Democratic and Popular Republic of Algeria

Ministry of Higher Education and Scientific Research

Houari Boumediene University of Science and Technology

Faculty of Computer Science
Department of Computer Systems

Master

Speciality
Computer Systems Security (SSI)

Theme

Design and implementation of a parental control tool for monitoring and controlling social media – WhatsApp's case –

Supervised by :

Mr BERBAR Ahmed

Presented by :

BOUKHEMIA Abdelrafik

FAHAM Billel

Supported on : 29/06/2025

Before the jury composed of:

Mrs BOUYAKOUB Samia	President
Mrs CHIKHAOUI Amina	Membre

Pair N° : 013 / 2025

Acknowledgements

We begin by extending our deepest gratitude and praise to Allah (Subhanu wa Ta'ala), for His grace, mercy, and the strength He bestowed upon us for the completion of this work.

Our sincere thanks then go to our esteemed supervisor, Mr. Ahmed BERBAR, for his patience, invaluable guidance, and continuous support, which have been of immense help throughout this journey.

We also express our heartfelt appreciation to the jury members for graciously dedicating their time and expertise to the evaluation of this thesis.

Finally, we would like to thank all those who, directly or indirectly, contributed and offered their support to this project.

Abstract

This thesis analyzes the Android environment and WhatsApp's functionality to develop a parental control solution. It addresses the data access challenges posed by Android's security measures and WhatsApp's encryption, rendering traditional non-rooted monitoring methods ineffective. The proposed solution leverages WhatsApp's "Linked Devices" feature for integrated and secure monitoring. The client-server architecture is detailed, including the target application on the child's device, a central server, and a parental dashboard. Features encompass access to messages, call history, audio/video recordings, contacts, shared media, app usage time, location, contact/group blocking, WhatsApp access blocking, and keyword-based alerts. This work demonstrates the technical feasibility of an ethical and reliable monitoring tool without requiring rooting, offering enhanced protection against digital risks for children.

Keywords: Android, WhatsApp, Parental Control, Data Monitoring

Résumé

Ce mémoire explore l'environnement Android et le fonctionnement de WhatsApp pour développer une solution de contrôle parental. Il analyse les défis d'accès aux données imposés par les mesures de sécurité d'Android et par le chiffrement de WhatsApp, rendant inefficaces les méthodes traditionnelles de surveillance sans root. La solution proposée exploite la fonctionnalité "Appareils connectés" de WhatsApp pour un monitoring intégré et sécurisé. L'architecture client-serveur est détaillée, incluant l'application cible sur le téléphone de l'enfant, un serveur central, et un tableau de bord parental. Les fonctionnalités couvrent l'accès aux messages, à l'historique des appels, aux enregistrements audio/vidéo, aux contacts, aux médias échangés, au temps passé sur l'application, à la localisation, au blocage de contacts/groupes, au blocage de l'accès à WhatsApp, et aux alertes par mots-clés. Ce travail démontre la faisabilité technique d'un outil de surveillance éthique et fiable sans nécessiter de root, offrant une protection accrue contre les risques numériques pour les enfants.

Mots Clés : Android, WhatsApp, Contrôle Parental, Surveillance des Données

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

ACID	Atomicity, Consistency, Isolation, and Durability
ACK	Android Common Kernel
AES	Advanced Encryption Standard
AI	Artificial Intelligence
AIM	America Online Instant Messenger
AOSP	Android Open Source Project
AOT	Ahead-of-Time
API	Application Programming Interface
APNS	Apple Push Notification Service
ART	Android Runtime
BBM	BlackBerry Messenger
DB	Database
CCPA	California Consumer Privacy Act
CDN	Content Delivery Network
CPU	Central Processing Unit
DBMS	Database Management System
DRY	Don't Repeat Yourself
FCM	Firebase Cloud Messaging

GC	Garbage Collector
GDPR	General Data Protection Regulation
GIF	Graphics Interchange Format
GKI	Generic Kernel Image
GNU GPL	General Public License GNU
HAL	Hardware Abstraction Layer
HTC	High Tech Computer Corporation
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ICQ	I Seek You
IRC	Internet Relay Chat
KMI	Kernel Module Interface
kswapd	Kernel Swap Daemon
LDM	Logical Data Model
LGPL	Lesser General Public License
LMK	Low Memory Killer
LRU	Least Recently Used
LTS	Long Term Support
MAC	Mandatory Access Control
OS	Operating System
MITM	Man-In-The-Middle
NLP	Natural Language Processing
ORM	Object-Relational Mapping
OTA	Over The Air

PIN	Personal Identification Number
RAM	Random Access Memory
RDBMS	Relational database management system
SD	Secure Digital
SMS	Short Message Service
SQL	Structured Query Language
SRTP	Secure Real-time Transport Protocol
TEE	Trusted Execution Environment
UID	User Identifier
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
XMPP	Extensible Messaging and Presence Protocol

General introduction

The evolution of mobile operating systems has undergone significant transformations over the past two decades, moving from simple, limited platforms to highly sophisticated environments where communication, collaboration, and data security play a major role. Originally designed for phones with basic features, these systems now form the basis for billions of connected devices worldwide. Among them, Android stands out as the dominant mobile platform, thanks to its openness, flexibility, and rich application ecosystem. Developed by Google, Android is based on a modern architecture, a Linux kernel adapted to mobile devices, and a proven security model. It thus benefits from the support of a vast community of developers and manufacturers, fostering innovation and customization of the system to meet specific needs.

Alongside the evolution of operating systems, instant messaging applications have revolutionized the way people communicate and share information in the digital age. Among them, WhatsApp occupies a central place. With more than two billion users worldwide, it has established itself as a major everyday tool, offering text, voice, and video messaging features as well as end-to-end security mechanisms ensuring the confidentiality of exchanges. However, despite the popularity of these tools, they raise crucial issues related to the protection of young people and the control of their use. The lack of appropriate parental control mechanisms is thus becoming a major concern, particularly given the sensitivity of the information exchanged and the exposure of children to potentially dangerous content.

This is the context in which this thesis is set. It proposes to analyze in depth the Android environment as well as the technical functioning of WhatsApp, detailing their essential mechanisms—from memory management to the security layer—while addressing the issue of parental control applied to WhatsApp. This work is based on a detailed study of the technologies, constraints, and issues related to parental monitoring, while keeping in mind the delicate balance to be achieved between surveillance, respect for privacy, and child protection.

To achieve this objective, the dissertation is structured into five chapters:

→ ***Chapter 1: Study of the Android environment***

This chapter details the architecture of the Android operating system, as well as its memory management, data storage, and security layer mechanisms. It also discusses the impact of Android's open-source nature and how it affects system customization, while detailing the built-in protection mechanisms.

→ ***Chapter 2: Instant messaging applications: focus on WhatsApp***

This chapter provides a detailed exploration of the instant messaging app landscape and an in-depth study of WhatsApp. It analyzes its technical architecture, encryption mechanisms, and specific features that have contributed to its global success.

→ ***Chapter 3: Study of existing parental control solutions***

This chapter lists the solutions available to date, detailing their features, advantages, and limitations. It highlights the constraints specific to the Android environment and to monitoring communications via WhatsApp.

→ ***Chapter 4: Conceptual study of the proposed solution***

This chapter details the proposed solution to address the issue of parental control. It specifies its general architecture, components, and methods used (including scraping and automation) to ensure accurate, secure, and privacy-friendly monitoring.

→ ***Chapter 5: Implementing the Solution*** This final chapter provides a detailed description of the implementation of the proposed project. It presents the working environment used (programming languages, frameworks, tools), as well as the detailed operation of the main functionalities implemented on the parent and child sides.

This research aims to demonstrate the importance of designing parental control solutions specific to the Android environment, while ensuring a balance between the protection of minors, respect for privacy, and the security of digital exchanges. It thus provides new perspectives to support future developments in mobile usage, while addressing crucial challenges of the modern connected society.

Chapter 1

Android Environment Study

Introduction

The Android mobile operating system, developed by Google, dominates the global smartphone and tablet market [1]. Its massive adoption by manufacturers and users is due to its open-source nature, flexibility and vast application ecosystem.

This introductory chapter provides a comprehensive overview of the Android environment, exploring its history, software architecture, key releases, key features, and security issues. This exploration will provide a solid foundation for understanding and appreciating the following chapters.

1.1 History of the Android operating system

Android Inc. was founded in 2003 in Palo Alto, California, by technology pioneers such as Rich Miner, Nick Sears, Chris White, and Andy Rubin. The initial goal of this small startup was to revolutionize the operating system for digital cameras. The idea was to improve image storage management to allow users greater flexibility and no longer be limited by the size of their storage cards. However, despite the innovative nature of this idea, it didn't really take off, forcing the team to rethink its strategy. It was against the backdrop of the expanding mobile phone industry that Android Inc. offered a free operating system platform, thus opposing the practices of industry giants. This innovative approach prompted Google to acquire Android in 2005, marking the beginning of an open-source mobile operating system accessible to all [2].

In September 2008, the first Android smartphone, the T-Mobile G1 (also known as the HTC Dream), was launched. It ran Android 1.0, which already featured the char-

acteristics of Google's strategy for its operating system. This model integrated various of the company's products and services, such as Google Maps, YouTube, and an HTML browser (before Chrome) using Google's search services. [3]

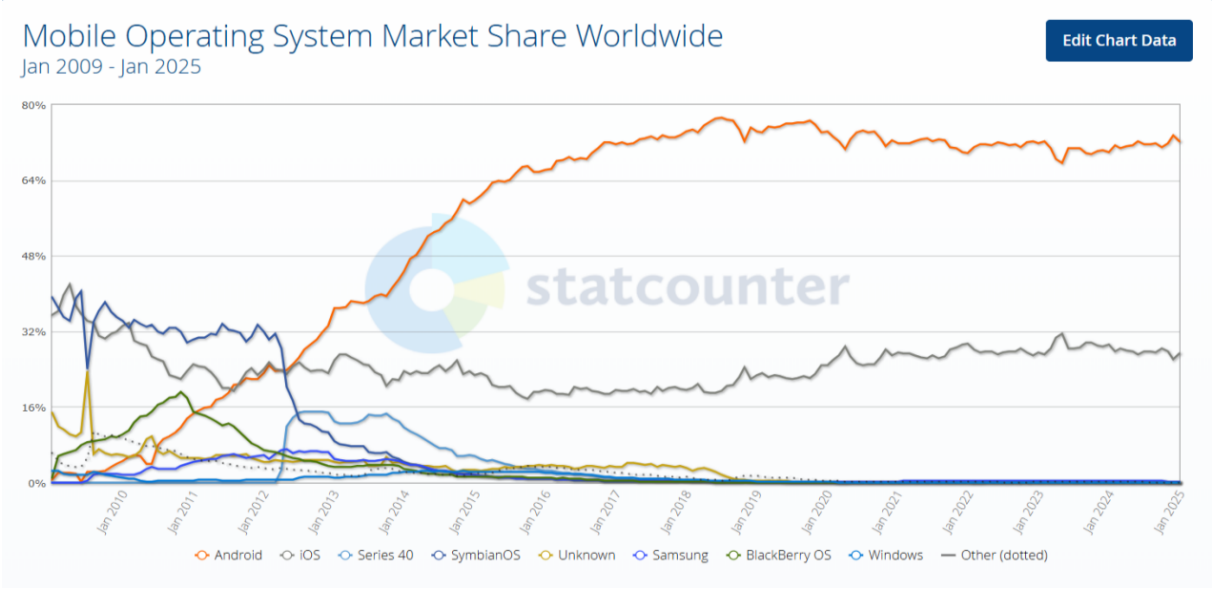

Figure 1.1: Market share of mobile operating systems since 2009 [1]

By analyzing the Figure 1.1, it is clear that Android maintains a dominant position in the mobile operating system market and consistently secures a significant share of the global market, often between 70 and 80 % thanks to its presence on devices in all price categories, from budget to high-end [1].

1.2 Android Platform Architecture

Android is an open-source, Linux-based software platform designed for a wide variety of devices. Its architecture is organized in layers, with each layer providing services and APIs to the upper layers. The Figure 1.2 illustrates the main components that make up the Android platform. Thanks to this modular structure, developers can create rich and powerful applications by leveraging the features offered by the platform [4].

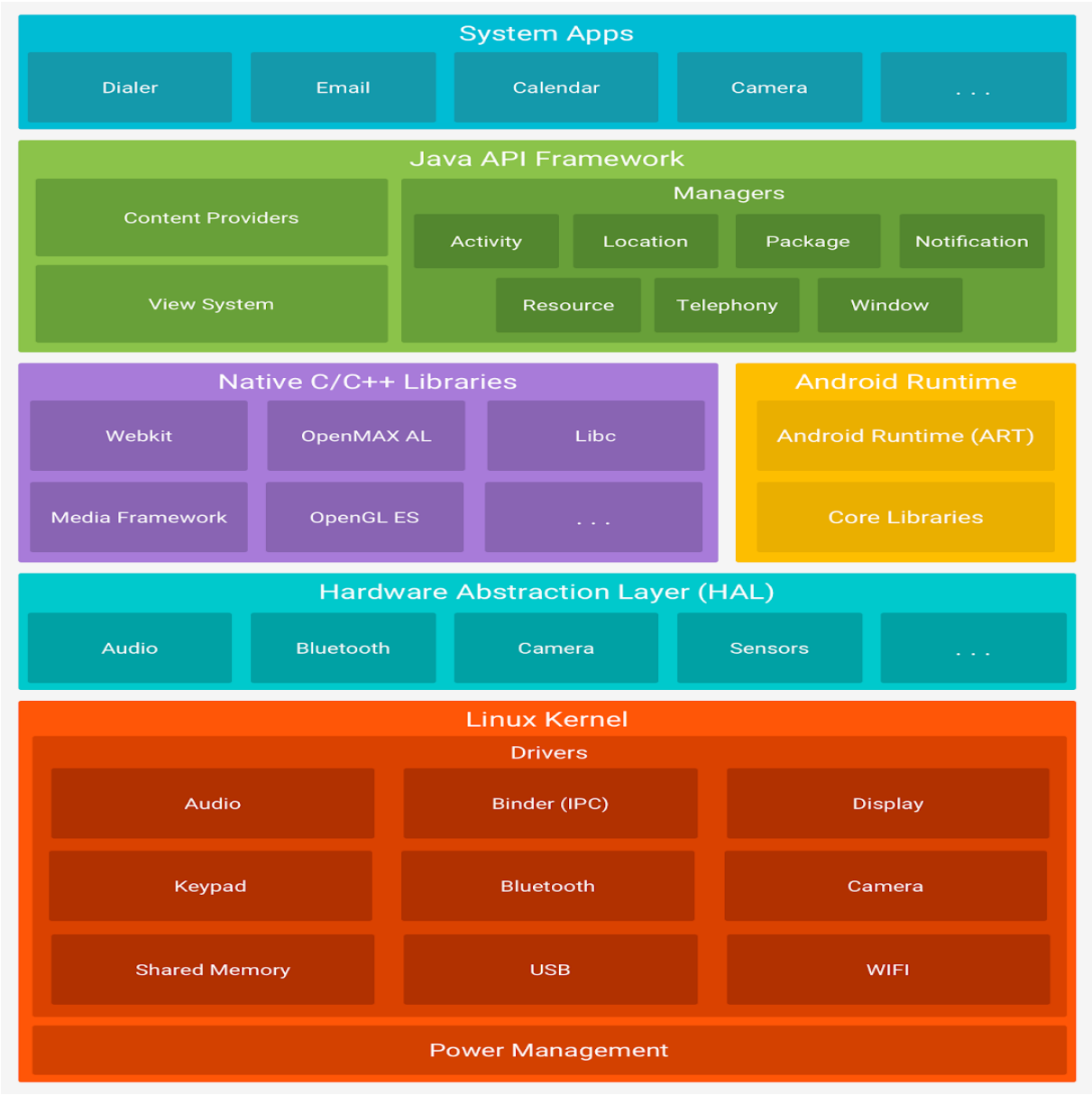

Figure 1.2: Android software stack [4]

1.2.1 Linux Kernel

At the heart of the Android architecture is the Linux kernel. It handles the basic functions of the operating system, such as memory, process, peripheral, and file system management. Android uses a tailored version of the Linux kernel, optimized for mobile devices, particularly in terms of power consumption and memory usage [4].
This adaptation of the Linux kernel for mobile devices consists of integrating Android-specific patches into long-term support (LTS) Linux kernels, now called, after modification, GKI (Generic Kernel Image) kernels [5].

GKI kernels play a key role in Android's modularity by ensuring the separation of generic, hardware-independent kernel code from hardware-specific modules, called vendor modules. This separation facilitates maintenance and updates to the operating system, while also allowing hardware manufacturers to integrate their own optimizations [5].
The interaction between the GKI kernel and vendor modules is done through the Kernel Module Interface (KMI). This interface, composed of symbol lists, defines the global data and functions required by vendor modules to interact with the kernel (see figure 1.3) [5].

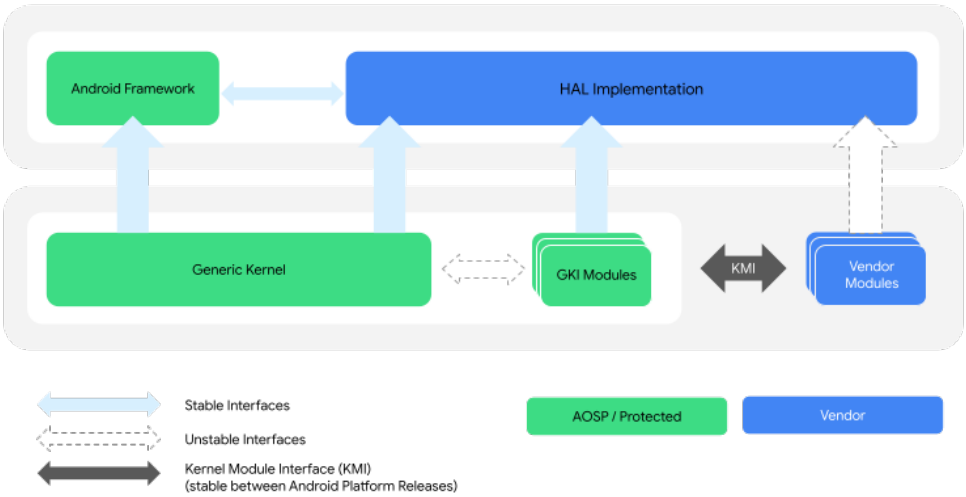

Figure 1.3: Architecture GKI [5]

Using the Linux kernel, and its adaptation through ACKs and GKIs, offers several advantages:

→ **Security :** Android takes advantage of the proven security mechanisms of the Linux kernel, such as permission management and process isolation.

→ **Portability :** The Linux kernel's compatibility with a wide range of processors and peripherals allows Android to be deployed on a multitude of devices.

1.2.2 Hardware Abstraction Layer (HAL)

The Hardware Abstraction Layer provides a standard interface between the Android application framework and the device's hardware. It allows developers to access hardware features, such as the camera, Bluetooth, or sensors, without having to worry about the specifics of each device [6]. The HAL is composed of modules, each implementing an interface for a specific type of hardware component. When an application uses a framework API to access the hardware, the Android system loads the corresponding HAL module to interact with the device [4].

1.2.3 Android Runtime Environment (ART)

The Android Runtime Environment is the virtual machine that runs Android applications. It transforms application code, written in Java or Kotlin, into instructions that the processor can understand and execute. ART has brought a host of significant improvements in performance, efficiency, and compatibility [7].

1.2.3.1 Performance improvement

ART uses Ahead-of-Time (AOT) compilation, which means the application code is compiled into native machine code upon installation. This AOT compilation is essential for smoother performance and faster application launches, as the CPU no longer needs to translate the code each time it is used. This significantly reduces application execution time and improves the user experience [8].

1.2.3.2 Optimized memory management

ART incorporates a garbage collector. This mechanism, a memory cleaning system, allows memory occupied by unused objects to be freed more efficiently, thus reducing memory fragmentation and improving system stability. ART's garbage collector uses a concurrent mark-and-sweep collection system, which means it can run in the background without interrupting application execution. This contributes to a smoother user experience and better application responsiveness [9].

1.2.3.3 Support for 64-bit architectures

ART was designed to support 64-bit architectures, which offer increased performance and greater memory capacity than 32-bit architectures. This compatibility has allowed Android to adapt to evolving hardware and fully exploit the power of modern processors. This translates into faster and more performant applications, and the ability to use resource-intensive applications [7].

1.2.3.4 Improved battery life

One of the most significant benefits of ART is its positive impact on battery life. ART's AOT compilation also contributes to improved battery life. By reducing the CPU load when running applications, ART helps consume less power [8].

1.2.3.5 Debugging with ART

ART also offers significant improvements in debugging. It provides better support for native code debugging, making it easier to develop and debug Android apps [7].

1.2.4 Native C/C++ Libraries

Native libraries are sets of code written in C/C++ that provide specific functionality to Android applications. They are accessible through the application framework and allow interaction with the hardware (OpenGL ES for graphics, Media Framework for multimedia) or provide key services (SQLite for databases, OpenSSL for cryptography) [4].

1.2.5 Java API Framework

The application framework provides developers with a set of services and APIs to build Android applications. It simplifies the reuse of system components and services, allowing developers to focus on the business logic of their applications [4].
Key components of the application framework include :

→ **View system:** A set of graphical components for creating the user interface of applications (buttons, lists, text boxes, etc.) [10].

→ **Resource Manager:** Allows access to application resources, such as images, text strings, and layout files [11].

→ **Notification Manager:** Allows applications to display notifications to the user [12].

→ **Activity Manager:** Manages the lifecycle of an application's activities (screens) [13].

→ **Content Providers:** Allow applications to share data with each other [14].

1.2.6 System apps

Apps are the top layer of Android architecture. They are the programs that users install and use on their devices. They can be pre-installed (system apps) or downloaded from the Google Play Store. System apps, such as Phone, Contacts, or Messaging, provide core functionality to both users and developers. Developers can access these features through the application framework's APIs, eliminating the need to recreate them [4].

1.3 Memory Management in Android

Memory management is a critical aspect of Android development, ensuring that apps run efficiently and don't consume excessive system resources. Android's memory management system is designed to optimize the use of limited resources on mobile devices, which often have less memory than desktop systems.

1.3.1 Memory Types in Android

Android devices use three main types of memory : **RAM**, **zRAM** and **storage** as illustrated in the Figure 1.4. Each type has a specific function in managing data and improving device performance [15] :

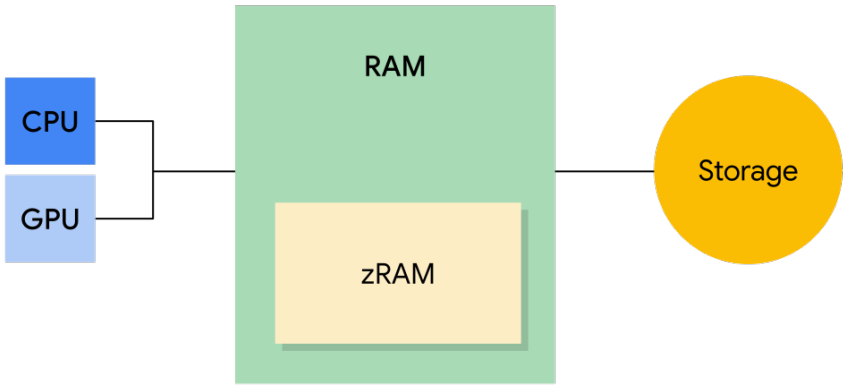

Figure 1.4: Memory types [15]

→ **RAM** is the fastest type of memory, but its size is usually limited. High-end devices generally have the largest amounts of RAM.

→ **zRAM** is a partition of RAM used for swap space. Everything is compressed when placed into zRAM, then decompressed when copied out of zRAM. The size of this portion of RAM increases or decreases as pages are placed into or removed from zRAM. Device manufacturers can set the maximum size.

→ **Storage** stores all persistent data, such as the file system and application code. Unlike RAM and zRAM, storage offers much greater capacity. It is important to note that Android does not use storage for swap space, unlike other Linux systems. Instead, it focuses on efficiently storing user and system data. Proper storage management is essential for maintaining device performance and ensuring sufficient space for user and application data.

1.3.2 Memory allocation strategies

Android uses a dynamic memory allocation system to manage memory between processes, ensuring efficient use of limited resources. The Android Runtime (ART) employs paging and memory mapping (mmap) techniques, where memory modified by an application remains resident in RAM and cannot be paged out. Each application process is initiated by the Zygote process, which preloads shared framework code to reduce memory usage by applications. The system prioritizes memory allocation based on process importance, with foreground applications receiving higher priority, while background processes can be terminated to free up memory in low-resource situations [16].

