

[26] Haninger, K., and Thompson, K. M. Content and Ratings of Teen-Rated Video Games. *JAMA 291*, 7 (02 2004), 856–865.

[27] IDEO. *The Field Guide to Human-Centered Design*, 1st. ed ed. IDEO.org.

[28] Kollnig, K., and Shadbolt, N. TrackerControl. `https://github.com/TrackerControl/tracker-control-android`, 2019.

[29] Lane, K., Harrington, L., and Ryan, P. Evaluating the impact of energy labelling and MEPS – a retrospective look at the case of refrigerators in the UK and australia. 8.

[30] Martin, K., and Shilton, K. Putting mobile application privacy in context: An empirical study of user privacy expectations for mobile devices. *The Information Society 32*, 3 (2016), 200–216.

[31] NeoApplications. Neo store. `https://github.com/NeoApplications/Neo-Store`, 2022.

[32] Nielsen Norman Group. UX & usability articles. `https://www.nngroup.com/articles`.

[33] Pollach, I. A typology of communicative strategies in online privacy policies: Ethics, power and informed consent. 221–235.

[34] Raskin, A. Privacy icons - MozillaWiki.

[35] Reinhardt, D., Borchard, J., and Hurtienne, J. Visual interactive privacy policy: The better choice? In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, Association for Computing Machinery, pp. 1–12.

[36] Rossi, A., and Palmirani, M. DaPIS: An ontology-based data protection icon set. 181–195. Publisher: IOS Press.

[37] Runge, J., Wentzel, D., Huh, J. Y., and Chaney, A. "dark patterns" in online services: a motivating study and agenda for future research.

[38] Sanchez, M. C., Brown, R. E., Webber, C., and Homan, G. K. Savings estimates for the united states environmental protection agency's ENERGY STAR voluntary product labeling program. 2098–2108.

[39] Simple Mobile Tools. Simple mobile tools android apps website. `https://www.simplemobiletools.com`.

[40] Stallmann, R. M. Why "free software" is better than "open source" - GNU project - free software foundation, 1998.

[41] Stöver, A., Gerber, N., Kaushik, S., Mühlhäuser, M., and Marky, K. Investigating simple privacy indicators for supporting users when installing new mobile apps. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021), CHI EA '21, Association for Computing Machinery.

[42] Taddei, S., and Contena, B. Privacy, trust and control: Which relationships with online self-disclosure? 821–826.

[43] Tucker, C. E. Social networks, personalized advertising, and privacy controls. 546–562. Publisher: American Marketing Association.

[44] Van Kleek, M., Liccardi, I., Binns, R., Zhao, J., Weitzner, D. J., and Shadbolt, N. Better the devil you know: Exposing the data sharing practices of smartphone apps. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (2017), ACM, pp. 5208–5220.

[45] Van Ooijen, I., and Vrabec, H. U. Does the GDPR enhance consumers' control over personal data? an analysis from a behavioural perspective. 91–107.

[46] Waldman, A. E. Data protection by design? a critique of article 25 of the GDPR. 147–167.

[47] Wilmet, S. GAFAM to MAGMA.

Appendix

Appendix A: Milestones

M1	**Milestone – Analysis**
Due date	2022-10-30 (Week 2)
Tasks	Identify and analyze literature contributions on privacy visualizations from recent years to identify the underpinnings of existing designs, what baseline data is needed for good privacy visualization and what are possible sources to provide this data.
Outcome	A streamlined summaries of the individual literature contributions. Initial listing of data sets usable for privacy visualizations. A list of recommendations for data-visualization design decisions.
Goal	Identify relevant designs from the literature, usable data for visualizations and its requirements.

M2	**Milestone – Initial Prototypes**
Due date	2022-11-06 (Week 3)
Tasks	Create lo-fi prototypes of the visualization system.
Outcome	Interactive design sketches, each following a specific privacy visualization model. Questions to be asked in the semi-structured expert interviews and in the final evaluation survey.
Goal	Design and creation of the first prototypes and selection of questions for the expert interviews and final evaluation.