1.3.3 Memory Pages in Android

In Android, memory is divided into fixed-size blocks called pages, typically 4 KB in size. These pages are the smallest unit of memory management and are classified as **free** or **used**. Free pages represent unused RAM, while used pages are actively used by the system and are classified into specific types based on their purpose and behavior, as follows [15]:

→ **Cached pages :** They store data that can be quickly reused, such as application resources or recently accessed files. Cached pages improve performance by reducing the need to reload data.

→ **Shared pages :** They are used to share common resources, such as libraries or application code, between multiple processes. This minimizes redundancy and optimizes memory usage.

→ **Anonymous pages :** They are private to a process and typically store data such as application-specific variables or runtime objects. They are not backed by files and are created dynamically during runtime.

By classifying memory pages into three categories, Android ensures efficient memory usage, reduces redundancy, and maintains smooth performance even in low-memory situations.

1.3.4 Garbage Collection in Android

Garbage Collection (GC) in Android is an automatic memory management process used by runtime systems and ART. This mechanism identifies and reclaims memory no longer in use by applications, ensuring efficient memory management and preventing

memory leaks that could degrade application performance or cause crashes. In a managed memory environment like ART, the system keeps track of every memory allocation. When it determines that a piece of memory is no longer in use, it releases that memory back to the heap, without programmer intervention. The main goals of garbage collection are to find data objects that can no longer be accessed in the future and to reclaim the resources used by those objects [16].

1.3.4.1 The generational pile

Android uses a generational heap strategy for garbage collection, which divides memory into different regions based on the expected lifetime of objects. This strategy typically includes the following [16]:

→ **Young generation :** where new objects are allocated. Most objects have a short lifespan and are quickly collected in this region.

→ **Old generation :** Objects that have survived several collections in the younger generation are promoted in this region, where they are expected to have a longer lifespan.

1.3.4.2 Advantages of the Generational Heap

→ **Efficiency :** By focusing on the younger generation, where most items have a short lifespan, garbage collection can be carried out more efficiently, reducing overhead costs.

→ **Reduction of break times :** This approach reduces the duration of "stop-the-world" pauses, which can disrupt the user experience.

→ **Performance Improvement :** Generational GC helps reduce fragmentation and optimize memory allocation, which is especially important for devices with limited resources.

1.3.5 Weak memory management mechanism

Android uses several mechanisms to efficiently manage low memory situations. These mechanisms are based on the Linux kernel and are designed to ensure system stability and performance, even under memory pressure. Two key components of Android's low memory management are the Kernel Swap Daemon (kswapd) and the Low Memory Killer (LMK) [15].

1.3.5.1 Kernel Swap Daemon

kswapd is a Linux kernel process responsible for managing memory by reclaiming unused memory pages. It operates in [15]:
• Identifying memory pages that are no longer actively used.
• Swapping or freeing up these pages to make room for new allocations.

1.3.5.2 Low Memory Killer

LMK is a critical Android-specific mechanism that operates when the system is under high memory pressure. It works in [15]:
• Monitoring the system's memory status through the Low Memory Killer Daemon (lmkd).
• Complete processes based on their priority and importance to the user.
• Using a Least Recently Used (LRU) strategy to determine which processes to terminate, to ensure that the most critical applications remain active.

1.3.6 Memory Management Challenges in Android

Memory management in Android presents several challenges that developers must address to ensure optimal app performance and user experience. Here are some of the main challenges :

1.3.6.1 Memory leaks

Memory leaks are a major issue in Android development. They occur when objects are retained in memory when they are no longer needed, preventing garbage collection. This can lead to increased memory consumption, slower app performance, and even crashes [17]. Common causes of memory leaks include :

→ **Retention of resources :** Instances of objects such as bitmaps and file streams, once used, are not explicitly freed, unnecessarily blocking memory.

→ **Maintaining references :** :Object references (e.g., via internal classes, static variables, or event listeners) persist after those objects are no longer needed, preventing them from being reclaimed by the GC.

1.3.6.2 Limited memory resources

Android devices often have limited RAM, especially lower-end models. The operating system keeps apps in memory to allow for rapid changes, which can lead to very little free

memory available for new processes. This limited memory can lead to poor performance and an increased likelihood of apps crashing due to memory exhaustion [15].

1.3.6.3 Garbage Collection overhead

Although Android uses automatic garbage collection to manage memory, the process can be quite overhead-intensive. Frequent garbage collection cycles can cause noticeable pauses in app performance, especially if the app allocates and deallocates memory inefficiently. Developers should understand how the garbage collector works to minimize its impact on performance [18].

Effective memory management is essential for developing high-performing Android apps. By understanding Android's memory architecture, recognizing common challenges, and implementing best practices, developers can create apps that are not only efficient but also provide a seamless user experience. As Android continues to evolve, staying informed about memory management techniques and tools will remain essential for industry professionals.

1.4 Disk Management in Android

Storage management in Android is a critical aspect of the operating system that ensures efficient processing and organization of data on mobile devices. With the increasing amount of data generated by apps and users, efficient storage management becomes essential to maintain device performance and user experience.

1.4.1 Android Storage Architecture

Android's storage architecture provides a flexible and secure approach to data management on mobile devices, leveraging a hierarchy inspired by the Linux file system. It meets the system's operational requirements and user expectations for security and accessibility. Android offers several storage options, such as secure internal storage, shared external storage, and mechanisms for managing user preferences and databases. This variety enables efficient data management while preserving the user experience and application security.

1.4.2 File system hierarchy

The Android file system hierarchy is organized to reflect both the system's operational requirements and user needs. The root file system ("/") is the top-level directory, beneath

which are several important directories, including:

Directory	Function
`/system`	Contains Android operating system files (read-only). This directory includes essential system files and libraries required for the device to function.
`/boot`	Contains the kernel and initial ramdisk (initrd), essential for booting the device. This partition is crucial to the boot process and is usually read-only.
`/data`	Stores application and user data (e.g., application private files, databases, shared preferences). This is the primary location for application data and user settings.
`/storage`	Mount point for external/internal storage (e.g., /storage/emulated). This directory provides access to internal and external storage, allowing applications to read and write data.
`/dev`	Device files (e.g., hardware interfaces). It contains device nodes representing hardware components, facilitating communication between the operating system and the hardware.
`/proc`	Virtual file system for system and process information. It provides a mechanism for the kernel to send information to processes and for processes to send information to the kernel.
`/cache`	Temporary system files (e.g., OTA updates). This directory stores cached data to speed up access to frequently used information.
`/vendor`	Vendor-specific files (e.g., hardware drivers). It contains files provided by the device manufacturer, including proprietary drivers and configurations.
`/mnt`	Temporary mount points for external storage or network shares. This directory is used to mount file systems, such as SD cards or network shares, while the device is running.

Table 1.1: Android File System Main Directories

1.4.3 Storage types in Android

Android offers a variety of storage options to suit different use cases, allowing developers to choose the most appropriate method for their app's data needs. Here's an overview of the main storage types available in Android [19]:

1.4.3.1 Internal storage

Internal storage in Android is app-specific, allowing only the created app to access its files, which enhances security and is ideal for storing sensitive data. It is particularly useful for saving user credentials and configuration files, as seen in banking apps that store login information to prevent unauthorized access and reduce the risk of data breaches.

1.4.3.2 External storage

The external storage in Android is shared storage accessible by multiple apps, including SD cards and built-in storage. It is ideal for media content (photos, videos, downloads) that users want to share or access across different apps.

1.4.3.3 Shared preferences

Shared preferences are a lightweight method for storing small amounts of data in key-value pairs, ideal for saving user settings and simple application configurations. They are commonly used to store user preferences, such as theme choices and notification settings, as well as specific data such as the last level played or sound settings in games. This approach allows developers to efficiently manage data and improve application customization and usability.

1.4.3.4 Databases

Android uses SQLite databases for storing structured data, making it suitable for applications that manage complex relational data sets. It is particularly effective for large data operations in e-commerce platforms and social media applications. By using SQLite, developers can query, update, and organize data efficiently, allowing for the creation of robust and responsive applications capable of handling large volumes of information.

1.4.3.5 Cloud storage

Of theServices like Google Drive, Dropbox, and OneDrive offer remote storage accessible over the internet. This type of storage allows users to sync their data across multiple devices and free up space on their device.

Android's storage management system is designed to balance flexibility, security, and ease of use. With features like Storage Manager, Scoped Storage, and support for different storage types, Android allows users and developers to efficiently manage data while maintaining privacy and performance.

1.5 Open source

1.5.1 What is Open Source ?

An open source software is software whose source code is publicly available. This means that anyone can view, modify, and distribute the code. Open source is based on the principles of collaboration, transparency, and freedom. It's important to note that open source does not mean "free". Open source software can be distributed for free, but it can also be sold commercially. What defines open source is the freedom to access the source code and the ability to modify and redistribute it under the terms of an open source license. These licenses define the rights and obligations of software users, particularly with regard to modifying and redistributing the code [20].
The implications of open source are numerous :

→ **Collaboration :** Open source encourages collaboration among developers around the world, which can speed up software development and improvement.

→ **Transparency :** With the source code accessible, users can verify the software's functionality and identify potential security issues.

→ **Freedom :** Users are free to use, modify, and distribute the software as they see fit, without license restrictions (subject to the terms of the open source license).

1.5.2 Open Source License Types

There are over 80 different flavors of free software licenses, but they generally fall into one of two main categories [21]:

1.5.2.1 Permissive licenses

Permissive open source licenses allow users to use, modify, and distribute software with minimal restrictions, without requiring derivative works to remain open source. Examples include the Apache License 2.0, which allows commercial modification and redistribution with proper attribution, and the MIT License, which only requires attribution to the original authors. These licenses are particularly popular in commercial free software projects because of their flexibility and ease of use [22].

1.5.2.2 Copyleft licenses

Copyleft licenses require that any derivative work be distributed under the same license, ensuring that the software remains free and open. The GNU General Public License

(GPL) requires that modifications also be subject to the GPL, while the Lesser General Public License (LGPL) offers a more permissive approach, often used for libraries, allowing integration with proprietary software under certain conditions. This licensing framework encourages continued sharing and improvement within the free software community [23].

1.5.3 Open Source Licenses Used by Android

Android is based on a Linux kernel, forming the technical foundation of the platform. Much of Android is developed within the Android Open Source Project (AOSP), led by Google, which provides the source code under the Apache 2.0 License. This permissive license allows developers to use, modify, and distribute the code freely, including for commercial purposes. However, Android is not completely open source and uses several licenses, including :

→ **Apache License 2.0 :** Mainly applied to AOSP, it authorizes use and distribution, even for proprietary uses [24].

→ **GNU GPL License :** Applies to some components, such as the Linux kernel, requiring that derivatives also be distributed under the GPL [23].

→ **Other licenses :** Includes components under other open source licenses like MIT or BSD [25].

This diversity of licenses gives Android the flexibility to integrate proprietary elements, which qualifies it as "partially open source".

1.5.4 Benefits of Open Source

→ **Flexibility :**The ability to modify the source code allows manufacturers to customize Android for their devices and developers to create innovative applications.

→ **Community :** The open source community contributes to the development and improvement of Android by reporting bugs, proposing fixes, and developing new features.

→ **Cost :** Using an open source base can reduce development costs for device manufacturers.

1.5.5 Disadvantages of Open Source

→ **Fragmentation :** The freedom to modify the source code can lead to a fragmented Android ecosystem, with different versions of the operating system on different devices. This can complicate application development and system maintenance. For example, applications may not work properly on all versions of Android, and security updates may be rolled out unevenly.

→ **Security :** The availability of source code can make it easier for malicious actors to identify security vulnerabilities. By making the code accessible to everyone, including hackers, open source can potentially increase the attack surface and expose the system to vulnerabilities.

→ **Google dependence :** Despite its open source foundation, Android remains heavily dependent on Google services, which can pose privacy and freedom issues for users.

1.6 Main features of Android

→ **Customizable user interface :** Android offers extensive customization options for both device manufacturers and users. Manufacturers can create custom user interfaces (UI skins), and users can personalize their devices with widgets, themes, and wallpapers.

→ **Application ecosystem :** Android provides access to the Google Play Store, a vast marketplace where users can download and install apps for various purposes: productivity, entertainment, education, games, and more.

→ **Notifications :** Android's notification system allows apps to send updates and alerts to users, providing them with relevant information without them needing to open the app.

→ **Connectivity :** Android devices support various connectivity options, such as Wi-Fi, Bluetooth, NFC, and mobile data, allowing users to stay connected to the internet and other devices.

→ **Integration with Google services :** Android integrates closely with the Google ecosystem, including services like Google Search, Google Maps, Gmail, Google Drive, and many others, improving the overall user experience.

→ **Gesture control :** Android 10 also introduced gesture control features, such as swiping from the left or right edge of the screen to go back, making navigation more intuitive and convenient.

1.7 Android and Security

Android has a robust security framework designed to protect user data and ensure safe app usage. With features like Google Play Protect, regular security updates, and a robust permissions model, Android provides effective protection against potential threats. However, despite these considerable strengths, the platform is not without its challenges. The fragmented nature of the Android ecosystem, characterized by a variety of device manufacturers and update support levels, creates vulnerabilities that can be exploited by malicious actors. Additionally, users often struggle to distinguish between safe and harmful apps, especially when downloaded from third-party sources. As the cyberthreat landscape continues to evolve, balancing Android's impressive security features with its inherent vulnerabilities remains an ongoing challenge for developers and users alike.
Below you will find a variety of security features available on Android devices, as well as several common Android security challenges.

1.7.1 Android Security Features

1.7.1.1 Application Sandbox

The Android platform leverages Linux user-based protection to identify and isolate application resources. To do this, Android assigns each application a unique identifier (UID) and runs it in its own process. Android uses this identifier to implement an application sandbox at the kernel level [26].

1.7.1.2 App signing

Allows developers to identify the app's author and update their app without creating complicated interfaces and permissions. Every app running on the Android platform must be signed by the developer [27].

1.7.1.3 Authentication

Android uses the concept of cryptographic keys related to user authentication which requires storage of cryptographic keys and authenticators from both the service provider and the user. On devices equipped with a fingerprint sensor, users can enroll one or more

fingerprints and use them to unlock the device and perform other tasks. The Gatekeeper subsystem performs device pattern or password authentication in a trusted execution environment (TEE). Android 9 and later include Protected Confirmation, which allows users to formally confirm critical transactions, such as payments [28].

1.7.1.4 Biometrics

Android 9 and later include a BiometricPrompt API that app developers can use to integrate biometric authentication into their apps in a device- and modality-agnostic manner. Only strong biometrics can be integrated with BiometricPrompt [29].

1.7.1.5 Encryption

Once a device is encrypted, all user-created data is automatically encrypted before being written to disk, and all reads automatically decrypt the data before returning it to the calling process. Encryption ensures that even if an unauthorized party attempts to access the data, they cannot read it [30].

1.7.1.6 Keystore

Android offers a hardware database that allows you to generate keys, import and export asymmetric keys, import raw symmetric keys, encrypt and decrypt asymmetric data with appropriate padding modes, and much more [31].

1.7.1.7 Security-Enhanced Linux

As part of the Android security model, Android uses Security-Enhanced Linux (SELinux) to enforce mandatory access control (MAC) on all processes, even those running with root or superuser privileges (Linux capabilities) [32].

1.7.1.8 Trusty TEE

Trusty is a secure operating system (OS) that provides a trusted execution environment (TEE) for Android. The Trusty operating system runs on the same processor as the Android operating system, but Trusty is isolated from the rest of the system by both hardware and software [33].

1.7.1.9 Verified Boot

Verified Boot strives to ensure that all executed code comes from a trusted source (usually OEMs), not an attacker or corruption. It establishes a complete chain of trust,

from a hardware-protected root of trust to the boot loader, boot partition, and other verified partitions [34].

1.7.2 Android Security Challenges

1.7.2.1 Rooting

Allowing an Android device to install a malware-infected app allows that malicious code to bypass most of the system's built-in security mechanisms. However, if you root your device to bypass Android's restrictions layer, you also bypass the security mechanisms that protect your system from malicious code that you haven't authorized to run on your device. This makes Android devices more susceptible to internet-based cyberattacks and the spread of these infections across the corporate networks they connect to [35].

1.7.2.2 App Permissions

Although Android has a robust permissions system, apps can still request permissions that may be confusing or perceived as risky by users. This can lead users to not install an app or to grant permissions when the creator of the permission has not been installed [36].

1.7.2.3 Insecure Apps

Another security threat to Android devices stems from the open nature of the app submission process on Google Play: apps don't ship with malicious code, but use insecure software designs. When developers leave security holes in their code, hackers or malware can exploit them to compromise your device. The malicious code uses the permissions you granted to the insecure app to bypass your device's security, much like thieves steal key cards from unwitting employees in movies [35].

1.7.2.4 Google Play Malware

LOne of the major security risks in the Android ecosystem is the risk of downloading apps from the Google Play store that contain malware. Google makes it easy for developers to add their apps to its app store, creating a vast selection of apps for Android users. However, the store's lack of enforcement makes it easier for programmers to upload apps containing malicious code to Google Play. These malicious apps can masquerade as anything from games to Android antivirus utilities [35].

1.7.2.5 Open Source Nature

Android's open source code allows for extensive scrutiny by developers and potential attackers, which can lead to the discovery and exploitation of vulnerabilities. This openness also contributes to ecosystem fragmentation, as different manufacturers modify the operating system to suit their devices, resulting in a wide range of hardware and software configurations [24].

1.7.2.6 Fragmentation Version

Fragmentation complicates the process of distributing security updates on time to all devices. Device manufacturers and carriers are responsible for implementing and distributing these updates, which can lead to delays and inconsistencies. Some older devices may remain vulnerable to security risks for extended periods [35].

Conclusion

This first chapter laid the essential foundation for our study by exploring the Android environment in depth: its history, architecture, key components, open source model, the evolution of its recent versions, and its features. A careful look was taken at security, a crucial issue for a parental control application.

With this understanding of Android, we move on to the heart of our topic: instant messaging apps. The next chapter will focus on a detailed analysis of these apps, with a particular focus on WhatsApp, a major global communications platform.

Chapter 2

Instant Messaging Apps: A Complete Overview with a Focus on WhatsApp

Introduction

Instant messaging has transformed the way we communicate, offering a fast and convenient alternative to traditional phone calls and text messages. Apps like WhatsApp, Telegram, and Signal are now essential tools in our daily lives. They allow us to exchange messages, photos, videos, and even make real-time voice and video calls. This in-depth analysis of instant messaging apps examines their evolution, technical features, and impact on society, with a particular focus on WhatsApp, one of the most popular apps in the world.

2.1 Instant messaging applications

2.1.1 Introduction to Instant Messaging Applications

2.1.1.1 Definition

Instant messaging refers to a form of text communication in which two or more people exchange messages in real-time or asynchronously through dedicated software applications or integrated chat platforms. These applications allow users to send text, multimedia, and even files over the internet, enabling seamless communication between different devices [37].

2.1.1.2 Évolution

The evolution of instant messaging has been remarkable, moving from basic text-based communication to feature-rich platforms. The first forms of instant messaging emerged in the late 1980s and early 1990s with Internet Relay Chat (IRC) [38], which allowed real-time communication within a group. In the mid-1990s, ICQ became one of the first commercial instant messengers, followed by other services such as AOL Instant Messenger (AIM).

The 2000s saw the rise of mobile instant messaging platforms, of which BlackBerry Messenger (BBM) is a notable example. The introduction of WhatsApp in 2009 and other apps like Telegram and WeChat further revolutionized the space by offering multimedia sharing, end-to-end encryption, and cross-platform compatibility. In the corporate world, tools like Slack (2013) and Microsoft Teams (2017) introduced real-time collaboration and integration with productivity tools, redefining workplace communication [39].

2.1.2 Features of Instant Messaging Applications

Instant messaging apps are distinguished by a set of features that make them popular and functional for users. Here are the main features that define these platforms [40]:

2.1.2.1 Real-time messaging

Instant messaging apps allow you to communicate instantly with your contacts. They also offer the ability to send and receive messages almost instantly. This promotes seamless and dynamic exchanges between users.

2.1.2.2 Bots and Automation

Many applications integrate bots that automate certain interactions, facilitating instant responses to common requests, booking management, or customer support. These systems improve the user experience and increase the effectiveness of communications.

2.1.2.3 Group discussions

Group chat features allow multiple users to chat simultaneously, making it easier and more organized to coordinate projects or communicate with friends. They can also share ideas, files, and information in real time.

2.1.2.4 Multimedia sharing

Instant messaging apps make it easy to share various types of multimedia content, such as images, videos, audio files, and documents. This sharing capability enriches conversations and provides a better communication experience.

2.1.2.5 Voice and video calls

The ability to make voice and video calls directly through the app is a key feature, allowing users to connect in a more personal and interactive way, without requiring other services or software.

2.1.2.6 Encryption

Data security is a major concern for many users. Instant messaging applications use encryption protocols to protect conversations and thus ensure the confidentiality and security of information exchanged between users.

These combined features make instant messaging apps an essential tool for modern communication, both personal and professional. They continue to adapt and evolve in line with users' growing expectations for functionality and security.

2.1.3 Popular Instant Messaging Apps

Instant messaging apps are indeed at the heart of modern communication, offering users quick and efficient ways to stay connected. Here's a rundown of the most popular apps :

2.1.3.1 WhatsApp

WhatsApp is a leading instant messaging app with over 2 billion active users worldwide. It allows users to send text messages, make voice and video calls, and share multimedia files, while also supporting group chats and instant status updates. One of WhatsApp's key features is its end-to-end encryption, which ensures the privacy and security of communications. The app is available on multiple platforms, including Android, iOS, and desktop, and uses internet data to offer low-cost messaging. Additionally, WhatsApp Business provides businesses with tools to communicate with customers through automated messaging and support, expanding its use in both personal and business settings [41].

2.1.3.2 WeChat

WeChat is a versatile messaging, social media, and mobile payment app developed by Tencent in China. It has garnered over 1 billion users since its launch in 2011. It allows users to send messages, make calls, share multimedia content, and post updates to Moments. WeChat stands out for its integration of various services, including games, online shopping, and access to government services, as well as support for mini-programs that enhance functionality. WeChat Pay offers seamless mobile payment options, making it a popular choice for transactions. Overall, WeChat has significantly changed communication and interaction in both personal and professional settings, particularly in China [42].

2.1.3.3 Facebook Messenger

Facebook Messenger is a popular messaging app developed by Facebook and launched in 2011. It enables seamless communication through text, voice, and video calls, and allows users to send multimedia messages, share photos and videos, and participate in group chats. Its main features include money transfer, integration with Facebook services, and support for chatbots. The app also offers interactive elements such as stickers and games, which enhance user engagement. With billions of users worldwide, Messenger has transformed communication and established itself as an essential tool for connecting with friends, family, and businesses [43].

2.1.3.4 Telegram

Telegram is a cloud-based messaging app launched in 2013 by Pavel Durov and known for its speed and security. It offers a variety of communication options, including text messaging, voice and video calls, and group chats, as well as the ability to share large files and create channels and supergroups for thousands of members. The app emphasizes privacy with features such as end-to-end encryption for secret chats and self-destructing messages. Additionally, Telegram supports bots for task automation and offers a user-friendly interface with customizable themes and stickers. As a result, Telegram has gained millions of users worldwide, becoming a popular alternative to other messaging platforms [44].

2.2 WhatsApp

2.2.1 Introduction to WhatsApp

WhatsApp is a cross-platform instant messaging app that allows users to send text messages, voice messages, images, videos, and make voice and video calls. It is owned by Meta Platforms (formerly Facebook) and is one of the most widely used communication apps in the world. As of June 2023, WhatsApp had approximately 2.78 billion unique active users worldwide [45]. The app is also popular with businesses, with WhatsApp Business experiencing significant growth, including 300 million downloads by 2023 [46].

2.2.2 History of WhatsApp

WhatsApp was founded in 2009 by Brian Acton and Jan Koum, both former Yahoo employees. The app was initially designed as a status-sharing platform, where users could post updates next to their name. Koum named it "WhatsApp" to evoke the phrase « what's up » [47].

In 2013, the app transitioned from a paid service to a freemium model, which significantly increased its popularity. Initial challenges included server outages and financial constraints, but the founders' focus on privacy and user experience allowed the app to thrive. In 2014, WhatsApp was acquired by Facebook (now Meta) for 19 billion $, marking one of the largest technology acquisitions in history [47].

Over the years, WhatsApp has introduced several features, including voice and video calls, group chats, and WhatsApp Business, which is aimed at business users. Its end-to-end encryption, introduced in 2016, has further solidified its reputation as a secure messaging platform.

2.2.3 Features of WhatsApp

WhatsApp is a powerful messaging platform that offers a wide variety of features to meet both personal and professional communication needs. Here are some of its most notable features :

→ **End-to-end encryption :** End-to-end encryption, ensuring that only the intended recipient can read messages [48].

→ **Group discussions :** Group chats allow multiple users to share information, coordinate events, and discuss various topics in real time [49].

→ **Multimedia sharing :** Users can easily share images, videos, documents, and even their location on WhatsApp. This versatility has made the app indispensable for personal and professional communication [50].

→ **Voice and video calls :** WhatsApp has moved beyond text messaging and now offers high-quality voice and video calls [51].

→ **Cross-platform accessibility :** WhatsApp is a highly versatile messaging platform that works seamlessly across multiple devices, including Android and iOS smartphones, as well as computers through WhatsApp Web and dedicated desktop apps for Windows and MacOs [51].

2.2.4 Popularity

As of April 2024, WhatsApp continues to dominate the global messaging app market with approximately two billion monthly users, making it the world's most popular mobile messenger. After WhatsApp, WeChat has a significant user base, particularly in China, while Facebook Messenger ranks third with nearly one billion users, although it has seen a decline in downloads in recent years. Other notable competitors include Telegram, known for its speed and security features, which appeals to users who prioritize privacy in their communications [52].
In conclusion, WhatsApp's leadership in the messaging app market is a testament to its importance in modern communication.

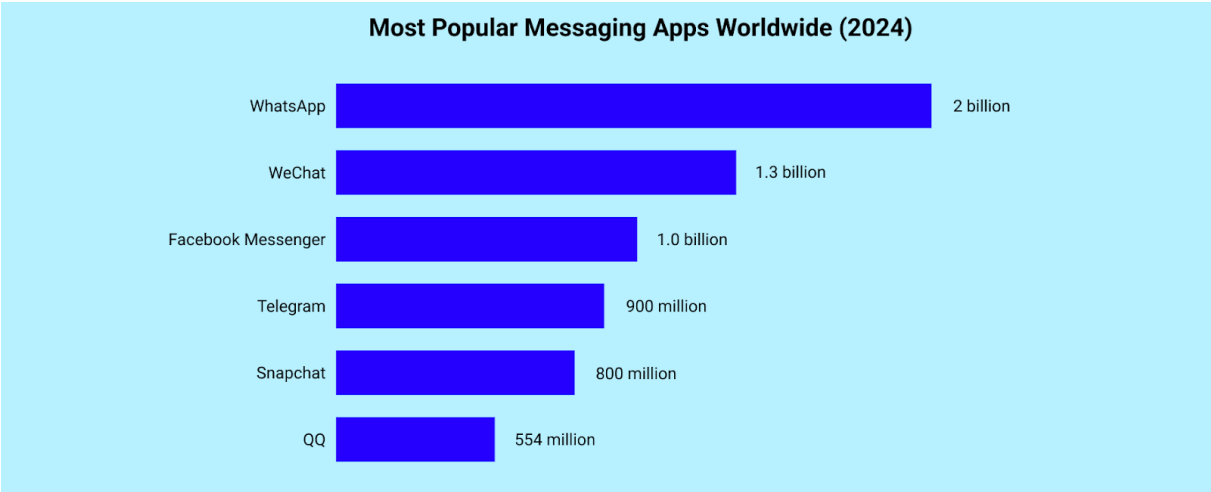

Figure 2.1: Most Popular Messaging Apps Worldwide in 2024 [52]

2.2.5 WhatsApp Technical Architecture

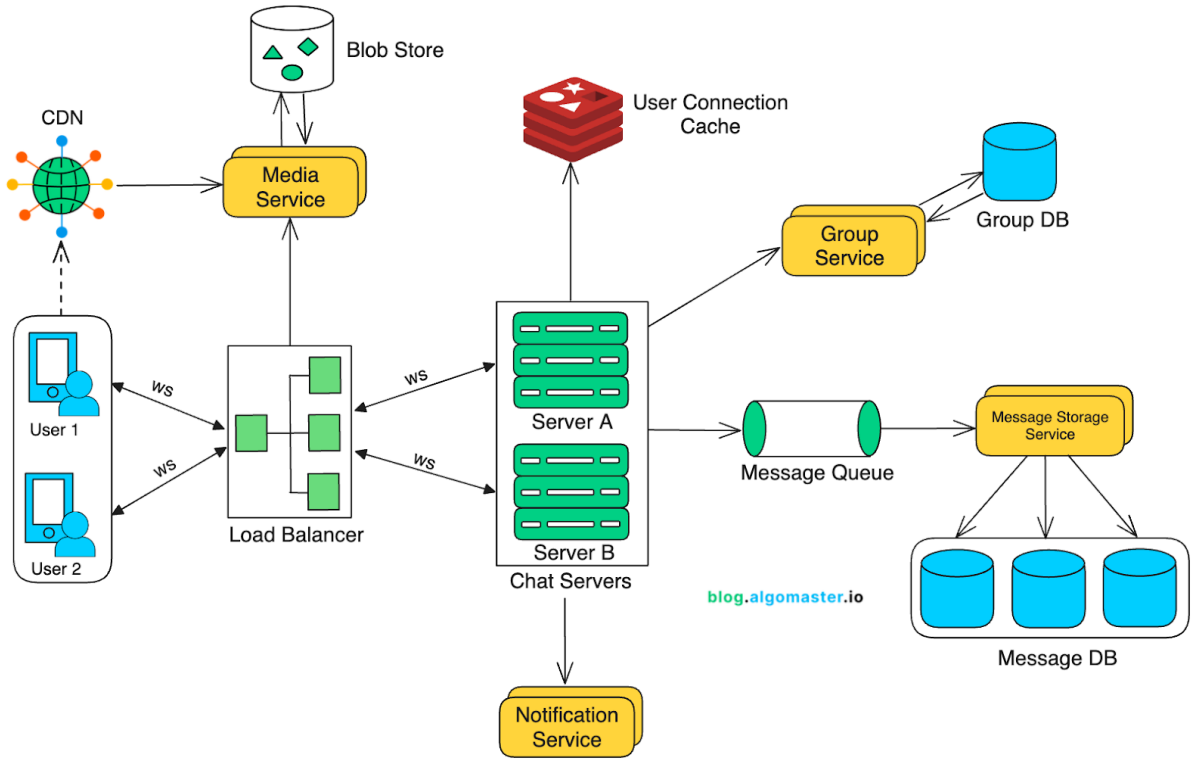

Figure 2.2: WhatsApp Architecture Diagram [55]

WhatsApp's architecture is designed to handle billions of users while ensuring real-time messaging, security, and scalability. It does this by leveraging a complex system optimized for speed and reliability. Below is an overview of its main architectural components :

2.2.5.1 Client-server model

WhatsApp uses a client-server model as a fundamental part of its architecture. Here's how it works in the context of WhatsApp :

→ **Customer side :** The WhatsApp application acts as a client on multiple platforms, including Android, iOS, and the web :

- It uses a lightweight SQLite database to store conversations locally on the user's device.
- WebSockets for persistent, real-time connectivity with its servers to facilitate seamless sending and receiving of messages.
- End-to-end encryption via the Signal protocol to ensure that only the intended sender and recipient can decrypt messages.

- For tasks that don't happen in real time, such as downloading media, WhatsApp communicates with its servers via HTTP and uses XMPP or a custom protocol for messaging. When a user sends a message, the client encrypts it with the recipient's public key before sending it to WhatsApp's servers, which only act as intermediaries in the communication process [53].

→ **Server side :** WhatsApp's server-side scaling strategy is meticulously designed to handle its vast user base while ensuring high availability, low latency, and stable performance. The platform combines various technologies and architectural principles to achieve these goals [54].
Key elements of WhatsApp's server scaling strategy :

- **Microservices :** WhatsApp uses a microservices architecture, which structures the application as a collection of loosely coupled services, each dedicated to specific functionalities and communicating through well-defined APIs. This design allows the chat servers, which handle many simultaneous connections and facilitate real-time communication with minimal latency using protocols such as WebSockets, to be scaled independently based on demand, improving the overall scalability of the platform. Furthermore, the resilience of the architecture means that if one service fails, the others remain operational, ensuring high reliability. Microservices also allow for the rapid development and deployment of new features without overhauling the entire system [54].
- **Load balancer :** Load balancing involves distributing traffic across multiple servers. This significantly improves an application's performance and reliability by preventing a single server from being overwhelmed. WhatsApp uses load balancing to efficiently scale its platform in several ways. By distributing traffic across its servers, load balancing ensures that no server is overloaded, improving overall performance for users. Additionally, this technique improves the platform's resilience to failures: if one server goes down, other servers can continue to handle traffic, maintaining the platform's overall reliability [54].
- **Caching :** Caching is a technique that improves application performance by storing frequently accessed data in high-performance locations such as memory or solid-state drives, reducing the frequency of database access. WhatsApp leverages various caching strategies to efficiently scale its platform. These include in-memory caching, where frequently accessed data is stored for quick retrieval, and disk caching for persistent storage. Additionally, WhatsApp

uses distributed caching to spread data across multiple servers, improving scalability. A specialized component of this caching system is the user login cache, which uses fast in-memory solutions like Redis to store each user's active login details, including the chat server they are linked to and the timestamp of their last activity. Clients periodically send heartbeat signals to update their **last_active** timestamp, allowing WhatsApp to efficiently manage **online/offline** status and **last seen** features. If the time elapsed since the last_active timestamp is below a set threshold, this configuration allows the app to efficiently reflect the user's real-time presence [54].

- **Notification Service :** WhatsApp's notification service is essential for sending real-time notifications to users, especially when they are offline. When a user is offline, the chat server forwards incoming messages to the notification service, allowing the user to receive messages even if they are not using the app. To improve efficiency, the chat server uses a message queue rather than contacting the notification service directly, reducing potential delays. The service is also compatible with push notification providers such as Firebase Cloud Messaging (FCM) and Apple Push Notification Service (APNS), allowing messages to be delivered as push notifications, encouraging user engagement in their conversations when they return online [55].
- **Message Service :** WhatsApp's messaging service consists of three components: the message queue, the message storage service, and the message database. The message queue is a high-speed, distributed system that temporarily stores messages before they are processed by the storage service. This system acts as an intermediary, separating message storage from real-time processing on the chat servers, thus reducing latency and improving the application's scalability. The message storage service is responsible for reliably storing chat messages, retrieving them quickly, and archiving them efficiently. It retrieves incoming messages from the message queue and saves them to the message database, which is designed for high write throughput and efficient retrieval to handle the large volume of messages typical of real-time chat applications, while ensuring users have seamless access to their old messages [55].
- **Group service :** WhatsApp's Groups service is essential for managing all group-related functions, such as creating groups, updating their details, and tracking their members. When a message is sent to a group chat, the chat servers query the Groups service for the current list of group members. The Groups database stores and retrieves all information related to chat groups,

including IDs, member lists, administrator roles, and settings [55].

- **Media Service :** The Media Service is responsible for uploading and managing media content, including images, videos, and audio files. It securely stores these files in blob storage, while keeping metadata organized in a separate database for easy access: file type, size, and upload timestamp. By offloading media storage from the main chat servers, the Media Service helps reduce bandwidth usage on those servers, improving overall application performance [55].
- **Blob Store :** WhatsApp's Blob Store serves as the storage base for the chat app's media content. It contains various media types such as images, videos, audio files, and documents. It is specifically designed to handle large volumes of media while ensuring fast, secure, and reliable access. The Media Store typically uses cloud-based object storage solutions such as Amazon S3, Google Cloud Storage, or Azure Blob Storage, which offer high durability, scalability, and cost-effectiveness to meet users' diverse media needs [55].
- **Content Delivery Network :** To minimize latency when uploading or downloading media, WhatsApp uses a content delivery network (CDN) that distributes files to locations geographically closer to users. When a user shares a media file, the client application uploads it directly to the CDN, ensuring that it is stored near the intended recipient. Rather than sending the file itself, the client passes the file URL to the chat server as part of the message, allowing other users to quickly and efficiently download and access the content from the nearest CDN location. Once the files are uploaded to the CDN, the media service retrieves them and stores them in a blob store for long-term retention. This strategy effectively reduces the load on the chat servers, minimizes latency, and significantly improves media delivery speed for users [55].

2.2.5.2 Communication protocol

WhatsApp uses a combination of custom and standardized communication protocols to ensure fast, secure, and reliable messaging, voice and video calls, and media sharing. Here's an overview of the main protocols :

→ **Extensible Messaging and Presence Protocol**
WhatsApp uses the XMPP protocol for its messaging, which ensures efficient message transmission and presence information. This protocol supports one-on-one and group chats and enables decentralized communication [56].

→ **WebSocket protocol**
WebSockets facilitate two-way communication between the client and server over a single connection. This enables real-time data transfer, ensuring messages are delivered instantly [57].

→ **Hypertext Transfer Protocol (HTTP)**
It is used to upload media content such as images and videos to WhatsApp servers, allowing users to share media seamlessly [58].

→ **Secure Real-time Transport Protocol (SRTP)**
This protocol allows multimedia content (such as images and videos) to be encrypted during transmission, thus providing increased security [59].

→ **Signal Protocol**
WhatsApp uses the Signal protocol for end-to-end encryption, meaning only the intended recipients can read messages. This enhances user privacy and protects communications from interception [57].

2.2.5.3 Data management and synchronization

→ **Server-side databases**
Mnesia is a distributed database system (DBMS) integrated into the Erlang/OTP environment. It is designed to ensure high availability, fault tolerance, and concurrency. It is very likely that WhatsApp, which relies on Erlang/OTP, used Mnesia for essential functions such as contact lists, user information, and session management [53]. Main use cases :

- **Session management :** Tracks user's online/offline status and active connections.
- **Message queues :** Temporarily stores undelivered messages for offline users (deleted after delivery).
- **Metadata :** Stores non-content data (e.g., timestamps, sender and recipient IDs, message routing information).

→ **Client-side database**
WhatsApp uses a local, file-based SQLite database on each user's device to store messages, contacts, and settings. This database, consisting of msgstore.db for messages and wa.db for contacts and settings, is encrypted using the Advanced Encryption Standard (AES) algorithm with a 256-bit key. The encryption keys are stored in

the key and base_key files. Security relies on device-specific keys, meaning each device has its own encryption keys. This ensures that even if a device is compromised, the encrypted data remains inaccessible without the correct encryption keys [60].

→ **Backup systems**

WhatsApp offers backup systems designed to help users protect their chat history and media. Here are the main features of these backup systems [61]:

a) Cloud backups :

- **Google Drive :** WhatsApp offers the ability to back up chat history and media to Google Drive. This is the primary cloud backup method for Android users.
- **iCloud :** iOS users can back up their WhatsApp data to iCloud. Like Google Drive, this cloud storage option allows users to back up their chat history and media.

b) Local backups : In addition to cloud backups, WhatsApp also supports local backups. This means users can store their chat history and media directly on their device or on external storage, such as a computer. Local backups are often used to restore chats without relying on an internet connection or if the user doesn't want to store their backup in the cloud.

→ **Data synchronization**

WhatsApp syncing refers to the seamless process that ensures messages, media, and chat history remain consistent across multiple devices. At its core, the app uses the phone as a hub for all data; when messages are sent or received, they are first stored on the phone. To sync this data with other devices, the phone must have an active internet connection, which allows it to act as a server that relays messages to and from WhatsApp's servers, which then echo them in real-time to secondary devices. The sync process begins when the user sets a secondary device, for example, when opening web.whatsapp.com. Once connected, the phone transmits an encrypted copy of recent chat history to the linked device, ensuring that any new messages or actions, such as replies or deleted chats, are instantly reflected across all connected platforms [62].

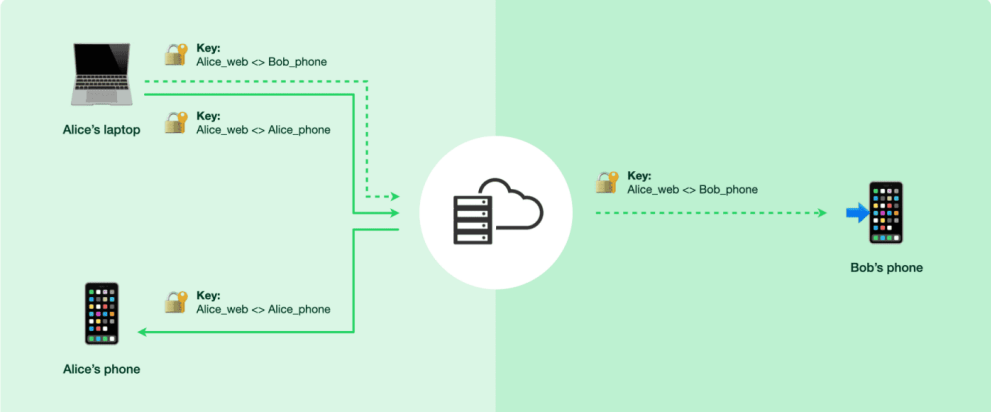

Figure 2.3: WhatsApp Multi-Device Sync [62]

WhatsApp's end-to-end encryption is essential for maintaining privacy during syncing. Messages are encrypted on the phone before being sent to WhatsApp's servers and then to the recipient, ensuring that unencrypted data is never transmitted for storage. Conversely, when syncing with another device, the phone shares encrypted messages with the linked device via a secure channel, thus maintaining the privacy of its communication. With the rollout of enhanced multi-device support, users can now link up to four additional devices, such as another phone, a tablet, or a computer, each with its own encryption key linked to the primary phone. When a message is sent, it is encrypted separately for each linked device, allowing independent functionality even if the primary phone is offline for up to 14 days, as these devices can retrieve data from WhatsApp's servers based on the last sync. Users can back up their chats to Google Drive on Android or iCloud on iPhone, and these backups are associated with their phone number. When the user upgrades to a new phone or reinstalls WhatsApp, they can restore their chat history from these backups. However, it's important to note that these backups aren't end-to-end encrypted unless the user manually opts for an encrypted backup option using a password or key [62].

2.2.6 Security and Privacy in WhatsApp

2.2.6.1 Security aspect

Ensuring secure communications is essential, especially in an era marked by growing privacy concerns and data breaches. Verification methods play a key role in confirming the authenticity of communication partners and the integrity of shared data.
Here are some verification methods commonly used by WhatsApp to ensure secure communications :

1. **End-to-end encryption**
   WhatsApp uses end-to-end encryption to protect messages, ensuring conversations remain private and secure throughout transmission. This encryption process means that when a message is sent, it is encrypted on the device and can only be decrypted by the recipient's device. No intermediary, including WhatsApp itself, can access the contents of the messages [63].

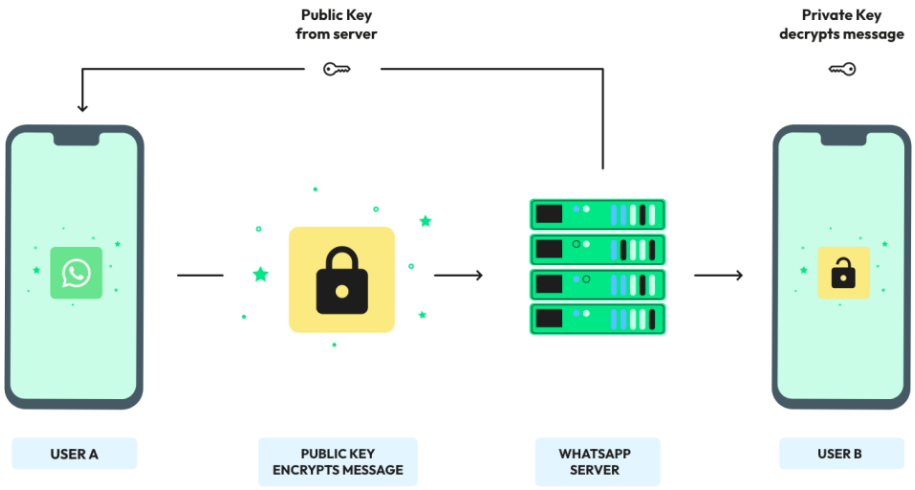

Figure 2.4: WhatsApp end-to-end encryption [63]

Here's a step-by-step explanation of how WhatsApp encryption works [63]:

→ **Key generation :** When you first install WhatsApp, the app creates a pair of cryptographic keys for your device: a public key and a private key. The public key is shared with other users, while the private key is securely stored on your device and never leaves it.

→ **Key exchange :** When you start a conversation with someone on WhatsApp, the app automatically exchanges public keys between the two devices. This key exchange happens transparently in the background.

→ **Message encryption :** When sending a message to a contact, WhatsApp encrypts it on the device using the recipient's public key. This encryption transforms the message into unreadable gibberish (ciphertext) before it is sent to WhatsApp's servers.

→ **Server Relay :** WhatsApp's servers act as relays for these encrypted messages. They receive the encrypted text and transmit it to the recipient's device without accessing its contents.

→ **Decrypting the message :** When the recipient's device receives the ciphertext, it uses their private key to decrypt the message. This private key, securely stored on their device, is the only one capable of decrypting the ciphertext.

→ **Displaying the message :** Once the recipient's device decrypts the ciphertext, it displays the message in its original, readable form.

In conclusion, WhatsApp's end-to-end encryption preserves the confidentiality and security of messages by generating unique cryptographic keys for each user, enabling automated key exchange without user intervention. This means that even though messages pass through WhatsApp's servers, only the intended recipient can access the content. This security framework protects communications from interception and emphasizes the importance of privacy, allowing users to communicate with confidence.

2. **Two-step verification**
   Two-step verification is an optional security feature that adds an extra layer of protection to WhatsApp accounts. Users can enable this feature to set a six-digit PIN, which is required every time they register their phone number on WhatsApp. This additional requirement helps protect accounts from unauthorized access because even if someone manages to obtain the user's phone number, they will still need to know the PIN to complete the registration process on a new device. By using two-step verification, users can significantly increase the security of their accounts [64].

3. **Biometric lock**
   For added security, WhatsApp offers a biometric lock feature on supported devices, allowing users to secure the app using biometric authentication methods such as fingerprint scanning or facial recognition. This feature enhances account security by ensuring that only the authorized user can access WhatsApp, even if someone else has physical access to the device. By enabling biometric lock, users can rest easy knowing that their private conversations and information remain safe from prying eyes [64].

4. **Disappearing messages**
   Disappearing messages is another notable feature that allows users to maintain a higher level of privacy in their conversations. When enabled, any message sent within a chat automatically disappears after a set amount of time, usually set to 7 days. This setting ensures that sensitive conversations don't linger, reducing the risk of unauthorized access to potentially private information. By using this feature,

users can feel more secure when sharing information they'd prefer not to have saved long-term [64].

In conclusion, WhatsApp offers a robust set of security features to protect user privacy and secure communications. These include end-to-end encryption of messages and calls, two-step verification, customizable privacy settings, secure backups, and tools to report suspicious activity. By utilizing these features and following best practices, users can enhance their security and protect their personal information in the digital world [64].

2.2.6.2 Privacy aspect

WhatsApp's privacy policy describes its data management practices, emphasizing transparency and user control. It describes the collection of personal and usage data, as well as how this information is used to improve services and security. Users can adjust privacy settings, delete their account, and export their chat history. While data can be shared with third parties and Facebook, WhatsApp ensures security through end-to-end encryption. Users can access and update their data, and the policy includes a commitment to notify them of significant changes, which promotes trust and transparency [65]. Here is a structured overview of the main aspects [65]:

1. **Data collection**
   WhatsApp collects various types of data to provide its services. Account information includes the user's phone number, profile name, photo, and status. For messages, while content is protected by end-to-end encryption, unread messages may be temporarily stored on servers. Device and usage data includes details such as device type, operating system, IP address, app activity (such as features used and interaction times), and cookies for analytics. For contacts, if users allow it, WhatsApp accesses contacts to facilitate messaging, with this information stored locally or encrypted on servers. Location data is only collected if shared by the user or when using live location features. Finally, for users who use payment services, WhatsApp collects transaction details, including payment and transaction information.

2. **Use of data**
   WhatsApp uses the collected data to improve the user experience and ensure the platform's security. This data is essential for core features such as messaging, calling, and group chats, as well as to personalize user interactions. The platform uses automated systems to detect suspicious activity and protect against spam, while implementing security measures such as two-step verification. WhatsApp keeps users informed of updates and policy changes and conducts analytics to improve

the service and operational efficiency. While no direct advertising is displayed, the data may be shared with Meta companies for more relevant advertising. Overall, WhatsApp strives to provide a secure and user-friendly messaging environment while prioritizing user privacy.

3. **Data sharing**
   WhatsApp shares information with Meta Companies, including phone numbers, transaction data, and device details, to improve experiences on Meta platforms, as specified in Meta's Data Policy. Additionally, third-party services that users interact with on WhatsApp may receive user data, such as phone numbers and chat histories, and may use third-party providers for hosting. To ensure legal compliance, WhatsApp responds to legal requests and prevents harm in accordance with applicable laws. While the content of encrypted messages cannot be disclosed, associated metadata, such as user IDs and timestamps, may be shared.