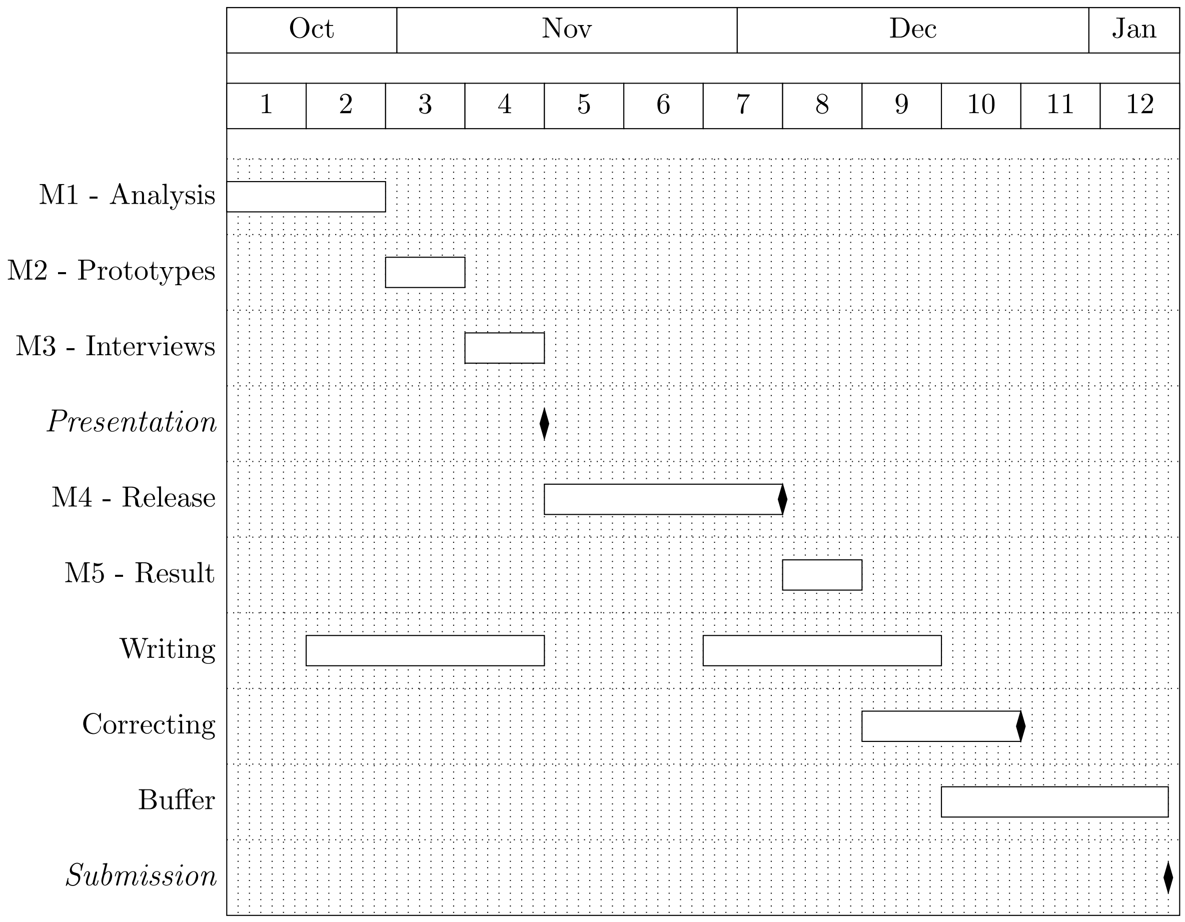
M3	**Milestone – Expert Interviews**
Due date	2022-11-13 (Week 4)
Tasks	Evaluate prototypes in a number of expert interviews with community members.
Outcome	Evaluation of the prototypes with positive and negative aspects and suggestions for improvement.
Goal	Better understand user preferences and select the design with the greatest potential.

M4	**Milestone – First Release**
Due date	2022-12-04 (Week 7)
Tasks	Adaptation of the selected prototype based on the feedback from the experts and subsequent implementation of the developed design in the Neo Store.
Outcome	The final design of the privacy visualization design.
	A first community/test release with the developed design.
Goal	Implementation and release of the developed system.

M5	**Milestone – Result, Feedback & Reflection**
Due date	2022-12-11 (Week 8)
Tasks	Survey the community to evaluate the system.
Outcome	Detailed analysis of the survey results.
Goal	Measure the acceptability of the developed design and identify possible improvements.

M6	**Milestone – Finalize**
Due date	2023-01-08 (Week 12)
Tasks	Finish documentation, reflect on the process and submit the bachelor thesis.
Outcome	Publicly accessible paper (to be discussed).
Goal	Public and documented release of the developed system.

Appendix B: Planned Timeline



Appendix C: Semi-structured Interview Questions

Meter Icon

- What benefits do you see in this design?
- What shortcomings do you see in this design?
- Do you have any suggestions for improvement?
- How should a system based on this design best be placed?

Visual Label

- What benefits do you see in this design?
- What shortcomings do you see in this design?
- Do you have any suggestions for improvement?
- How should a system based on this design best be placed?

State(ful) Icons

- What benefits do you see in this design?
- What shortcomings do you see in this design?
- Do you have any suggestions for improvement?
- How should a system based on this design best be placed?