4. **User Rights**
   WhatsApp offers users significant rights and controls over their data and privacy, including the ability to export their account information and delete their account, which removes their data from the platform. The app includes various privacy settings that allow users to manage the visibility of their latest status, profile picture, and status updates, as well as block unwanted contacts and disable contact syncing and location sharing. Additionally, WhatsApp supports data portability in accordance with regulations such as the GDPR and CCPA, underscoring its commitment to user empowerment and privacy.

WhatsApp's privacy policy emphasizes user control and transparency in data processing. It ensures that users are informed about how their information is used and provides them with various options for managing their privacy. By regularly reviewing and using the available privacy settings, users can protect their personal information while still enjoying the app's features [65].

2.2.7 WhatsApp's Challenges and Global Impact

WhatsApp, one of the most widely used messaging apps in the world, has profoundly changed the way people communicate. With over two billion users, its influence is undeniable. However, this popularity also brings significant challenges and criticism.

2.2.7.1 The global impact of WhatsApp

WhatsApp, with over 2 billion active users worldwide, has profoundly influenced personal communication, business operations, and social and cultural dynamics. Here's a look at its global impact :

1. **Personal communication**
   WhatsApp revolutionized connectivity by making instant messaging accessible and affordable, especially in regions where texting or calling costs can be prohibitive. With its low data usage and free voice and video calls, the app has successfully bridged communication gaps for millions of people, enabling seamless interactions across great distances. Furthermore, the ability to share multimedia content, such as photos, videos, and voice notes, has enriched personal communication, making it more expressive and engaging, and thus improved the overall quality of interactions between users around the world [66].

2. **Professional use**
   WhatsApp has become an essential tool for customer engagement, with over 50 million businesses using the platform to interact with their customers. Over 175 million people communicate with business accounts daily, highlighting the app's effectiveness as a direct communication channel that improves customer service and satisfaction. Meanwhile, WhatsApp's real-time messaging features contribute to operational efficiency by streamlining internal communications. This improves team collaboration and accelerates decision-making processes [67].

3. **Social and cultural influence**
   WhatsApp transcends cultural boundaries, providing users with a platform to share their experiences, traditions, and perspectives, fostering the emergence of a global community. This rich cultural exchange has played a significant role in the app's widespread adoption, particularly among millennials who value connectivity and diverse interactions. Furthermore, WhatsApp has proven essential for organizing and mobilizing social movements, enabling the rapid dissemination of information and facilitating coordination among activists. Its ability to serve as a tool for social change underscores its influence, which goes far beyond simple communication, and highlights its importance in contemporary societal dynamics [68].

In short, WhatsApp's integration into everyday life has changed the way individuals communicate, businesses operate, and cultures interact. This has solidified its position as an essential tool in the digital age.

2.2.7.2 WhatsApp Challenges

WhatsApp faces several major security and privacy challenges. These include :

1. **Privacy Concerns**
   Although it uses end-to-end encryption through the Signal protocol, WhatsApp collects extensive metadata, including user interactions and device information, raising privacy concerns. This data may be shared with its parent company, Meta (formerly Facebook), raising concerns about user surveillance and data misuse [69].

2. **Disinformation**
   WhatsApp is often used to spread misinformation, particularly in political or public health contexts. Studies have shown that misleading messages can spread quickly, leading to serious consequences, including physical violence in some cases. WhatsApp has implemented measures to limit the sharing of frequently forwarded messages to combat misinformation, but the challenge remains significant [70].

3. **Phishing and social engineering**
   WhatsApp is a common target for phishing attacks, where attackers pose as trusted contacts to extract sensitive information. Users are often tricked into sharing verification codes or clicking on malicious links, which can compromise their account [71].

4. **Malware distribution**
   WhatsApp faces serious malware distribution issues through three main channels. First, cybercriminals share malicious links in messages that can infect users' devices. Second, malicious attachments can be downloaded and executed by users without their knowledge. Finally, vulnerabilities in group chats allow malicious content to spread quickly if a member shares it, increasing the risk of widespread infection [71].

5. **Competition**
   WhatsApp faces significant competitive challenges in the messaging app market, primarily from emerging platforms such as Telegram, Signal, and Discord. These apps offer unique features such as enhanced privacy, expanded group functionality, and seamless integrations with other services. These competitors often cater to specific demographics and offer a more customizable user experience, which can attract users looking for tailored solutions. Furthermore, growing concerns about data privacy and competition from local apps in various international markets further complicate WhatsApp's position. To remain relevant, the app must continually innovate, address privacy concerns, and improve user engagement while taking into account the diverse preferences and needs of its global user base [72].

To address these challenges, continued efforts are needed to strengthen security measures, protect user privacy, combat misinformation, and effectively navigate the competitive and regulatory landscape.

Conclusion

Instant messaging apps have revolutionized communication, offering a fast, convenient, and accessible alternative for everyone. WhatsApp, with its user-friendly interface, innovative features, and end-to-end encryption, has established itself as one of the most popular apps in the world. Its impact on personal communication, interpersonal relationships, and professional interactions is undeniable.

Despite challenges and criticism, WhatsApp continues to innovate and grow by offering new features and adapting to its users' needs. Its role in emerging markets and its social and cultural influence make it a major player in global communication. WhatsApp's future looks bright, with planned new features and a continued expansion of its user base. The increasing integration of artificial intelligence in messaging apps, the growing importance of privacy and security, and the potential impact of emerging technologies like Web3 are all factors that will shape the future of instant messaging.

Chapter 3

Study of existing solutions

Introduction

Parenting in the digital age presents unique challenges due to rapidly evolving technology. While online platforms provide valuable learning and socialization opportunities, they also expose children to risks such as inappropriate content, cyberbullying, identity theft, and privacy issues. To address these challenges, implementing parental controls has become essential to ensure children's online safety and promote a healthy digital experience. Advanced tools and monitoring apps allow parents to manage their children's activities, set screen time limits, and filter content to help them navigate the digital world safely and responsibly [73].

3.1 Is WhatsApp safe for children ?

In some circumstances, WhatsApp can be a useful tool for staying connected with friends and family. However, it poses significant risks to children because it lacks message controls and allows users to send and receive inappropriate, violent, and sexually explicit photos and texts. Additionally, WhatsApp does not require age verification, raising concerns about its safety for younger users. While the app can facilitate effective communication, parents and guardians should be aware of these potential dangers to make informed decisions about their children's digital lives. Understanding the risks associated with WhatsApp is essential for families who want to protect their loved ones while navigating the complexity of online interactions [74, 75]. Here are some of the main risks associated with children using WhatsApp [75]:

→ **Inappropriate content :** Without filters, users can send any type of media, including through the "view once" feature, which allows photos and videos to be viewed only once before disappearing. This makes it difficult for parents to monitor or preserve evidence of harmful content. The "disappearing messages" feature, which automatically deletes messages after a set period of time, can be disabled by parents, but children can easily re-enable it when unsupervised. As a result, some children may send risqué or intimate content in the mistaken belief that it will disappear permanently, without realizing that screenshots can still be taken.

→ **Foreigners and scammers :** Members of a WhatsApp group can easily copy and share chat links, allowing anyone to join the group without prior permission. This allows malicious strangers or scammers to infiltrate private groups, initiate private conversations, or invite members to join other potentially dangerous groups. Furthermore, children who randomly join public online chat groups risk being contacted by unknown and potentially dangerous people.

→ **Cyberbullying :** Group messaging can facilitate cyberbullying by allowing individuals to target others within the group, or by spreading inappropriate images and rumors. For example, a bully may publicly share a person's WhatsApp number, resulting in the victim receiving a barrage of hurtful messages from strangers. Another particularly damaging form of cyberbullying is "doxing", which involves publicly sharing personal information about the victim with the intention of harming or embarrassing them.

→ **Addictiveness :** Parents should also be aware that chatting, as well as sending emoticons and GIFs, can be addictive and lead their children to spend excessive time on their phones.

→ **Location sharing :** Children can share their live location with their contacts. However, this feature can be risky if shared with people outside their trusted circle.

While WhatsApp is a valuable communication tool, it poses certain risks for children. To ensure their safety, parents should use parental control apps to monitor their children's activities on the platform. In the digital age, children's online safety must be a priority, hence the need for parental controls.

3.2 Main Features of Parental Control Apps

When choosing a parental control app, it's important to prioritize features that will protect your children's online experience and well-being. Here are some key features to consider [76]:

→ **Monitoring and managing application usage :**App management and blocking features are essential in parental control apps because they allow parents to restrict access to certain apps, such as social media and messaging platforms, that may be distracting or inappropriate. These tools often offer the ability to temporarily block certain apps during school hours or other designated times, which helps promote focus and create a more productive environment for children.

→ **Location tracking :** Parental control apps offer location tracking and geofencing features that allow parents to monitor their child's location in real time. Geofencing allows you to create virtual boundaries and send alerts when a child enters or leaves certain areas, such as school or home. This feature can help ease parents' concerns about their child's whereabouts, especially in cases of communication gaps, by alerting them when their child enters or leaves certain areas.

→ **Activity reports :** Detailed logs and reports give parents insight into their child's device usage, including app usage, time spent in each app, and web browsing history. This information can help identify concerning patterns or behaviors, allowing parents to better understand their child's digital habits and make informed decisions about policies and restrictions.

→ **Family Account Management :** A simplified family account lets you manage all your children's devices and accounts from a single interface, making it easy to set up and customize controls for multiple users. These apps are designed to be user-friendly and easy to navigate, so even non-tech-savvy users can set up and manage controls quickly and without frustration.

While these features are fundamental prerequisites for effective parental controls, they play a crucial role in helping parents protect their children from various digital dangers.

3.3 Popular Parental Control Apps for WhatsApp

Given the number of options available, each offering a different range of features such as text message monitoring, real-time location tracking, and usage restrictions, parents should consider several key factors to make an informed decision, including the app's ease of use, price, and compatibility with their child's device, to find the solution best suited to their specific needs [77].
Here are some popular parental control apps that can help parents monitor and keep their children safe while using WhatsApp [78]:

3.3.1 Bark

Bark is a parental control app that monitors social interactions on platforms like WhatsApp, using advanced AI to detect risks like cyberbullying and explicit content. It provides real-time alerts, allowing parents to intervene quickly when concerning activity is identified. While Bark excels at alerting parents to potential risks, its subscription cost is higher than basic parental control apps, which may be a disadvantage for some families. Additionally, it focuses less on screen time management than other solutions. New users can get a free trial, and the Bark Premium plan is priced at $14.00 per month and offers comprehensive monitoring and alert features.

3.3.2 mSpy

mSpy is a monitoring app designed to allow parents to discreetly monitor their children's online activity, particularly on platforms like WhatsApp. It excels at tracking interactions with text messages, messaging apps, and emails, and provides insight into sent and received WhatsApp messages, as well as access to exchanged media files and contact information.
While mSpy offers features like inappropriate content blocking and a user-friendly interface, it has limitations, including the lack of a free version, a money-back guarantee, and advanced parental controls. Subscription prices are set at $48.99 per month, $83.99 per quarter, and $139.99 per year, making cost a consideration for potential users.

3.3.3 FamiSafe

FamiSafe is a parental control app that helps parents monitor their children's online activities. It offers real-time location track-

ing, geofencing, and WhatsApp usage management, making it ideal for keeping children safe in both digital and physical environments. Its key features include GPS tracking, customizable alerts, screen time management, content filtering, app blocking, and social media monitoring.
The app is easy to use, supports multiple devices, and is simple to set up. However, some features require a subscription, and continuous GPS tracking can reduce battery life. The annual subscription costs $59.99 and supports an unlimited number of devices.

3.3.4 Qustodio

Qustodio is a comprehensive parental control app designed to help parents monitor their children's online activities, including popular messaging apps like WhatsApp. It allows for content filtering, screen time management, and detailed tracking to promote safe digital behavior. Specifically, it tracks WhatsApp usage metrics such as time spent, message volume, and contacts, while also offering the ability to block inappropriate content and set limits on app usage.
However, while the basic version is free, many features, including WhatsApp monitoring, require a premium subscription. The installation process can also be complex for less technically savvy users. Premium subscriptions cost $99.95 per year ($8.33/month) for the full plan.

In conclusion, while parental control apps offer valuable features to keep children safe online, their high cost and limited free options can be a significant barrier for many families. The importance of these tools for protecting young users is clear, but the lack of affordable plans may lead parents to seek alternative solutions. Ultimately, it is essential for families looking to protect their children's digital experience to strike a balance between the need for effective control and budgetary considerations.

3.4 Challenges and limitations

→ **Technical requirements :** Some parental controls require rooting (for Android) or jailbreaking (for iOS) devices to access deeper system data, which can void warranties, create security vulnerabilities, and may not be practical for all users [79].

→ **Cross-platform nature of WhatsApp :** The cross-platform nature of WhatsApp, which is available on Android, iOS, and web platforms, poses a significant obstacle for parental control apps seeking to monitor user activity. Indeed, the

app's widespread presence across multiple devices and interfaces complicates efforts to implement effective monitoring measures.

→ **A false sense of security :** Parents can develop a false sense of security by relying too much on parental control apps, which can lead them to neglect open and important discussions about online safety with their children [80].

→ **Lack of parental control :** WhatsApp doesn't have built-in parental controls, which can make it difficult for parents to monitor conversations or intervene if necessary. This lack of control can sometimes lead children to use the platform intentionally, knowing that their parents are less likely to monitor their activities [75].

→ **Data breaches :** A major challenge for parental control apps that handle sensitive children's information, as demonstrated by the 2020 KidGuard hack that exposed security vulnerabilities. Such incidents can lead to identity theft and a loss of trust between parents. To mitigate these risks, parents should choose reputable providers with strong security practices, prioritize apps that use encryption and enforce transparent privacy policies, and ensure software is regularly updated [81].

Conclusion

In conclusion, while the need for parents to monitor their children's phone usage - especially on platforms like WhatsApp - has never been more pressing, the high cost of parental control apps poses a significant barrier. Many parents feel compelled to invest in these tools for their benefits and features, but these subscriptions can be financially overwhelming, especially for middle- and lower-income families and larger households facing device limits. As a result, these essential safety measures risk remaining out of reach of those who need them most.

Chapter 4

Conceptual study

Introduction

The conceptual design phase is a key step in the software lifecycle. It addresses the needs identified during the preliminary analysis, allowing the system to be developed to be specified and its operation to be described. This step provides essential support for implementation.
In this chapter, we will present in detail the design phase of the project through our architecture and with the help of UML diagrams.

4.1 Objective

The overall objective of this thesis is to address the growing need for ethical and technically robust monitoring solutions for instant messaging platforms, focusing specifically on WhatsApp as one of the most widely used solutions worldwide. The aim is to design a monitoring system that allows access to communication data within defined frameworks: child protection, risk prevention, while navigating the complexities imposed by the security mechanisms of operating systems and the applications themselves. This thesis aims to explore the limitations of current approaches and propose an innovative architecture.

In this perspective, this chapter focuses on analyzing the technical challenges related to accessing WhatsApp messages on the Android platform and on presenting in detail the architecture of the proposed solution.

Before diving into designing a monitoring solution, it's essential to understand the security landscape of the Android platform. The constant evolution of this operating system has a direct impact on the possibilities for accessing application data.

4.2 Android Security Challenges

Android 13 (API level 33), and subsequent releases, have enhanced security features to strengthen user privacy and overall platform security [82] to the detriment of application accessibility to data.

4.2.1 Restricted access to application data (/data/data)

Android 13 and later versions have maintained and strengthened the principle of sandboxing. Access to an app's private data directory (/data/data/[package.name]), where the most sensitive information such as databases and keys are stored, remains prohibited to other apps without root privileges [83].

4.2.2 "Restricted Settings" feature

This is a security measure designed specifically to counter the abuse of particularly dangerous permissions by potentially malicious applications, especially those installed outside of official app stores (sideloading) [84].
The two permissions primarily targeted by this restriction are:

→ **Access to accessibility services :** Because of their ability to read screens and intercept input, these services are frequently hijacked by malware to steal credentials, banking information, bypass two-factor authentication, and spy on communications [85].

→ **Accessing the notification listener :** This permission allows you to read the contents of all notifications [84].

4.3 Prerequisites for WhatsApp monitoring

To perform meaningful monitoring of WhatsApp conversations, access to the following is essential [86]:

→ **Message database :** This is the database where WhatsApp stores message history, metadata (timestamps, sender/recipient), group information, etc. It resides in the shared storage /media/com.whatsapp/WhatsApp/databases/msgstore.db.

→ **Encryption key :** The msgstore.db database is encrypted to protect message privacy. It is stored separately in WhatsApp's private directory: /data/data/com.whatsapp/files/key.

→ **Media files :** Exchanged media files (images, videos, audio, documents) are usually stored in shared storage, in a folder accessible with the new media permissions /media/com.whatsapp/WhatsApp/media/.

Android 13's enhanced security mechanisms make it virtually impossible for a third-party app to access this critical data. As a result, traditional WhatsApp monitoring solutions are rendered ineffective, and developing reliable alternatives has become prohibitively difficult.

The main obstacle lies in accessing the encryption key. Without this key, the msgstore.db message database, although accessible, remains unusable because it is encrypted. Android's enhanced sandboxing measures prohibit direct access to this private directory by an unprivileged third-party application. This situation renders monitoring approaches that relied on simply recovering the database and its key obsolete.

Faced with the inability to access WhatsApp data on an Android device, an alternative approach is to gain privileged access to the system via rooting.

4.4 Root Android

4.4.1 Definition

Rooting an Android device is the technical process of allowing the user to gain privileged control, known as "root" or "superuser" access, over the Android operating system. Because Android is based on the Linux kernel, root access is analogous to administrator permissions on Linux or Unix systems [87].

4.4.2 Benefits

Users root their devices for a variety of reasons, including :

→ **Advanced customization :** Change the appearance (themes, fonts), remove preinstalled applications ("bloatware") [88].

→ **Installation of Specialized Applications :** Use applications that require root access (advanced firewalls, comprehensive backup tools like Titanium Backup [89], deep automation applications) [90].

→ **Performance Improvement :** :Adjust CPU frequency (overclock for more speed, underclock to save battery) [90].

→ **Complete Backup and Restore :** Create full system images (Nandroid backups) via custom recovery [91].

4.4.3 Risks induced by rooting

Despite its benefits, routing carries significant and multiple risks that must be carefully considered.

→ **Increased security vulnerabilities :** Rooting disables or bypasses many of Android's built-in security mechanisms, including application sandboxing. A malicious app that gains root access can access all data on the device, modify critical system files, install other malware, log keystrokes, and more without restrictions [88].

→ **Cancellation of Warranty :** Nearly all manufacturers and operators consider routing to be unauthorized software modifications and a violation of their terms of service. As a result, the device's hardware and software warranty is usually voided [90].

→ **System Instability and Risk of "Bricking" :** The routing process, if executed incorrectly (bad files, bad procedure, interruption), can lead to serious errors rendering the device completely unusable – a state called "bricked" [88].

→ **Refusal of service :** Many security-sensitive applications, including banking applications, mobile payment applications, some video streaming applications, and enterprise applications, incorporate root detection mechanisms [92]. If they detect that the device is rooted, they may refuse to work or limit their functionality for security reasons [88].

→ **Potential data loss :** Unlocking the bootloader, often a necessary preliminary step for rooting, systematically causes a factory reset of the device, erasing all user data [93]. Errors during the process of flashing a recovery or ROM can also corrupt the data partition and lead to data loss if no prior backup has been made [88].

→ **Problems with official updates :** Rooted devices generally cannot install official software updates provided (OTA) by the manufacturer or carrier [93].

The main and almost unique advantage in this context is to overcome all data access restrictions imposed by the operating system, allowing direct recovery of the key file necessary for decrypting WhatsApp messages.

Ultimately, while device routing does indeed allow bypassing Android's restrictions and directly accessing the encryption key and decrypting the WhatsApp message database, thus solving the initial problem of direct data access, the associated risks are still considerable. This method exposes the device to increased security vulnerabilities, warranty voiding, potential system instability and the risk of "bricking", as well as denial of service for certain critical applications. These major drawbacks make device routing an unviable and undesirable solution for a consumer application.

Therefore, while routing offers a technical solution to access, priority must be given to developing parental control solutions that do not require such a system compromise, while remaining effective in their monitoring mission. Thus, a new problem clearly emerges : *" How is it possible to access the content of WhatsApp messages reliably, without resorting to device routing and, therefore, without being able to access the database ? "*.

4.5 Current non-root monitoring techniques

With the increased restrictions introduced by Android 13 and later, particularly regarding direct access to app data and the granting of sensitive permissions, rootless surveillance methods have had to adapt. They now rely heavily on indirect techniques, rather than attempting to access stored databases.

4.5.1 Notification Mirroring

This technique, using the Notification Listener permission, remains a viable method on Android 13+. Apps granted this permission can read the contents of WhatsApp notifications as soon as they arrive [94].Commercial applications like AirDroid [94] and potentially other parental control tools use it to provide insight into messaging activity.

→ **Advantages :** Less resource intensive than screen reading, can work even if WhatsApp is not in the foreground.

→ **Disadvantages :** Restricted settings, limited content, ineffective if notifications are turned off.

4.5.2 Accessibility Services (Screen Reading)

Accessibility Services allow you to read text content displayed on the screen [95]. Modern surveillance applications (often marketed as parental control or spyware) exploit this ability to extract the text of WhatsApp conversations directly from the screen when

the application is in use by the target. They analyze the layout of user interface elements to identify incoming and outgoing messages, contact names, and potentially the time of messages [94].

→ **Advantage :** Allows text content to be captured as it appears to the user, providing more context than notifications.

→ **Disadvantages :** Depends on the structure of the WhatsApp interface, performance degradation.

4.5.3 Screen Mirroring/Recording

Android's native screenshot method, originally designed for legitimate screen recording and sharing, can be hijacked by surveillance apps. These apps, such as AirDroid Parental Control, mSpy, and KidsGuard Pro, offer screen recording or live streaming (mirroring) to a remote device [94].

→ **Advantage :** Captures the entire screen, including full messages, images, videos, statuses, and the WhatsApp interface as seen by the user.

→ **Disadvantages :** Requires explicit user authorisation each time capture is started, consumption of resources (CPU, bandwidth, storage).

4.5.4 Keylogging

Aims to capture everything the user types via Accessibility Services. An Accessibility Service can be configured to listen for text input events in user interface fields, including WhatsApp message fields [85].

→ **Advantage :** Capture typed messages before sending.

→ **Disadvantages :** Restricted settings, does not capture voicemails, limited content.

4.5.5 Connecting to WhatsApp Web

This involves using a dedicated server to run an instance of WhatsApp Web in an isolated environment. This instance is connected to the target WhatsApp account. Automated scripts extract data from the WhatsApp Web interface. This information is presented on a dashboard accessible to parents.

→ **Advantage :** Provides full access to conversations.

→ **Disadvantages :** Security risks related to account access, login persistence, does not capture calls.

4.6 Limitations of existing approaches

An analysis of current rootless monitoring techniques reveals several methods for attempting to monitor WhatsApp activity. However, each of these approaches has limitations that hinder the achievement of comprehensive and reliable monitoring as targeted by our project :

→ Notification mirroring only captures received messages and is completely ineffective if the user has disabled notifications for WhatsApp or for specific conversations.

→ Screen reading drastically impacts device performance.

→ Screen mirroring/recording requires explicit user permission for each session and is very resource intensive.

→ Keylogging is also problematic. This method severely lacks context for sent messages and does not capture received messages.

→ Finally, linking to WhatsApp Web, while potentially providing more comprehensive access to conversations, introduces significant security risks by requiring direct access to the user's WhatsApp account.

Beyond the limitations of the software circumvention techniques analyzed, the search for access to the WhatsApp encryption key via Android vulnerabilities proved unsuccessful.

As existing methods did not meet the requirements of reliability, completeness, and security, a new approach was required. The objective was therefore to design a solution capable of overcoming these obstacles, aiming for integrated, secure, and efficient monitoring, the architecture of which is presented below.

4.7 Proposed solution

4.7.1 Introduction

Given the limitations of existing monitoring methods, the security risks posed by routing, and the difficulties of direct data access imposed by recent Android versions, an

alternative approach was necessary. The solution we designed and developed is based on leveraging WhatsApp's native *"Connected Devices"* feature. This method allows our application, installed on the target device, to act as an authorized "connected device," receiving message and data streams directly from the main WhatsApp instance on the source phone. A dedicated web interface then allows parents to access the collected data and manage monitoring settings.

4.7.2 Architecture

Our solution adopts a distributed client-server architecture. It is based on three main components: the application installed on the target device, a central application server, and a parental dashboard. The Figure 4.1 illustrates the interactions between these components.

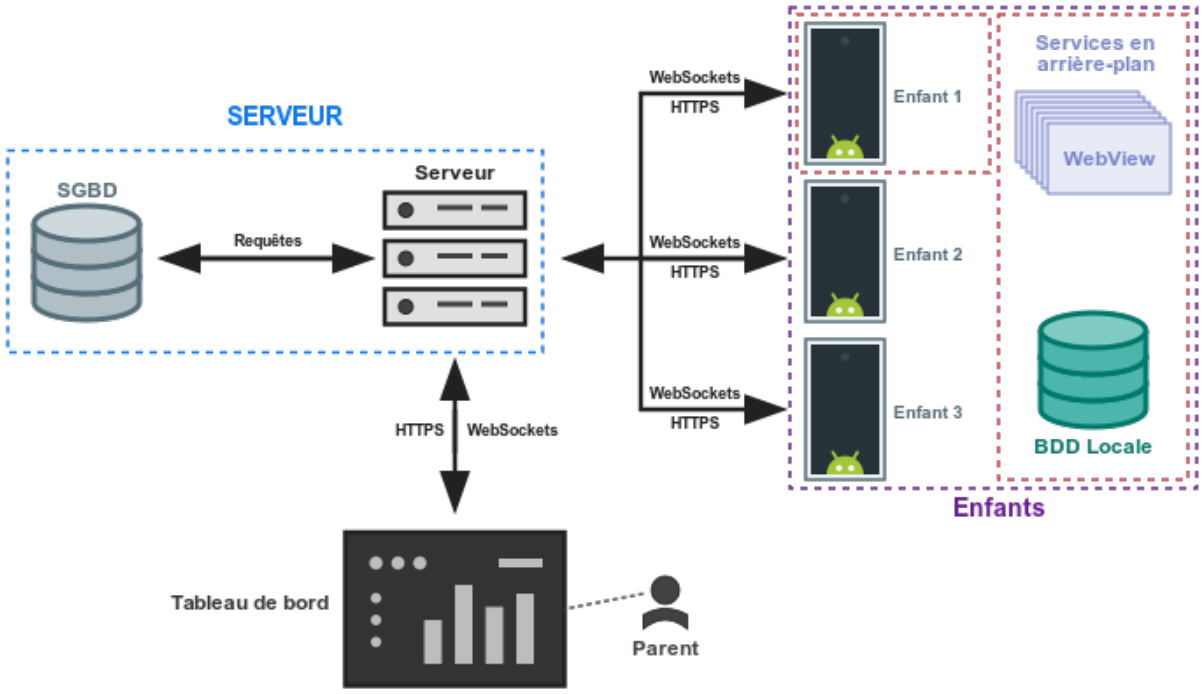

Figure 4.1: Solution architecture

4.7.3 Components

4.7.3.1 Central server

→ Hosts a server application responsible for core business logic, including secure authentication of target applications and dashboard users.

→ Manages communications and data flows between target applications and parental dashboards:

- Specific management of WhatsApp communication data (messages, media, contact details): The server acts as a secure relay and does not store this information persistently. When a parent requests to view this data, the server forwards the request to the target application. The retrieved information is then routed via the server to the parental dashboard for real-time display, without being stored on the server's DBMS.
- Processing parent-initiated commands: The server receives commands from the dashboard and securely forwards them to the relevant target application for execution.

→ A Database Management System (DBMS) is used exclusively to store :

- Information relating to user accounts (family accounts, child and parent profiles, associated devices).
- Monitoring configurations specific to each child profile.
- The alerts generated.
- Operation and usage metadata (screen time, location).
- Call recordings made.

→ A caching mechanism, powered by an in-memory store, to serve as a fast access layer in front of our main DBMS.

4.7.3.2 Target application

→ Installed on the child's device, this application integrates services running in the background.

→ Integrates a client that establishes a secure connection (real time / client server) with the central server.

→ Use a web page to interact with a local instance of WhatsApp Web.

→ Requires certain special permissions to function optimally.

→ WhatsApp data (messages, media, accounts, etc.) is dynamically retrieved via the established connection and transmitted directly to the server only when the parent requests it from the dashboard.

→ A local database (DB) is used for :

- WhatsApp screen time storage.
- Temporarily cache data waiting to be transmitted to the server.
- Maintain specific configurations like keyword list for alerts or scheduling settings for application blocking.

4.7.3.3 Parental Dashboard

→ Provides secure access to view data collected by the target application.

→ Allows management of monitoring configurations and initiation of remote control actions.

→ Allows the management of multiple child profiles in an individualized manner.

4.7.4 Modeling the solution

Solution modeling aims to represent the functional and structural aspects of the system. It begins by identifying key user interactions with the system through use case diagrams.

4.7.4.1 Use Case Diagrams

The following use case diagrams illustrate the functionality offered by the system from the perspective of the main actors: the Parent and the Child.

The Figure 4.2 below is a diagram of the parent's interactions with the dashboard. This diagram details the features accessible to the parent via the dashboard. The main interaction is "Access the Dashboard", which determines access to other features. This initial access is broken down into: Registration or login followed by a session recording.

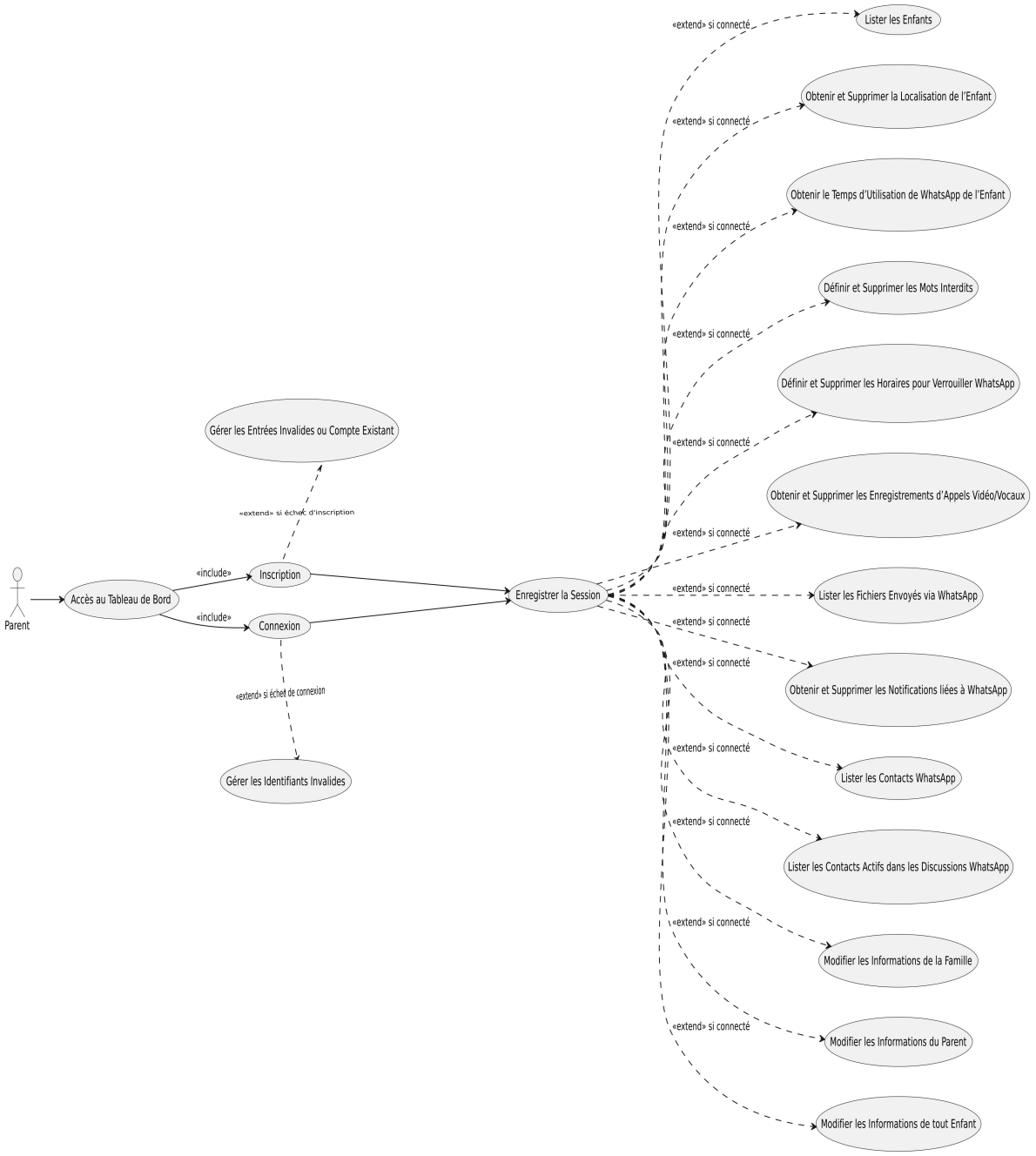

Figure 4.2: Use Case Diagram - "Parent" Actor

In the Figure 4.3 below are the main interactions of the child with the application installed on his device. The process begins with access to the application, this access can lead to registration or login if no session is recorded. The child can then see the basic information of his profile, the time spent on WhatsApp and the time slots of use or restrictions of WhatsApp.

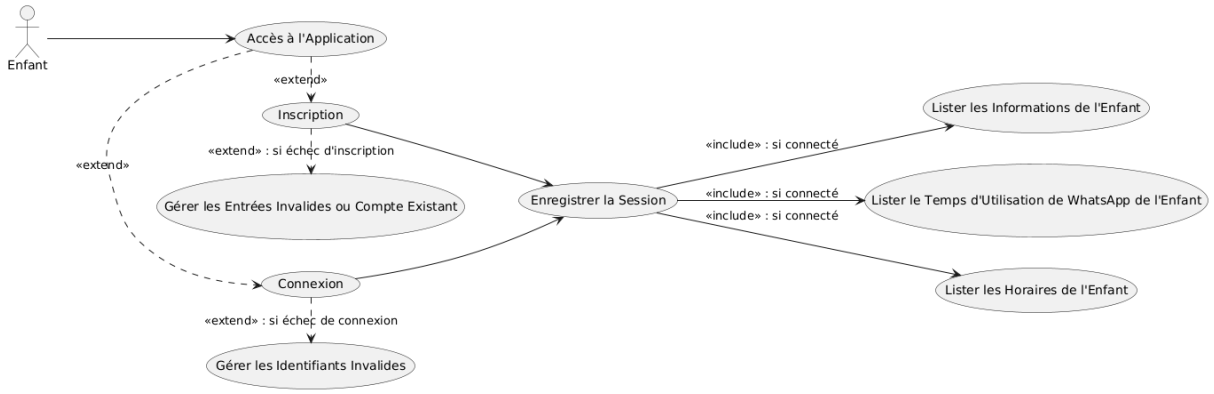

Figure 4.3: Use Case Diagram - "Child" Actor

4.7.4.2 Logical Data Model

The LDM specifies the persistence structure of data managed by the central server :

Relational schema :

User (**id**, username, password, email, first_name, last_name, is_staff, is_active, is_superuser, last_login, date_joined)
BaseUser (**id**, gender, birthday, phone_number, first_ip, ip, created_at, deleted, photo)
Parent (id*, user_id*, is_confirmed, conform_code, qr_code, qr_image, photo)
Child (id*, user_id*, phone_locked, photo)
Family (**id**, name, about, father_id*, mother_id*, deleted, created_at, last_updated, qr_code, qr_image, photo)
Family_kids (**family_id***, **child_id***)
ResetPassword (**id**, username_email, phone_number, code, created_at, checked)
HourlyUsage (**id**, hour, usage_seconds)
UserUsage (**id**, **child_id***, date)
UserUsage_hourly_usages (**userusage_id***, **hourlyusage_id***)
ChildLocation (**id**, **child_id***, latitude, longitude, accuracy, timestamp, created_at, is_deleted)
Day (**id**, value)
Schedule (**id**, **child_id***, name, start_time, end_time, start_date, end_date, created_at, is_deleted)
Schedule_days (**schedule_id***, **day_id***)
Notification (**id**, **child_id***, title, content, timestamp, type, is_read, is_deleted)
BadWord (**id**, word)
ChildBadWords (**id**, **child_id***)
ChildBadWords_bad_words (**childbadwords_id***, **badword_id***)

ChildCallRecording (**id**, **child_id***, date, timestamp, is_deleted, is_read, record_file, recording_type)

4.7.4.3 Sequence diagrams

To illustrate the dynamic behavior of the system and the interactions between its main components, the following sequence diagrams are presented for typical usage scenarios :

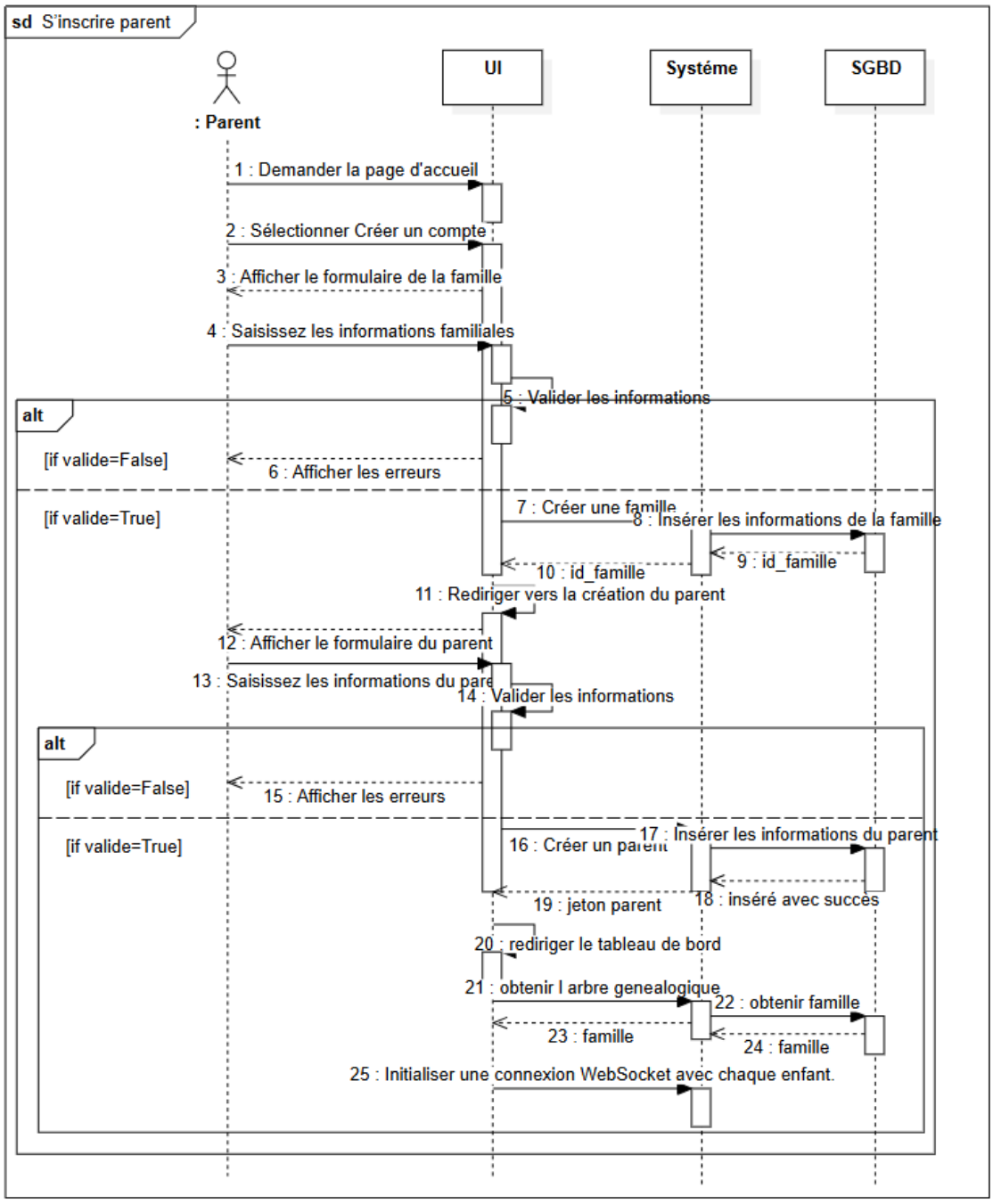

Figure 4.4: Sequence Diagram - Parent Registration

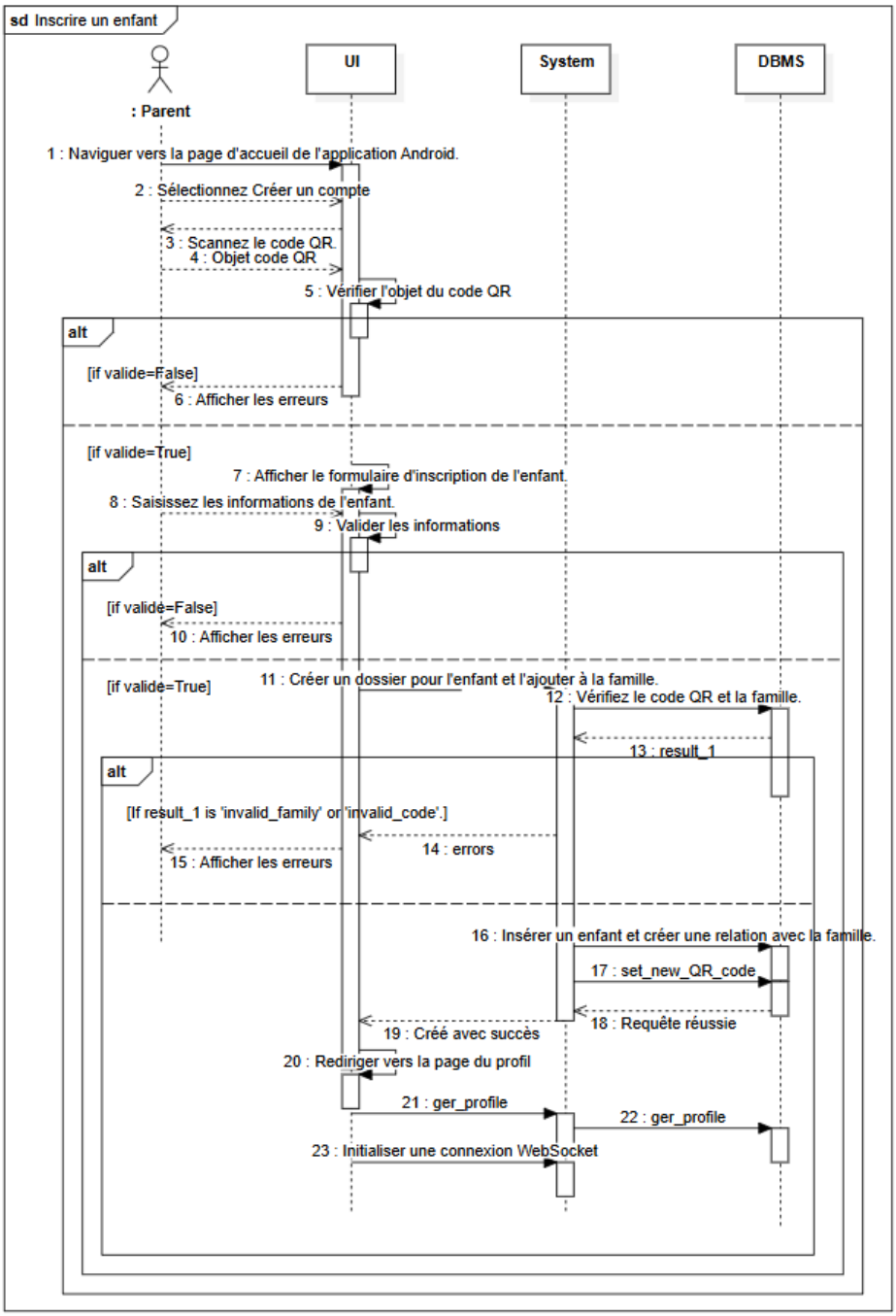

Figure 4.5: Sequence Diagram - Child Registration

4.7.4.4 Class diagram

The class diagram is generally considered the most important in object-oriented development. It represents the conceptual architecture of the system :

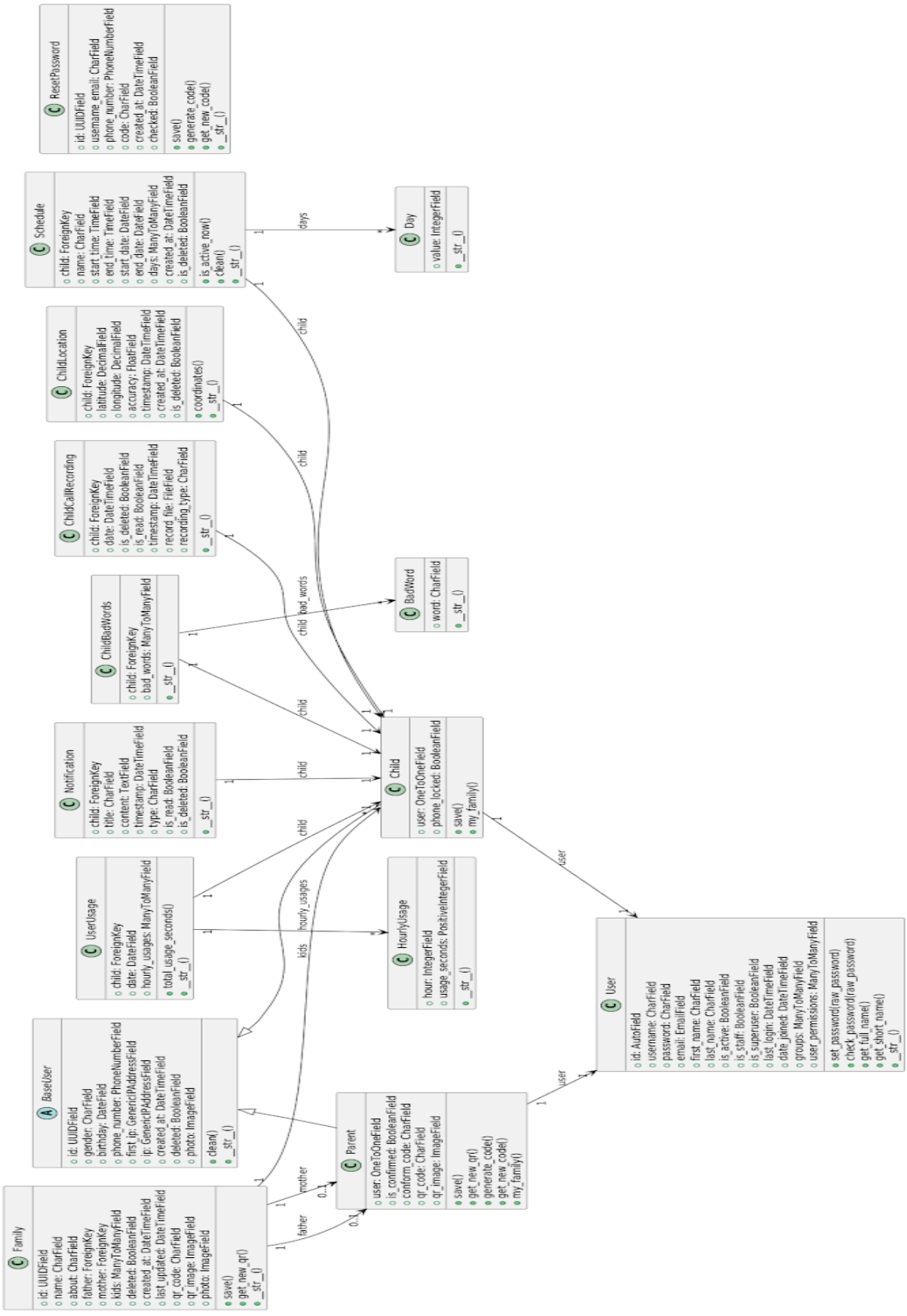

Figure 4.6: Class diagram

4.7.5 Scraping algorithms

Below we describe some of the main web scraping algorithms used in this project, to provide an overview of their implementation and functionality.