Descriptive Label

- Wether this did help you to investigate the privacy level of the app?
- Would you add more information or icons to this visualization?
- Would you remove any of the information from this visualization?
- How would you categorize (move, remove, or add) the information differently?
- Would you suggest other visualization formats?

Personal

- What is your age?
  - ☐ Under 18
  - ☐ 18-24
  - ☐ 25-34
  - ☐ 35-44
  - ☐ 45-54
  - ☐ 55 or older
  - ☐ Prefer not to answer
- What is your gender?
  - ☐ Women
  - ☐ Men
  - ☐ Non-binary
  - ☐ Prefer to self describe:
  - ☐ Prefer not to answer
- What are your areas of expertise (work, education, and civic engagement)?
- Any general feedback on the survey and/or the project?

Appendix D: Study Informed Consent

Study Informed Consent

1. Basic information

Name of interviewee(s):	____________________________
Interview date / place:	__.11.2022
Research project title:	Privacy Visualizations: Introducing an interactive visualization of privacy indicators based on Exodus Privacy to F-Droid
Investigating researcher:	Antonios Hazim

Thank you for agreeing to be interviewed as part of the above research project. Research ethics and data protection regulations (i.e., GDPR[1]) require that interviewees explicitly agree to being interviewed and how the information contained in their interview will be used. This information and consent form is necessary for us to ensure that you understand the purpose of your involvement and that you agree to the conditions of your participation.

2. Context of the research project

The project aims at investigating the implementation of a privacy visualization system in smartphone app stores, more specifically in F-Droid's client, Neo Store.
It is carried out at the Human- Centered Computing research group at the Freie Universität Berlin.

2.1. Methodology of the research project

The design process is based on "Human Centered Design", a method that puts people at the center. This includes three phases:

1. Inspiration: review of literature on the relevant topics and analysis of existing sources or solutions.
2. Ideation: In this work, the ideation process includes two iterations.
    1. Determine What to Prototype: the first step is to design three prototypes based on different designs for privacy visualizations proposed in the literature, and then interview a handful of experts.
    2. Integrate of Feedback and Iterate: the second iteration involves further development of a selected prototype based on the feedback from the experts.
3. Implementation: implementation of the developed system in the Neo Store.

The work phase will conclude with a community survey to obtain feedback on the developed design, critique the work, and outline future research that builds on this work.

1 General Data Protection Regulation, legal text available at https://gdpr-info.eu/

2.2. Your role in the study

You will be interviewed as an expert to provide feedback on the content and design of the initial prototypes. Some follow-up questions may also arise during the selection process and during the later consideration of the selected design. If so, I will contact you by your preferred way to request this. Also, it would be appreciated if you would provide explicit feedback during the final evaluation.

2.3. Interview procedure

The semi-structured interview will take about 60 min. The discussion may be audiotaped and a transcript of the interview or extracts of it will be produced as a text file. Additionally, photos or short videos may be taken documenting work-related aspects (e.g., screenshots), following separate permission by yourself.
It is not anticipated that there are any risks associated with your participation, but you have the right to stop the interview or withdraw from the research at any time. You are free to decline to answer any questions you don't wish to, or to stop the interview at any time.

3. Data use and publishing

Elements of the transcript and visual material may be quoted and displayed directly or indirectly in academic articles, books, conference presentations, and blog posts.

4. Compensation

You will not be paid for taking part in this study. It is hoped that the research will provide benefits to the Free and Open-Source Software community and more specifically F-Droid.

5. Confidentiality

Your study data will be handled confidentially. Access to the interview transcript will be limited to me and academic colleagues with whom I collaborate as part of my research. I will anonymize individual names in all versions of the transcripts and quotations that are shared with others.
However, due to public visibility of the interviewees' work (Github accounts, publications, presentations, etc.), there is a chance that confidentiality is compromised. I will take precautions to minimize this risk such as anonymization of individual names, places, institutions, upon request by the interviewees.

6. Data protection and storage

The audio recording, transcript and visual material will be archived only locally on my device. When the research is completed, I may save the tapes and notes for use in future research carried out by myself.

7. Data Subject Rights

According to GDPR, you have the following rights as soon as your personal data has been produced:

- Right to gain access to the stored personal data (Article 15 GDPR).
- Right to rectification if data concerning your person is incorrect or incomplete (Article 16 GDPR).
- Right to deletion of data concerning your person, insofar as one of the legal requirements applies and there is no legal exception to the contrary (Article 17 GDPR).