Algorithm 1 Extract WhatsApp contacts from chat list

Data : List of visible WhatsApp chats
Result : JSON array containing contact information
Simulate the `Down Arrow` key to activate navigation
Focus the search bar
Retrieve the total number of visible chats
Initialize an empty `contacts` array
foreach *chat i in the liste* **do**
  Wait $10 \times i$ ms
  Select the active chat
  if *chat is valid* **then**
    Extract `name`, `timestamp`, `last message`, `unread count`, `icon`
    Add to the `contacts` array
  else
    Skip the chat
  end
  Simulate pressing `Down Arrow`
end
Detect the WhatsApp interface language
Define translated labels: `"Unread"`, `"Close chat"`, etc.
foreach *chat i in the list* **do**
  if *chat has a green badge* **then**
    Wait 300 ms
    Open the context menu
    Click on `"Mark as unread"`
  end
end
Wait 300 ms
Click on the chat menu icon
Click on `"Close chat"`
if *the chat panel exists* **then**
  Reset the scroll to the top of the list
end
Convert `contacts` to JSON
Save in local storage under the key `"WHATSAPP_CONTACTS"`

This algorithm automates the extraction of contact information from WhatsApp chats by browsing the chat list via keyboard navigation ("Down Arrow" key). For each chat, it collects the name, timestamp, last message, number of unread messages, and contact icon. It also allows actions such as marking a chat as unread or closing it, while handling

WhatsApp interface language variations. Short timeouts are included to ensure execution stability, and the extracted data is saved locally for future use.

4.7.6 Features

→ **Access to text messages exchanged on WhatsApp**
Allows parents to ensure their child's safety by identifying and preventing potential risks such as cyberbullying, exposure to inappropriate content or contact with malicious strangers. To retrieve messages from the conversation, the target application uses a web scraping technique. A dedicated service launches a web view that simulates access to WhatsApp Web. Scripts then analyze the content displayed in this WebView to extract the exchanged messages before transmitting them to the parent application.

→ **Access to incoming and outgoing call history**
Provides parents with visibility into their child's voice interactions, helping to identify unknown contacts, calls at inappropriate times, or a frequency of calls that could indicate a bullying or addiction issue. To do this, the target application is configured to listen and intercept notifications generated by WhatsApp during calls.

→ **Viewing call recordings**
Allows parents, facing major concerns, to access full recordings (audio and video) of calls made via WhatsApp. When a child participates in an audio or video call on WhatsApp, the target app is designed to enable recording. These multimedia recordings (audio/video files) are then temporarily stored on the device securely before being transmitted to the central server.

→ **Access to contacts**
Allows parents to know who their child is in contact with or may come into contact with on WhatsApp. This helps identify unknown or potentially suspicious contacts. To retrieve the WhatsApp contact list, the target application uses a web scraping technique. A dedicated service launches a web view that simulates access to WhatsApp Web. Scripts then analyze and interact with the content displayed in this WebView to extract the complete list of available contacts.

→ **Access to exchanged media**
Allows parents to view the media their child sends and receives on WhatsApp. Images, videos, and voice messages can reveal context, emotions, or crucial information not apparent in text alone (e.g., visual cyberbullying, sexting, exposure to graphic

content, potentially harmful material). Access to this media is therefore vital for comprehensive protection against online risks. To access the exchanged media, the target application uses the device's system permissions to access media files. It allows parents to identify excessive use or potential addiction by providing clear statistics on the child's periods of activity.

→ **Access to time spent on WhatsApp**
This feature uses app usage history permissions on Android to collect accurate information about time spent on WhatsApp. It allows parents to identify excessive use or potential addiction by providing clear statistics on the child's periods of activity.

→ **Access to device location on demand**
The app uses GPS and a real-time connection to locate the child. If GPS is disabled, parents are notified. This feature is essential for ensuring safety and knowing where the child is at all times.

→ **Block a contact or Leave a group**
Allows parents to intervene directly to protect their child in the event of proven harassment, contact with a malicious person, or exposure to a WhatsApp group distributing inappropriate or dangerous content. This feature relies on scraping techniques to automatically perform certain actions, such as blocking a contact or leaving a WhatsApp group, directly from the child's phone.

→ **Block access to WhatsApp**
The app allows you to block access to WhatsApp on your child's phone, based on rules set by the parent. This blocking is enforced by a PIN system, preventing the child from changing settings or bypassing the restriction. Two blocking modes are available : Instant blocking, Scheduled blocking.

→ **Keyword-based alerts**
Allows parents to be automatically notified when specific words or phrases, which they have previously defined, are detected in their child's WhatsApp messages.

4.8 Choice justification

This approach was favored for several reasons :

→ **Full and reliable access :** Unlike notification capture (limited to previews) or screen reading (vulnerable to updates), our solution allows for more comprehensive and structurally more stable access to conversations.

→ **Independence of restricted permissions :** It does not directly depend on the most problematic permissions under Android 13+ like Accessibility Services or Notification Listener, which are subject to "Restricted Settings" during manual installations (sideloading).

→ **No root required :** The solution works without requiring device rooting, thus avoiding major security risks, warranty voiding, and compatibility issues associated with rooting.

Conclusion

This chapter defined the design of our monitoring solution for WhatsApp, a challenge complicated by Android's robust security mechanisms. After realizing the inadequacy of traditional approaches - routing was too risky and existing rootless techniques were too limited in reliability and functionality - we had to innovate.

Our solution leverages WhatsApp's "Connected Devices" feature. We presented its client-server architecture, detailed modeling (use cases, data model, interaction sequences), and all the monitoring features it offers.

Chapter 5

Implementation

Introduction

After the conceptual study of the previous chapter, we now enter the phase of realizing and implementing our application. We will begin by examining the working environment we have chosen, then we will review the tools and techniques we used to complete this project. Finally, to better illustrate the result, we will highlight the main features using screenshots.

5.1 Work Environment

To ensure efficient development, we have carefully selected a set of tools and resources tailored to our needs. Here are the main elements of our technical environment :

5.1.1 Programming languages

5.1.1.1 Kotlin

Kotlin is a modern, statically typed programming language that is fully interoperable with Java. It was designed to improve developer productivity and code security. It is the preferred language for Android app development because it offers features such as null safety, coroutines for streamlined asynchronous programming, and a robust library ecosystem. Kotlin minimizes boilerplate code, improves code expressiveness, and is supported by tools such as Android Studio and IntelliJ IDEA, making it an optimal choice for developing reliable and maintainable applications [96].

5.1.1.2 Python

Python is an interpreted, readable, and versatile programming language that supports object-oriented, functional, and procedural paradigms. Ideal for web development, data science, automation, and artificial intelligence, it stands out for its clear syntax, extensive standard library, and rich ecosystem (Django, Flask, NumPy, Pandas), which optimizes development. Easy to learn, scalable, and supported by a large community, it is popular with beginners and experts alike [97].

5.1.1.3 Javascript

JavaScript is a versatile scripting language essential for creating dynamic and interactive features on web pages. It enables features such as real-time content updates, interactive maps, and animations, making it a fundamental element of modern web development. JavaScript works alongside HTML and CSS, forming the core technologies of the web. Its flexibility allows developers to significantly enhance the user experience by implementing complex features that go beyond static content [98].

5.1.2 Frameworks used

5.1.2.1 Django

Django is a high-level Python web framework designed for rapid development and clean, pragmatic design. Free and open-source, it allows developers to create secure and easy-to-maintain websites. Its key features are an automatic administration interface, an ORM (Object-Relational Mapping) system, and built-in security measures against common threats such as SQL injection and cross-site scripting. Django's architecture promotes the DRY (Don't Repeat Yourself) principle, which streamlines the development process. The framework is particularly renowned for its speed and scalability, making it suitable for both small projects and large-scale applications [99].

5.1.2.2 Django Channels

Is an extension of the Django framework that allows you to handle asynchronous protocols such as WebSockets, chat protocols, and IoT protocols, in addition to traditional HTTP requests. It relies on the Asynchronous Server Gateway Interface (ASGI), which allows for both synchronous and asynchronous communication styles. This feature is essential for developing real-time web applications, such as chat applications, which require long-lived connections [100]. The Figure 5.1 illustrates how Django channels work :

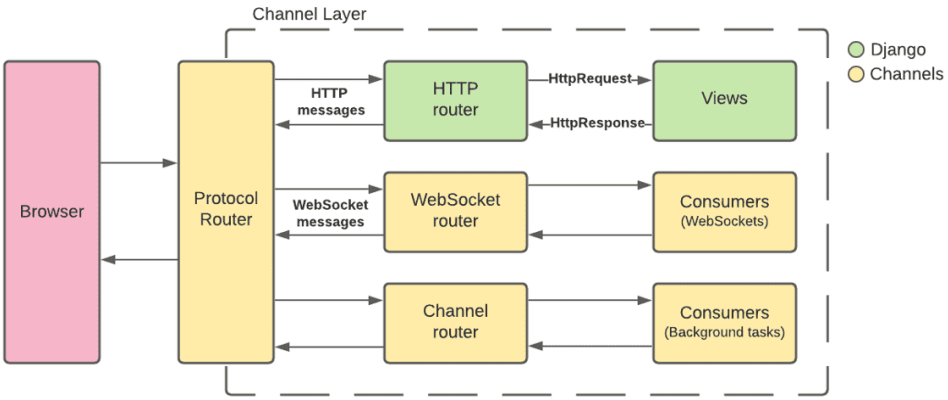

Figure 5.1: Communication flow between the browser and the server

5.1.2.3 Django Framework rest

Is a powerful and flexible toolkit for building web APIs. It offers features such as a searchable API, authentication policies, complex data serialization, and comprehensive documentation. This framework allows developers to create APIs that are easy to use and integrate with various data sources, including ORM and non-ORM data. In addition, it supports customizable views and routing, making it suitable for a wide range of applications [101].

5.1.2.4 SvelteKit

Is a framework designed to quickly develop robust and performant web applications using Svelte. It provides a complete solution for building web applications by incorporating modern best practices and addressing the most common development challenges. Key features include a filesystem-based router, build optimizations, offline support, and configurable rendering options, making it easy to create efficient and user-friendly applications. For those familiar with other frameworks, SvelteKit is analogous to Next.js for React and Nuxt.js for Vue [102].

5.1.3 Databases

5.1.3.1 PostgreSQL

PostgreSQL is a powerful open-source relational database management system (RDBMS) that emphasizes extensibility and conformance to the SQL standard. It supports transactions with atomicity, consistency, isolation, and durability (ACID) properties, automatically updatable views, materialized views, triggers, foreign keys, and stored procedures. PostgreSQL was originally called POSTGRES and was developed at the University of California, Berkeley, under the direction of Professor Michael Stonebraker. The project was renamed PostgreSQL in 1996 to reflect its support for SQL [103].

5.1.3.2 Redis

Redis (REmote DIctionary Server) is an open-source, in-memory NoSQL key and value store used primarily as a fast cache or database. By storing data in memory, it enables extremely fast read and write operations, improving application performance compared to disk-based databases. Redis supports many data structures and offers features such as replication, high availability, and disk persistence, making it ideal for low-latency, high-throughput applications such as real-time analytics and session management [104].

5.1.4 Development environments

5.1.4.1 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, designed specifically for developing Android applications. It is based on JetBrains' IntelliJ IDEA software and was announced on May 16, 2013, at the Google I/O conference. Android Studio provides a comprehensive set of tools and features that help developers design, build, test, and debug Android applications [105].

5.1.4.2 Visual Studio Code

Visual Studio Code is a free, lightweight, and powerful code editor from Microsoft, compatible with Windows, macOS, Linux, and Raspberry Pi OS. It provides an efficient environment for coding, debugging, and running code, and supports many languages (JavaScript, TypeScript, Node.js, Python, etc.) and extensions. Features include IntelliSense, graphical debugging, linting, multicursor editing, and Git integration [106].

5.1.5 Outils

5.1.5.1 Git

Git is a free, open-source, distributed version control system that efficiently manages projects of any size. It allows multiple developers to work simultaneously without conflicts and is popular for its speed, branching, and merging capabilities. With features such as a repository and support for different workflows, Git is a flexible tool for collaborative development [107].

5.2 Main Features

To better illustrate what our application can do, we've selected a few screenshots. These visuals provide a concrete overview of the main features and the ergonomics of the final product.

5.2.1 Child side (Android App)

On the child's side, we provide an Android app that serves as a central point for collecting essential information such as location, WhatsApp activity, and time spent on the app.
Below are some screenshots illustrating our application's registration and login processes. Users can easily register or log in by simply scanning a QR code. Each QR code is unique and cannot be reused to start a new session, which ensures that access remains both simple and secure.

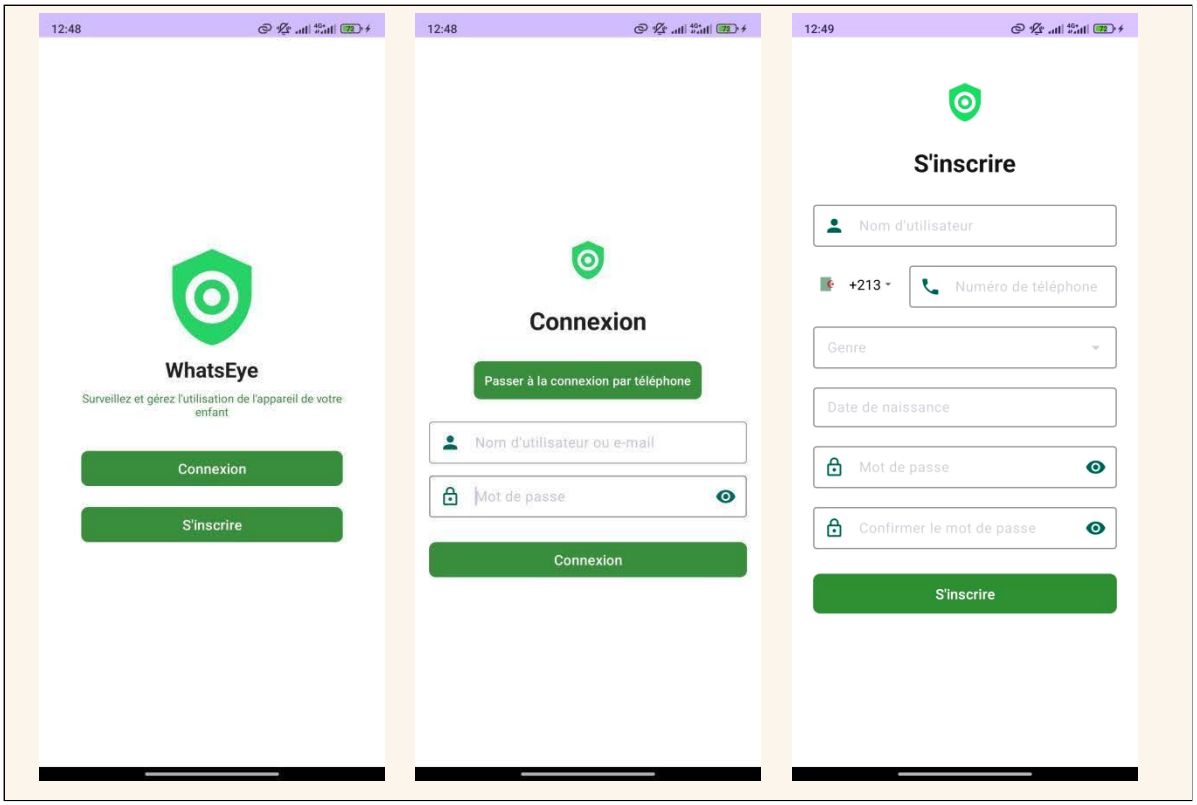

Figure 5.2: Interfaces for registration and connection - Android

5.2.2 Parent side (Web Dashboard)

For parents, we offer an online dashboard that allows them to monitor multiple children simultaneously. Parents can track their children's WhatsApp usage time, set time slots, and access their children's chat rooms. Below, we present some pages of the dashboard to illustrate the usage process.

5.2.2.1 Login and Registration

The screenshots in Figure 5.3 illustrate the login and registration pages. To begin, the parent must first create a family group. Once the family group is created, the parent's account is automatically added to it.

Figure 5.3: Interfaces for login and registration - Web

Conclusion

This section provided a better understanding of the work accomplished by presenting the different interfaces of the project. It thus concludes the development phase of our parental control application, which includes an Android application and a web dashboard.

General conclusion

This work allowed for an in-depth analysis of the Android environment as well as the detailed functioning of WhatsApp, two major components of modern mobile communication. Through the study of the system architecture, memory and storage management mechanisms as well as the security features integrated into Android, we showed that this ecosystem is based on concepts that are both rich and complex, offering unprecedented flexibility for the development of applications adapted to specific uses. Attention was then focused on WhatsApp, a globally essential instant messaging application, in order to detail its operation, structure as well as the end-to-end encryption mechanisms guaranteeing the confidentiality of exchanges. This study highlighted the specific issues related to the intensive use of WhatsApp by the youngest, as well as the limits of existing parental control systems. It is precisely in this context that the proposed solution was developed and detailed, aiming to design a targeted parental monitoring tool, respectful of privacy, adapted to the Android environment, and capable of effectively responding to concrete needs such as the protection of children vis-has-of dangerous content, at-risk groups, or malicious contacts. The design work and implementation of the solution illustrate both the technical feasibility of the project and the value it can bring to parents concerned about their children's digital safety.

This work is certainly not perfect and could be improved in many ways. Other features could thus complete the envisaged solution, including the addition of real-time alerts when suspicious content is detected, the implementation of natural language processing (NLP) algorithms to proactively identify risky conversations, as well as the strengthening of the user experience of the parent dashboard through more detailed and intuitive visualizations. Furthermore, the integration of machine learning models could make it possible to refine the precision of filtering as well as the personalization of monitoring rules according to the child's age or the context of use. Ultimately, this work is part of a broader reflection on the protection of the youngest in a constantly evolving mobile environment. As technologies advance, the implementation of more adapted, more precise and more privacy-respecting control mechanisms will remain a major challenge. This thesis thus provides a solid

basis for future work aimed at designing more efficient, more intelligent parental control mechanisms that are better adapted to the diversity of digital uses, thus contributing to building a safer digital environment for all.

Appendix A

Android Version History

The Figure A.1 presents the main versions of Android since the first release in 2008.

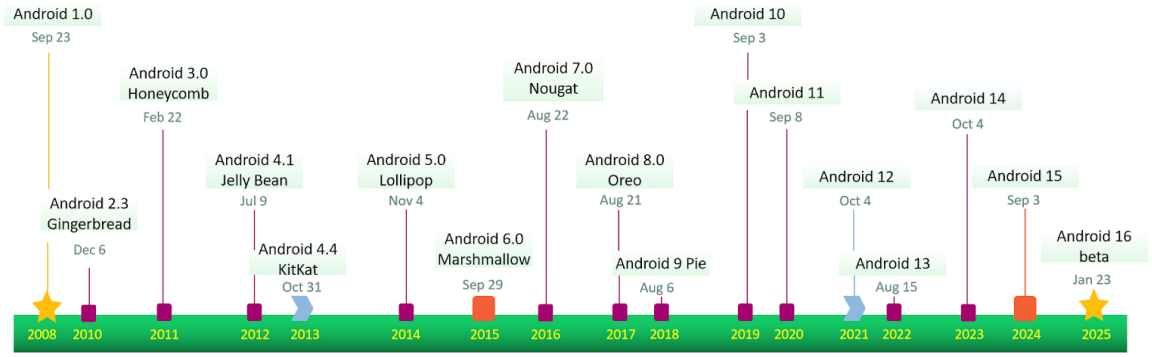

Figure A.1: Versions d'Android 2008-2025

Since its inception in 2008, the operating system has undergone significant transformations, with numerous versions released over the years. From version 1.0 to the latest beta 16 in 2024, the operating system has evolved considerably, incorporating new features, improvements, and enhancements.

On August 22, 2019, it was announced that Android Q would be referred to simply as "Android 10" [108], marking a change in Google's naming convention. This change marks the end of the practice of using alphabetical codenames based on confectionery products for major releases. This new approach aims to simplify and clarify the nomenclature, making it easier for users to identify releases without resorting to dessert-themed codenames.

In 2017, Google made a significant change to its Play Store policies, requiring apps to target a recent version of Android. This initiative aims to ensure that all apps take full advantage of the latest features, improvements, and security enhancements offered by newer versions of Android [109].

Due to this significant change, we will only cover major features and updates from recent Android 13 to Android 16 beta versions, available in the table A.1.

Version	Features
Android 13	• Material You Personalization: Enhanced customization options allow non-Google apps to adapt to the wallpaper theme and colors, creating a more consistent look on your home screen. However, developers must opt in to this feature [110]. • Android 13 lets you further customize Sleep Mode with wallpaper dimming and a dark theme. These screen options help your eyes adjust to the dark when you go to bed, and help you get back to sleep if you wake up in the night to check your phone [110]. • Gone are the days when you had to share your entire media library with your apps. In Android 13, you can choose which photos and videos apps should have access to [110]. • Prevent unwanted access to your clipboard. If you copy sensitive data like your email address, phone number, or login credentials to your device, Android will automatically clear your clipboard history after a certain period of time [110]. • Android 13 lets you control your notifications and ensure you only receive the alerts you've expressed a need for. Apps will now need your explicit permission to send notifications, instead of being allowed by default [110].
Android 14	• Android 14's updated customization switcher makes it easier to switch between wallpapers and update what you want to see at a glance. You can now set custom lock screen shortcuts, like the QR reader or the Google Home app, for quick, one-touch access to commonly used commands, right from the Android lock screen [111]. • Android 14 offers more intuitive ways to connect and interact with your hearing aids. These include a dedicated hearing aid setup flow in Accessibility settings, an easy way to route audio to different outputs, and a shortcut for quick access to hearing aid controls. Likewise, notification sounds, ringtones, or alerts [111].

Android 14	•You also get better visibility into how your data is used by the apps that request it. With new data sharing updates in Android 14, when you're asked to allow apps to access information like your location, you'll be notified if an app is sharing location data with third parties, and you can make an informed decision about whether to allow access to that data [111]. • Digital protection in Android 14 also extends to your most sensitive information, like your device's personal identification numbers (PINs). Android 14 strengthens PIN security by encouraging you to choose a six-digit PIN. After entering the correct PIN of six or more digits, your device will unlock automatically, without requiring you to press the Enter key [111].
Android 15	• Android 15 expands support for satellite connectivity, allowing preloaded SMS and RCS messaging apps as well as carrier messaging apps to use satellite connectivity to send and receive messages [112]. • On foldable devices and tablets, you can easily pin and unpin your taskbar to the screen. This lets you customize your view and keep your favorite apps, like Google Photos or Gmail, close at hand for faster access and improved productivity [112]. • The new theft detection lock uses artificial intelligence to keep your data safe. It automatically locks if someone steals on foot, by bike, or by car. You can also use Remote Lock to quickly lock your device from any device by entering your phone number in a simple security check. These features are now available for most Android 10+ devices [112]. • Private Space in Android 15 acts like a digital vault on your phone. You can create a separate space to organize sensitive apps, like dating, banking, or social apps. When Private Space is locked, apps remain virtually invisible to others and are hidden from your apps list, recent apps view, notifications, and settings. To access it, an extra layer of authentication helps secure apps and keep them away from prying eyes. For added privacy, you can also choose to hide the existence of Private Space on your phone [112].

Android 16 beta	• Live Updates These are dynamic notifications that help users track and quickly access important ongoing activities, such as ride-sharing and food delivery, in real time [113]. • Adaptive apps This feature allows apps to adapt their functionality based on the user's context and preferences, providing a more personalized experience [113]. • Android 16 also supports the APV (Advanced Professional Video) codec, which offers high-bitrate intra-frame-only coding, perceptually lossless video quality, and high bitrates, even when shooting in 4K and 8K [113]. • Privacy Sandbox enables improved data encryption and handling of sensitive information, providing increased security for users [114]. • With Android 16, Google is adding support for robust security features in Wi-Fi location on devices compatible with the 802.11az standard of Wi-Fi 6. With this, apps can now combine the protocol's higher accuracy, scalability, and dynamic scheduling with security enhancements, including AES-256 encryption and protection against MITM attacks. This makes it more secure for use in proximity use cases, such as unlocking a laptop or a vehicle door [114].

Table A.1: New features in the latest versions of Android

Appendix B

Solution Interfaces

B.1 Child side (Android App)

B.1.1 Required Permissions

Below you will find screenshots illustrating the permissions requested by the application. These permissions are essential to ensure the proper functioning of the application (see Figures B.1 and B.2).

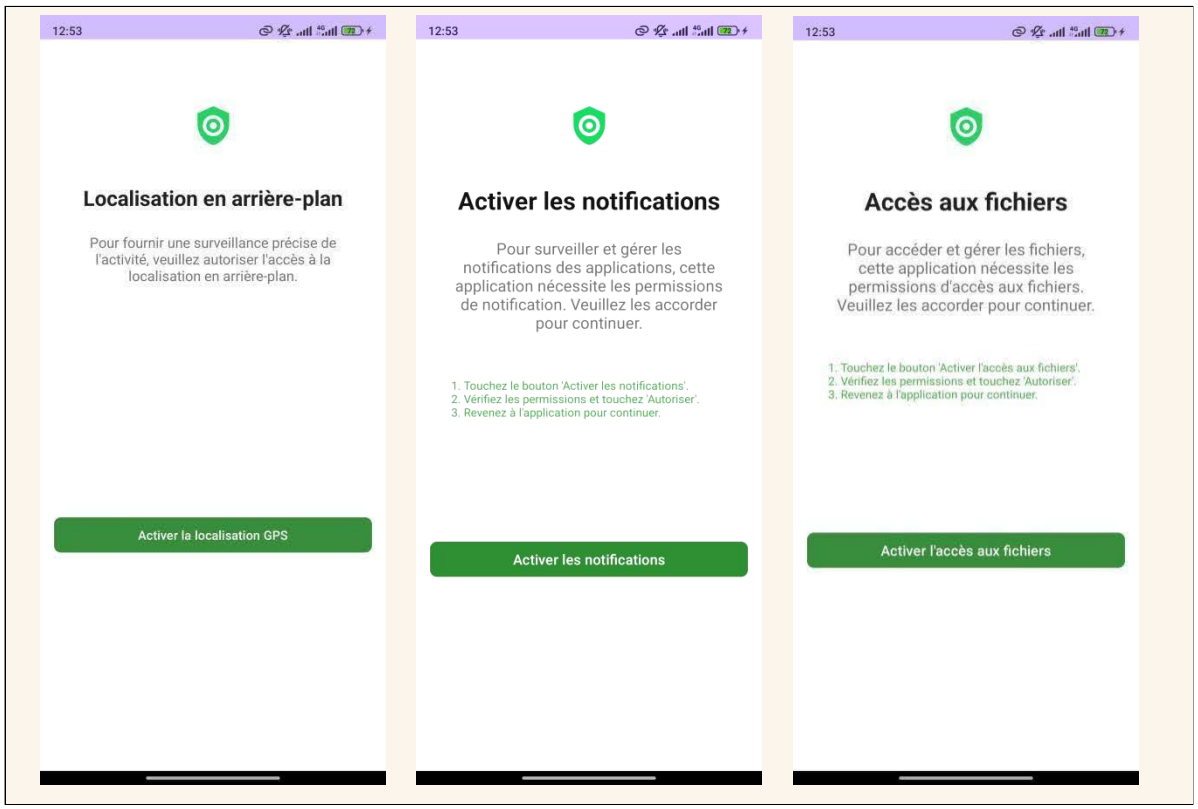

Figure B.1: Required Permissions I

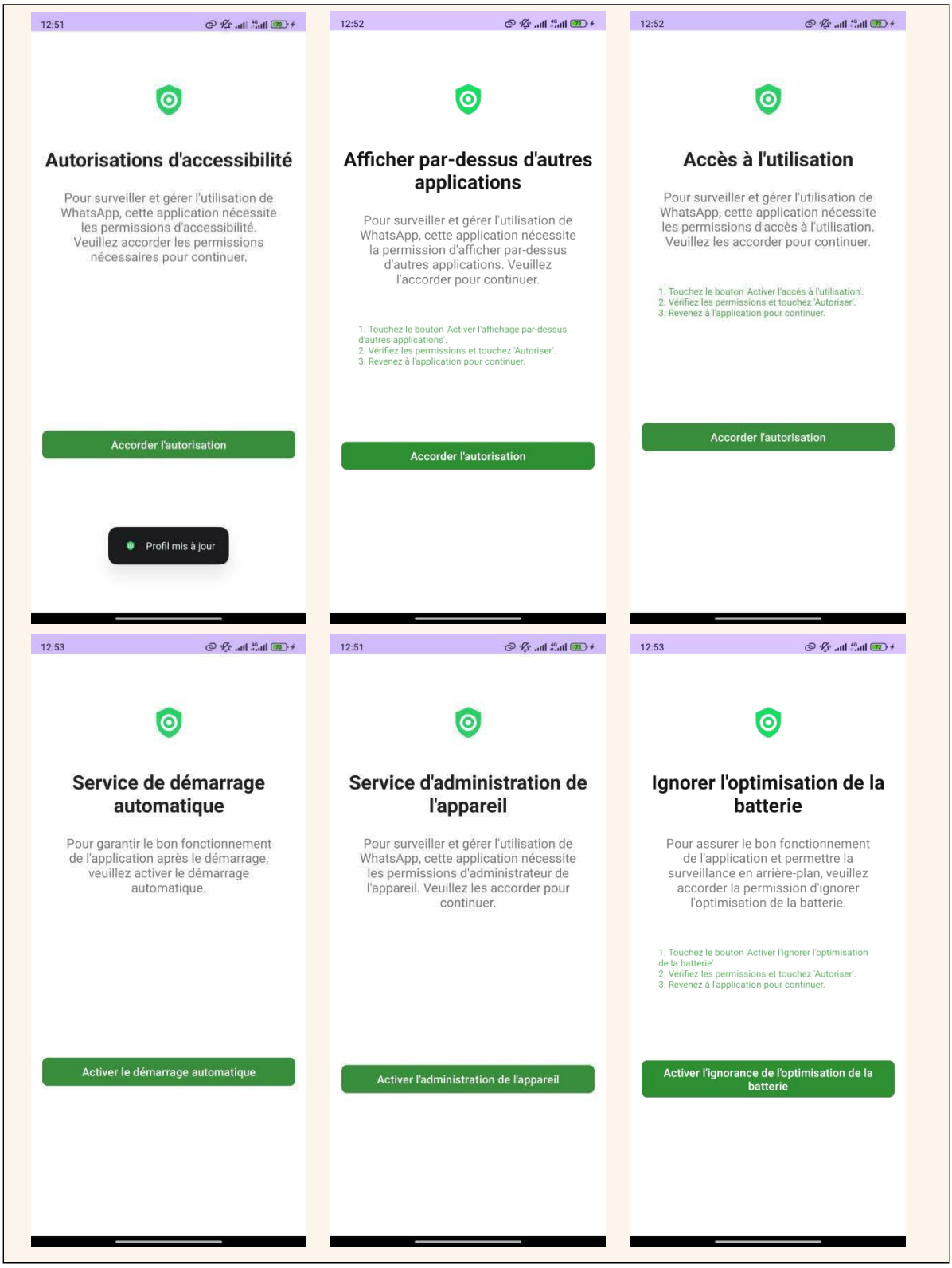

Figure B.2: Required Permissions II

B.1.2 Application Interfaces

The screenshots in Figure B.3 illustrate the PIN code request page, the default page accessible to the child, and the page allowing the parent to update the settings from the child's phone. The latter is protected by a PIN code for security reasons.

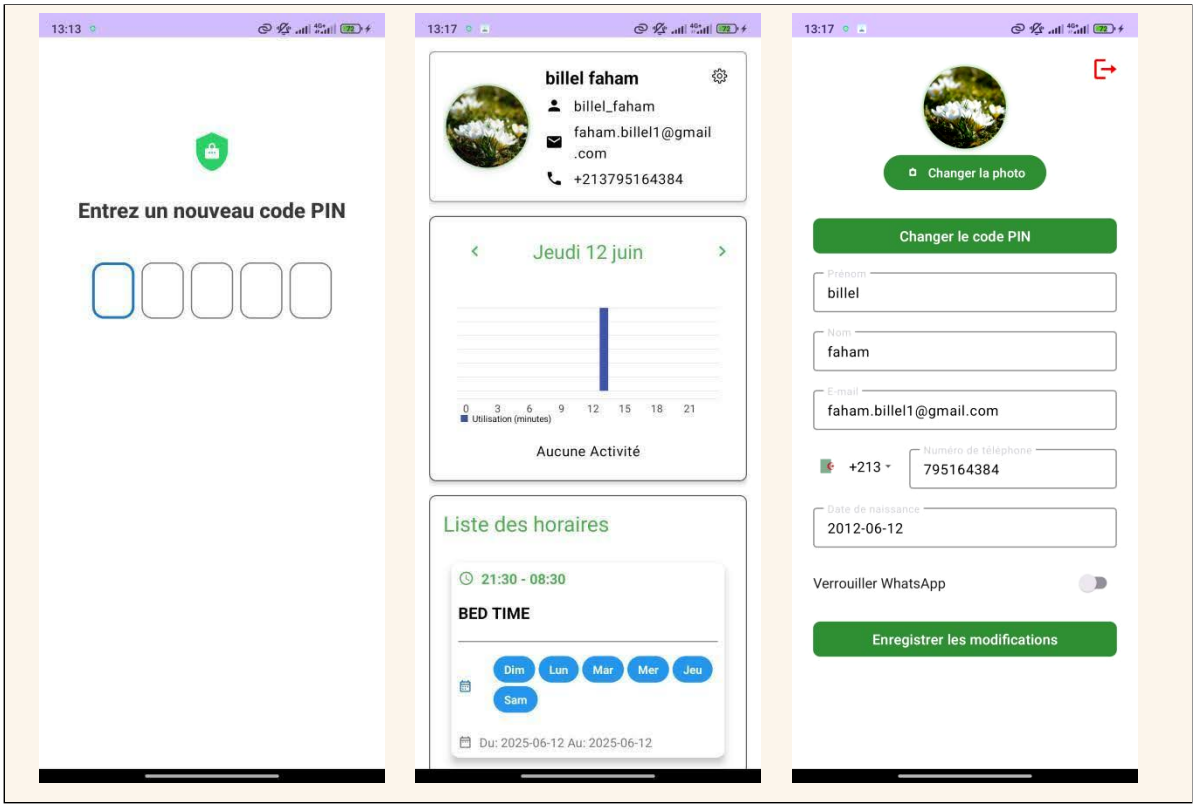

Figure B.3: Android Intefraces

B.2 Parent side (Web Dashboard)