- Right to restriction of processing, in particular if the correctness of data is disputed, if one of the reasons stated by law intervenes, in particular at your request also instead of deletion of the data (Article 18 GDPR).
- Right to data portability. You have the right to demand access to all personal data stored about you in a structured, commonly used, and machine-readable format and have the right to transmit those data to another controller without hindrance from the controller to which the personal data have been provided (Article 20 GDPR).
- Right to lodge a complaint with a supervisory authority. The responsible supervisory authority can be any data protection supervisory authority (Datenschutzaufsichtsbehörde) (Article 77 GDPR).
- Right to withdraw consent (Article 7 GDPR)

8. Contact

8.1 Responsible Person

If you have any questions about this research, please feel free to contact me.

I can be reached at
Antonios Hazim
antonios.hazim@fu-berlin.org

You can also contact my supervisor:
Peter Sörries
peter.soerries@fu-berlin.de

Human-Centered Computing
Institute for Computer Science, Freie Universität Berlin
Königin-Luise-Str. 24/26, 14195 Berlin

8.2 Data Protection Officer

If you are concerned about this research or if you have concerns about how it is being conducted, you can contact the data protection officer at Freie Universität Berlin. You can reach him or her at:

Dr. Karsten Kinast, LLM
Am Hohenzollernring 54
50672 Köln
E-Mail: leidinger@kinast.eu

Declaration of consent

By signing this form, I voluntarily consent to take part in the research according to the conditions stated in the information sheet, which I have read thoroughly (pages 1 – 3 of this document). I understand that I can refuse to answer questions and I can withdraw from the study at any time, without having to give a reason.

Additionally, I declare that

☐ I can be quoted with my real name.

☐ I wish to review the (anonymized) quotations pertaining to my participation in the interview before they become part of a publication.

☐ I wish that the original video and audio recordings is deleted at the end of the project period (09.01.2023).

☐ I agree that written transcripts based on the video and audio recordings of the semi-structured interview are saved on internal servers of Freie Universität as an Appendix of Antonios Hazim's Thesis.

Any variation of the conditions above will only occur with your further explicit approval.

____

Participants Signature

____

Researchers Signature

________________________, _____________

Place, date

This information sheet and consent form are licensed under CC BY-SA 4.0, enabling further re-use and adaptation, while requiring attribution.

Appendix E: Evaluation Survey Questions

- What, if anything, does the new privacy visualization design communicate to you? Please describe your thoughts briefly.
- Have you ever seen similar design in an app store before?
  - ☐ Yes. In what context?
  - ☐ No
  - ☐ I am not sure
- Imagine if you saw a similar design while navigating an app store. Please describe what do you think would happen if you clicked on this symbol?
- Please explain why you find the design above good in communicating privacy-related info on an app.
- Please describe why this design might not align with your needs for communicating privacy in apps.
- Do you think this design might be useful in other app stores?
- Which of the designs supplemented in the accompanying file do you think best conveys privacy-related info on an app in Neo Store?
  - ☐ Meter Icon
  - ☐ Visual Label
  - ☐ Stateful Icons
- Why?
- How do you evaluate the new Privacy Panel (the page shown when clicking the gear icon beside the privacy visualization)?
  - ☐ Just Bad
  - ☐ Somehow Bad
  - ☐ I don't really know
  - ☐ Somehow Good
  - ☐ Really Good
- Why?
- Which of the following best describes your relation to privacy?
  - ☐ I'm generally not concerned about privacy.
  - ☐ I'm specifically concerned with state surveilance.

- ☐ I'm specifically concerned with big corps (e.g. MAGMA) control over our technologies.
  - ☐ I'm a privacy advocate, trying to strengthen privacy awareness.
  - ☐ Prefer not to answer.
- What is your age?
  - ☐ Under 18
  - ☐ 18-24
  - ☐ 25-34
  - ☐ 35-44
  - ☐ 45-54
  - ☐ 55 or older
  - ☐ Prefer not to answer
- What is your gender?
  - ☐ Women
  - ☐ Men
  - ☐ Non-binary
  - ☐ Answer not included here
  - ☐ Prefer not to answer
- Which of the following best describes your educational background or job feld?
  - ☐ I have an education in, or work in, the field of computer science, computer engineering or IT.
  - ☐ I do not have an education in, or work in, the feld of computer science, computer engineering or IT.
  - ☐ Prefer not to answer
- If you have any feedback or other comments on the survey and/or the project, please write it here.

Appendix F: Evaluation Survey Result

Link: https://box.fu-berlin.de/s/MdjfB8bmSzRAwkD
Password: geduldig36Grade68!Trust

Appendix G: Screen-record of the Final Design

Link: https://box.fu-berlin.de/s/iPxsnFR2X24w5R4
Password: geduldig36Grade68!Trust