B.2.1 Parent Dashboard Homepage

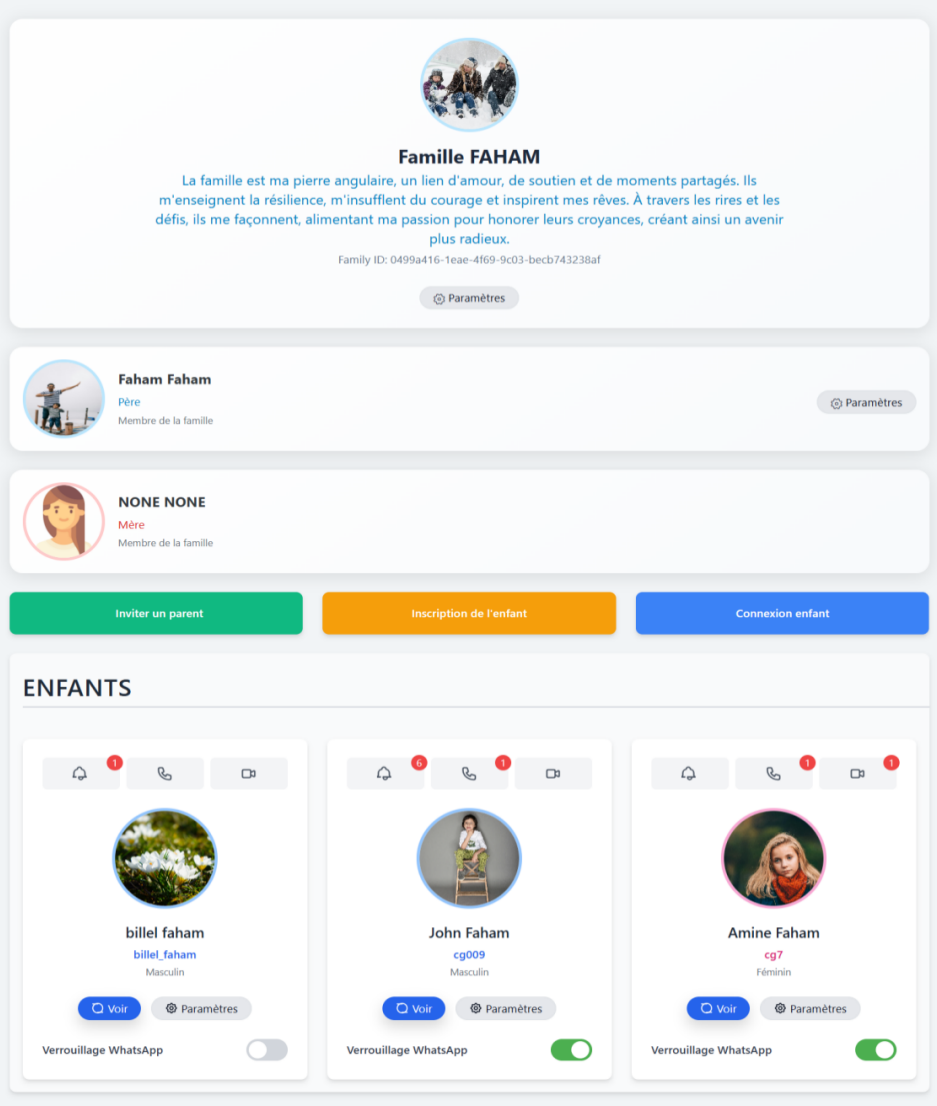

Figure B.4: Dashboard - Home

B.2.2 Dashboard Homepage after selecting a Child profile

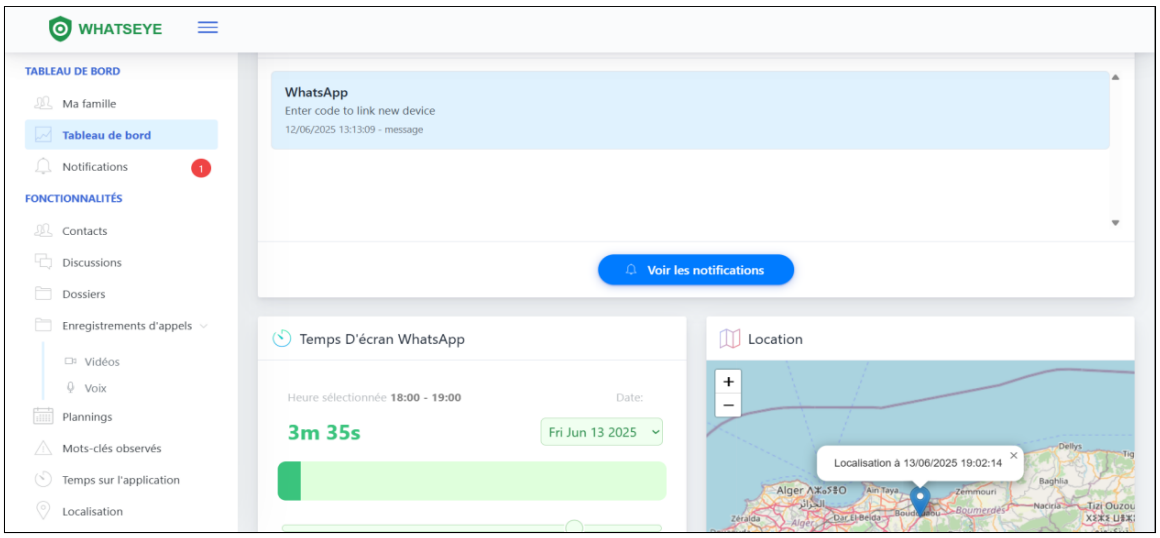

Figure B.5: Dashboard - Selected Child

B.2.3 Notifications Page

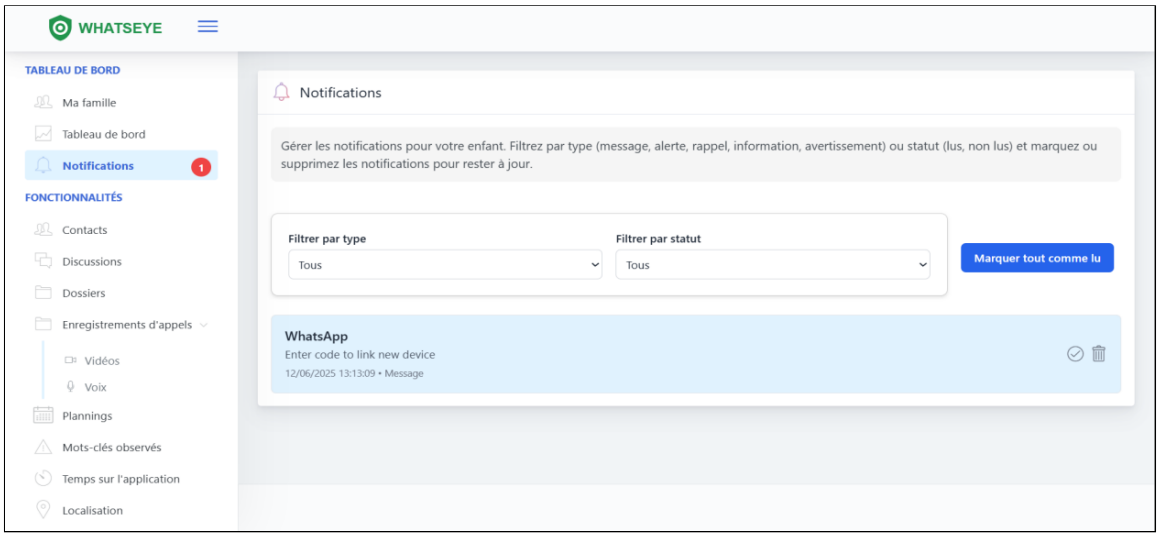

Figure B.6: Dashboard - Notifications

B.2.4 WhatsApp Contact List Page

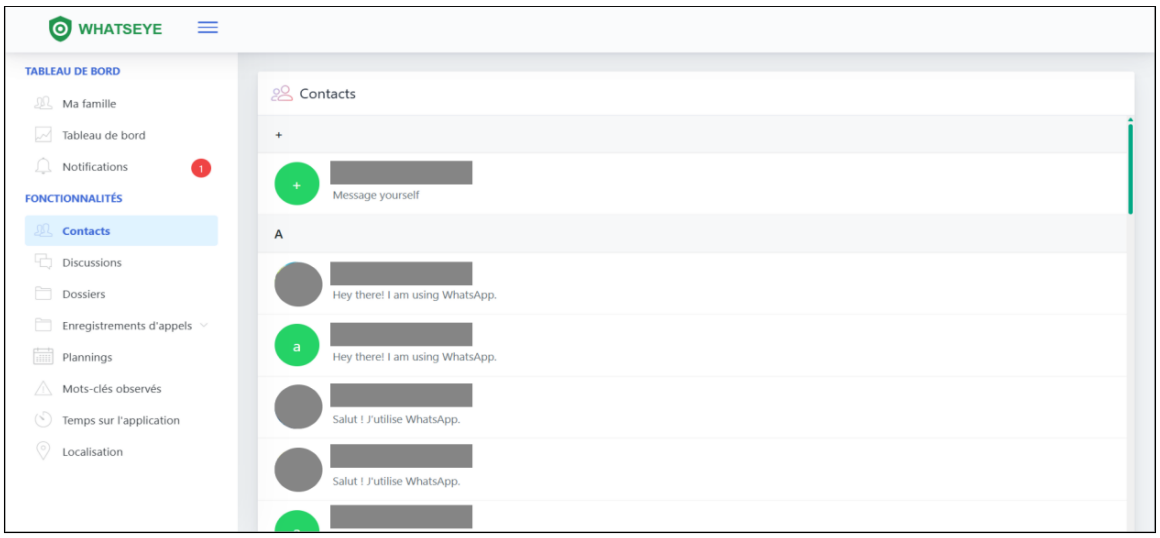

Figure B.7: Dashboard - Contacts

B.2.5 WhatsApp Discussion and Chat Pages

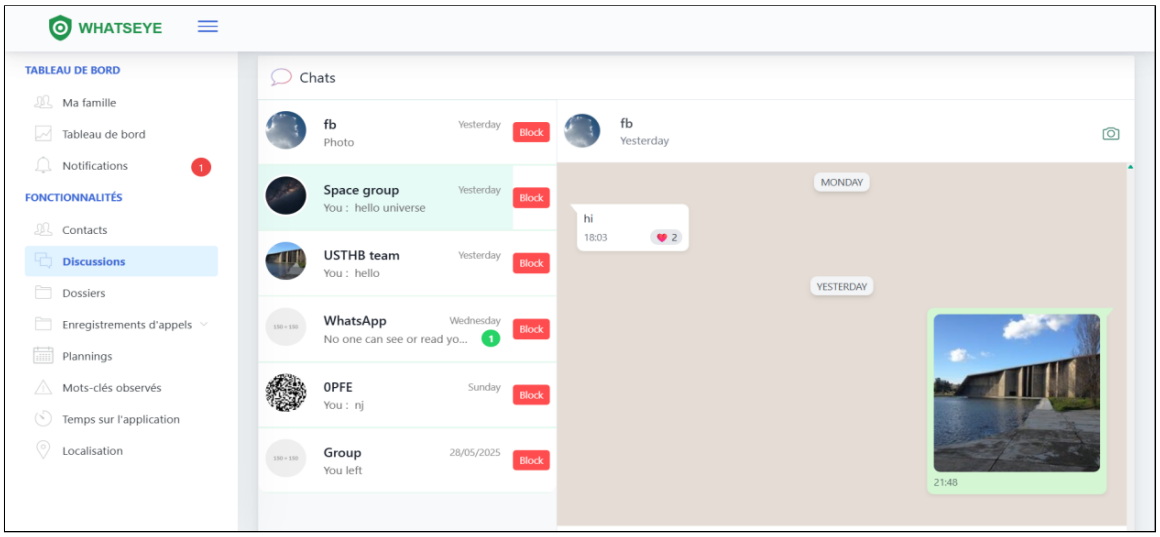

Figure B.8: Dashboard - Discussions

B.2.6 WhatsApp Media Folder Access Page

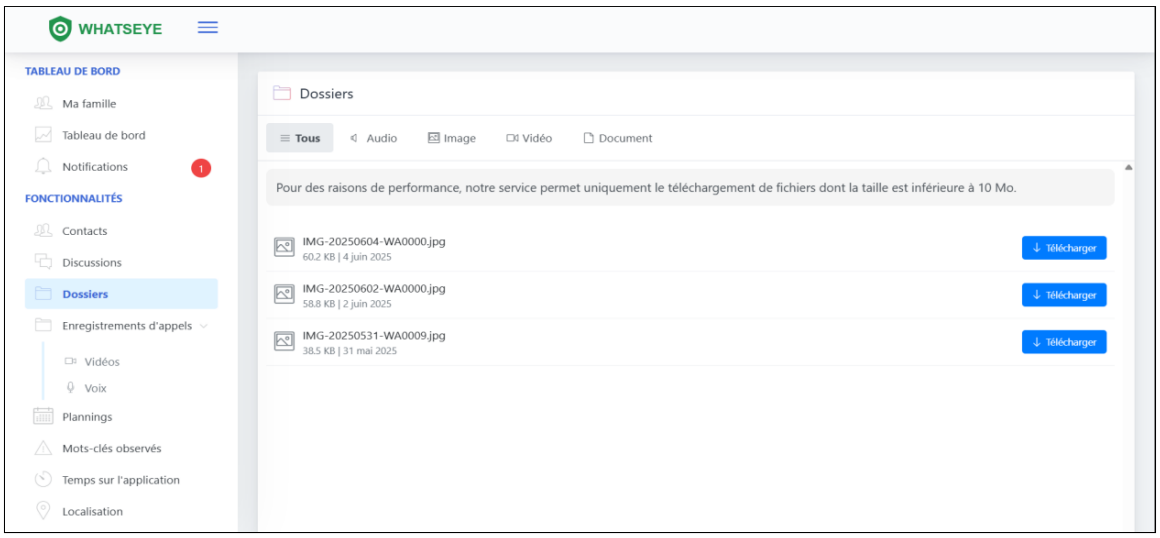

Figure B.9: Dashboard - Folders

B.2.7 WhatsApp Usage Schedule Configuration Page

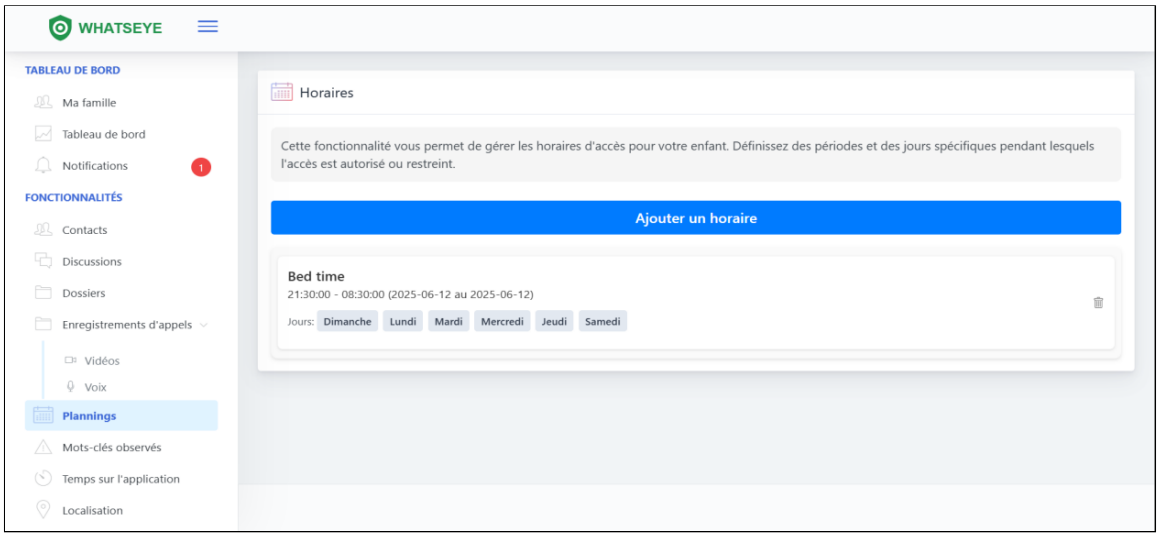

Figure B.10: Dashboard - Plannings

B.2.8 Voice and Video Call Recording Page

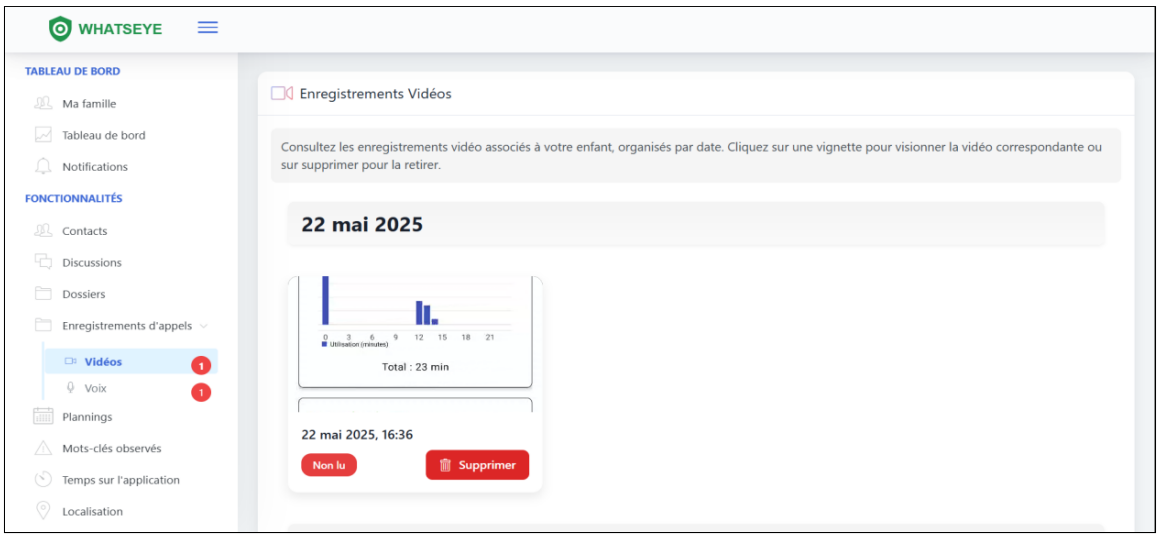

Figure B.11: Dashboard - Vidoe recording

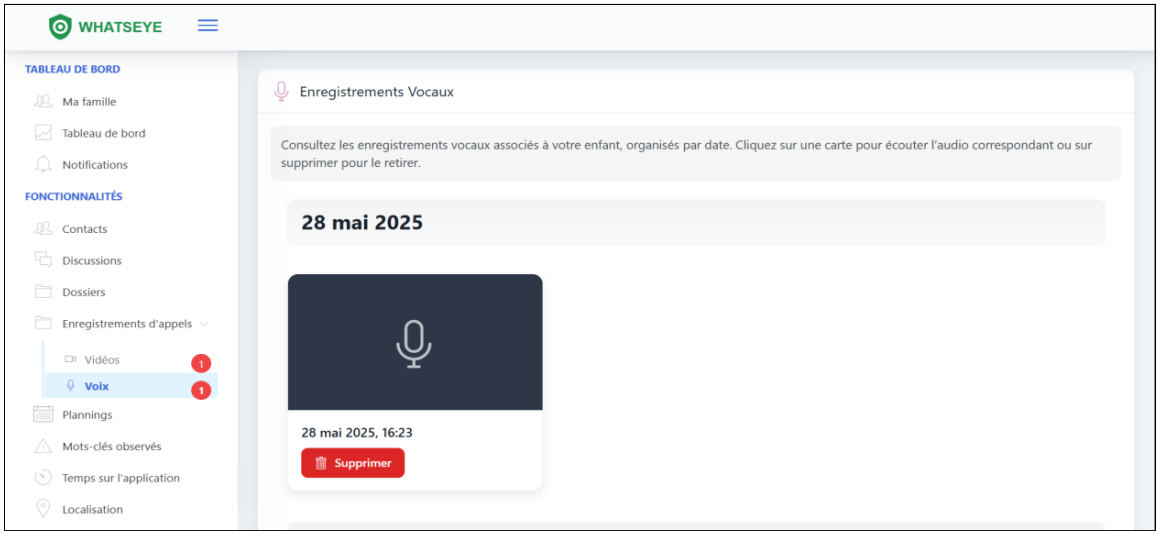

Figure B.12: Dashboard - Voice recording

B.2.9 Inappropriate Language Alert Page

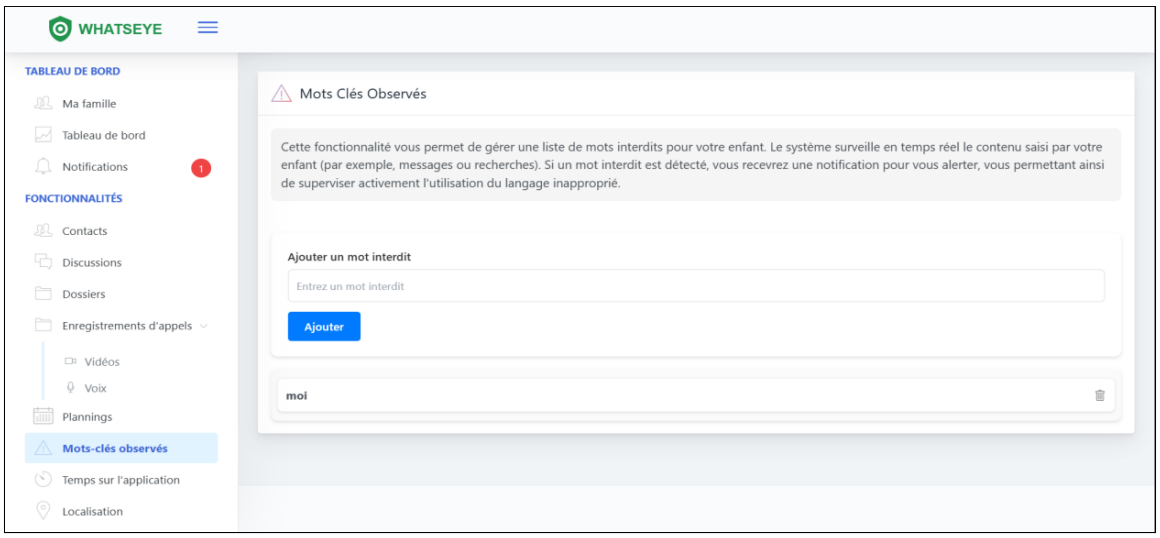

Figure B.13: Dashboard - Keywords

B.2.10 WhatsApp Usage Statistics Page

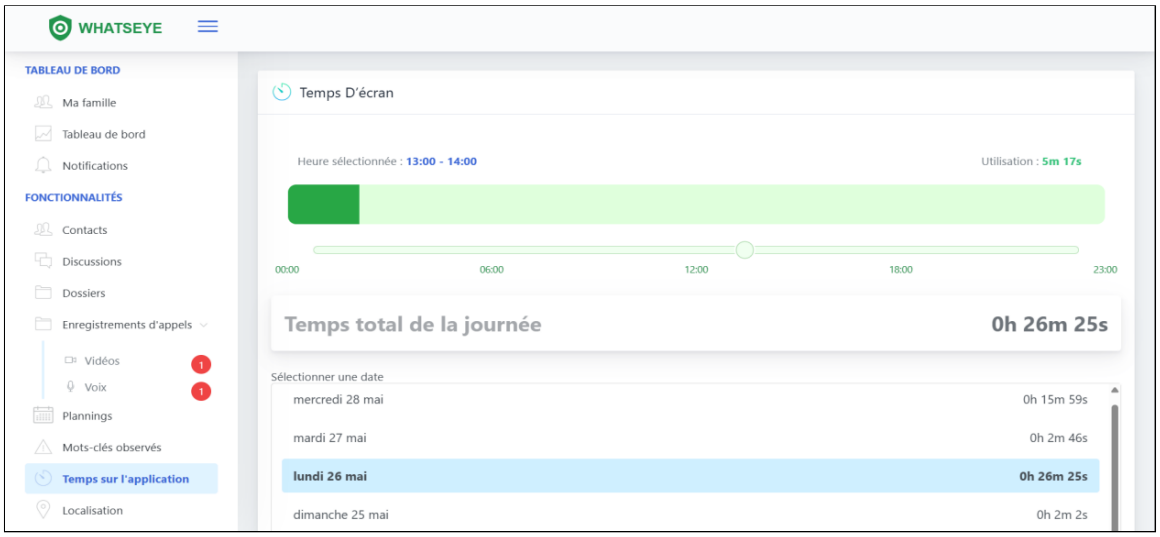

Figure B.14: Dashboard - Screen Time

B.2.11 Location Page

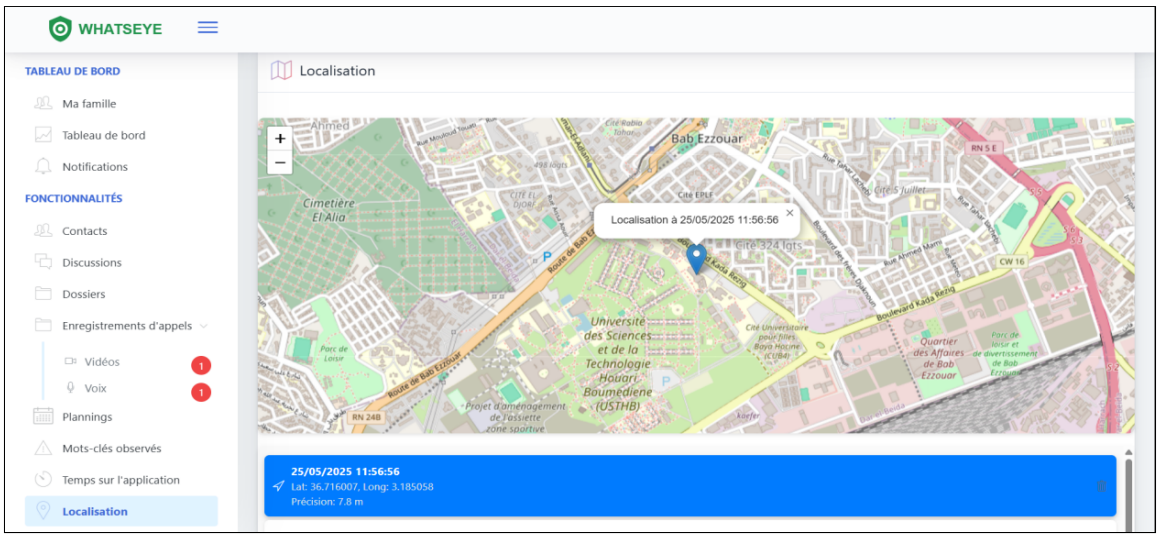

Figure B.15: Dashboard - Location

Bibliography

[1] « Mobile Operating System Market Share Worldwide », StatCounter Global Stats. Accessed: February 14, 2025. [Online]. Available at: https://gs.statcounter.com/os-market-share/mobile/worldwide

[2] B. Logerot, « De 2003 à 2023, retour sur l'histoire d'Android et comment l'OS s'est imposé dans le monde », L'Éclaireur Fnac. Accessed: February 14, 2025. [Online]. Available at: https://leclaireur.fnac.com/article/336140-de-2003-a-2023-retour-sur-lhistoire-dandroid-et-comment-los-sest-impose-d

[3] « Android history: The evolution of the biggest mobile OS in the world ». Accessed: February 14, 2025. [Online]. Available at: https://www.androidauthority.com/history-android-os-name-789433/

[4] « Platform architecture », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/guide/platform

[5] « Kernel overview », Android Open Source Project. Accessed: February 14, 2025. [Online]. Available at: https://source.android.com/docs/core/architecture/kernel

[6] « Hardware abstraction layer (HAL) overview », Android Open Source Project. Accessed: February 14, 2025. [Online]. Available at: https://source.android.com/docs/core/architecture/hal

[7] « Android runtime and Dalvik », Android Open Source Project. Accessed: February 14, 2025. [Online]. Available at: https://source.android.com/docs/core/runtime

[8] KmDev, « Exploring the Differences Between Dalvik and ART Runtimes in Android », Medium. Accessed: February 14, 2025. [Online]. Available at: https://medium.com/@mkcode0323/exploring-the-differences-between-dalvik-and-art-runtimes-in-android-4d1bacfb3dc

[9] C. Ferry, « ART's Garbage Collector: Strategies for best performance », Medium. Accessed: February 14, 2025. [Online]. Available at: https://sonique6784.medium.com/arts-garbage-collector-strategies-for-best-performance-1f5c79208851

[10] « Get started with Jetpack Compose », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/develop/ui/compose/documentation

[11] « App resources overview | App architecture », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/guide/topics/resources/providing-resources

[12] « Notifications overview | Views », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/develop/ui/views/notifications

[13] « Application fundamentals | App architecture », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/guide/components/fundamentals

[14] « Content providers | App data and files », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/guide/topics/providers/content-providers

[15] « Memory allocation among processes | App quality », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/topic/performance/memory-management

[16] « Overview of memory management | App quality », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/topic/performance/memory-overview

[17] « Memory management best practices | Places SDK for Android », Google for Developers. Accessed: February 14, 2025. [Online]. Available at: https://developers.google.com/maps/documentation/places/android-sdk/memory-best-practices

[18] S. G, « Memory Management in Android ». Accessed: February 14, 2025. [Online]. Available at: https://www.robosoftin.com/blog/memory-management-in-android

[19] « Data and file storage overview | App data and files », Android Developers. Accessed: February 14, 2025. [Online]. Available at: https://developer.android.com/training/data-storage

[20] Opensource.com, « What is open source? | Opensource.com ». Accessed: February 17, 2025. [Online]. Available at: https://opensource.com/resources/what-open-source

[21] « Licences open source: types et comparaison », Snyk. Accessed: February 14, 2025. [Online]. Available at: https://snyk.io/fr/articles/open-source-licenses/

[22] « All About Permissive Licenses - FOSSA », Dependency Heaven. Accessed: February 14, 2025. [Online]. Available at: https://fossa.com/blog/all-about-permissive-licenses/

[23] « All About Copyleft Licenses - FOSSA », Dependency Heaven. Accessed: February 14, 2025. [Online]. Available at: https://fossa.com/blog/all-about-copyleft-licenses/

[24] « Contributor license agreements and headers », Android Open Source Project. Accessed: February 14, 2025. [Online]. Available at: https://source.android.com/docs/setup/contribute/licenses

[25] « Android, AOSP, and what Open-Source Licensing means for you or your company », electrikjesus blog. Accessed: February 14, 2025. [Online]. Available at: https://electrikjesus.github.io/blog/Android-AOSP-and-what-Open-Source-Licensing-means-for-you-or-your-company/

[26] « Android security features », Android Open Source Project. Accessed: February 14, 2025. [Online]. Available at: https://source.android.com/docs/security/features

[27] « App signing | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/apksigning

[28] « Authentication | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/authentication

[29] « Biometrics | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/biometric

[30] « Encryption », Android Open Source Project. Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/encryption

[31] « Hardware-backed Keystore | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/keystore

[32] « Security-Enhanced Linux in Android | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/selinux

[33] « Trusty TEE | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/trusty

[34] « Verified Boot | Android Open Source Project ». Accessed: February 19, 2025. [Online]. Available at: https://source.android.com/docs/security/features/verifiedboot

[35] M. McDunnigan, « Security Risks of Androids », Chron - Small Business. Accessed: February 14, 2025. [Online]. Available at: https://smallbusiness.chron.com/security-risks-androids-68511.html

[36] « Android Security Issues ». Accessed: February 14, 2025. [Online]. Available at: https://www.cse.wustl.edu/~jain/cse571-14/ftp/android_security/index.html

[37] « The Evolution of Instant Messaging: A Comprehensive Guide - nativeMsg ». Accessed: April 16, 2025. [Online]. Available at: https://nativemsg.com/resources/text-marketing/the-evolution-of-instant-messaging-a-comprehensive-guide/

[38] « The Internet Relay Chat (IRC) turned 30 – and it probably changed our lives ». Accessed: April 16, 2025. [Online]. Available at: https://www.zmescience.com/science/internet-relay-chat-irc-08112018/

[39] « A Brief History of Chat Apps · Guide to Chat Apps ». Accessed: April 16, 2025. [Online]. Available at: https://towcenter.gitbooks.io/guide-to-chat-apps/content/introductionthe_dawn_of/a_brief_history.html

[40] « Les 12 meilleurs outils de messagerie instantanée en 2024 - Freshworks ». Accessed: April 16, 2025. [Online]. Available at: https://www.freshworks.com/fr/messaging-channels/software/

[41] G. E. Goodwin, « What is WhatsApp? A guide to navigating the free Meta-owned communication platform », Business Insider. Accessed: April 16, 2025. [Online]. Available at: https://www.businessinsider.com/guides/tech/what-is-whatsapp-guide

[42] C. Liu, « Introduction to WeChat », MavSocial. Accessed: April 16, 2025. [Online]. Available at: https://mavsocial.com/introduction-to-wechat/

[43] « What is Facebook Messenger? | dummies ». Accessed: April 16, 2025. [Online]. Available at: https://www.dummies.com/article/technology/social-media/facebook/what-is-facebook-messenger-221164/

[44] « What is Telegram Messenger and why should I use it? - Android Authority ». Accessed: April 16, 2025. [Online]. Available at: https://www.androidauthority.com/what-is-telegram-messenger-979357/

[45] « WhatsApp Statistics, Users, Demographics as of 2024 », Whats the Big Data. Accessed: April 16, 2025. [Online]. Available at: https://whatsthebigdata.com/whatsapp-statistics/

[46] M. Dey, « WhatsApp Statistics By Region, User Growth, Active Users And Time Spent », Electro IQ. Accessed: April 16, 2025. [Online]. Available at: https://electroiq.com/stats/whatsapp-statistics/

[47] « WhatsApp | History & Facts | Britannica ». Accessed: April 16, 2025. [Online]. Available at: https://www.britannica.com/topic/WhatsApp

[48] « Confidentialité WhatsApp | Messages sécurisés et privés », WhatsApp.com. Accessed: April 16, 2025. [Online]. Available at: https://www.whatsapp.com/privacy

[49] « WhatsApp », WhatsApp.com. Accessed: April 16, 2025. [Online]. Available at: https://www.whatsapp.com/community

[50] « Exprimez-vous | Fonctionnalités WhatsApp », WhatsApp.com. Accessed: April 16, 2025. [Online]. Available at: https://www.whatsapp.com/expressyourself

[51] « Gardez le contact | Des messages, des appels et plus encore sur WhatsApp », WhatsApp.com. Accessed: April 16, 2025. [Online]. Available at: https://www.whatsapp.com/stayconnected

[52] « Most popular messaging apps 2025 », Statista. Accessed: April 16, 2025. [Online]. Available at: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/

[53] « Understanding WhatsApp's Architecture & System Design ». Accessed: April 16, 2025. [Online]. Available at: https://www.cometchat.com/blog/whatsapps-architecture-and-system-design

[54] derek, « Scaling with DevOps Best Practices – The WhatsApp Way» PipeOps Blog ». Accessed: April 16, 2025. [Online]. Available at: https://blog.pipeops.io/how-whatsapp-scaled-its-platform-using-devops-best-practices/

[55] A. P. Singh, « Design a Chat Application like WhatsApp - System Design Interview ». Accessed: April 16, 2025. [Online]. Available at: https://blog.algomaster.io/p/design-a-chat-application-like-whatsapp

[56] R. Verma, « How WhatsApp works », Medium. Accessed: April 16, 2025. [Online]. Available at: https://medium.com/@rajendra_51543/how-whatsapp-works-197bfc6d6b95

[57] « WhatsApp Tech Stack Explored — The Tech Behind Series », Intuji. Accessed: April 16, 2025. [Online]. Available at: https://intuji.com/whatsapp-tech-stack-explored/

[58] « Making messaging interoperability with third parties safe for users in Europe », Engineering at Meta. Accessed: April 16, 2025. [Online]. Available at: https://engineering.fb.com/2024/03/06/security/whatsapp-messenger-messaging-interoperability-eu/

[59] M. Schirrmacher, « Analyzing WhatsApp Calls », Medium. Accessed: April 16, 2025. [Online]. Available at: https://medium.com/@schirrmacher/analyzing-whatsapp-calls-176a9e776213

[60] « Digital Forensics: Artifact Profile - WhatsApp Messenger - Magnet Forensics ». Accessed: April 16, 2025. [Online]. Available at: https://www.magnetforensics.com/blog/artifact-profile-whatsapp-messenger/

[61] « Where is WhatsApp Backup Stored on Android and iPhone - BOBcloud ». Accessed: April 16, 2025. [Online]. Available at: https://www.bobcloud.net/where-is-whatsapp-backup-stored/

[62] « How WhatsApp enables multi-device capability », Engineering at Meta. Accessed: April 16, 2025. [Online]. Available at: https://engineering.fb.com/2021/07/14/security/whatsapp-multi-device/

[63] « WhatsApp Data Security: End-to-End Encryption and Backups ». Accessed: April 16, 2025. [Online]. Available at: https://www.wati.io/blog/understanding-whatsapp-data-security-understand-end-to-end-encryption-and-backup

[64] Ashwin, « WhatsApp Privacy and Security: Tips to Keep Your Chats Safe », Wati.io. Accessed: April 16, 2025. [Online]. Available at: https://www.wati.io/blog/whatsapp-privacy-security/

[65] « Politique de confidentialité », WhatsApp.com. Accessed: April 16, 2025. [Online]. Available at: https://www.whatsapp.com/legal/privacy-policy

[66] « WhatsApp Revolutionizing Global Communication - Fansoria ». Accessed: April 16, 2025. [Online]. Available at: https://www.fansoria.com/whatsapp-revolutionizing-global-communication/

[67] « The Rise of WhatsApp: Stats and Story Behind Its Global Success - TimelinesAI ». Accessed: April 16, 2025. [Online]. Available at: https://timelines.ai/whatsapp-stats/

[68] K. Allil, S. Faisal, et A. Zia, « Why Millennials Continue to Use WhatsApp? A Focus on Culture and Computer–Human Dialogue », Human Behavior and Emerging Technologies, vol. 2024, no 1, p. 8439194, 2024, doi: 10.1155/2024/8439194.

[69] « Is WhatsApp Secure - Comparitech ». Accessed: April 16, 2025. [Online]. Available at: https://www.comparitech.com/blog/vpn-privacy/is-whatsapp-secure/

[70] « Serious doubts raised over WhatsApp's misinformation strategy – new report », Loughborough University. Accessed: April 16, 2025. [Online]. Available at: https://www.lboro.ac.uk/news-events/news/2024/february/doubts-over-whatsapp-misinformation-policy/

[71] K. Labs, « WhatsApp Hack: Threats, Real Cases, and Security Strategies - Keepnet », Keepnet Labs. Accessed: April 16, 2025. [Online]. Available at: https://keepnetlabs.com/blog/whats-app-hack-threats-and-protection-strategies

[72] A. J. Mohammed Abubakar, « 10 Best WhatsApp Alternatives You Must Try in 2025 », Beebom. Accessed: April 16, 2025. [Online]. Available at: https://beebom.com/whatsapp-alternative-apps/

[73] « 5 Reasons Why Parental Control Is Important In The Digital Age - Halt.org ». Accessed: April 16, 2025. [Online]. Available at: https://www.halt.org/5-reasons-why-parental-control-is-important-in-the-digital-age

[74] « Is WhatsApp Safe for Kids? How Predators Find Victims on the App ». Accessed: April 16, 2025. [Online]. Available at: https://gabb.com/blog/is-whatsapp-safe-for-kids/

[75] C. POPOV, « What Parents Need to Know: Is WhatsApp Safe for Children? », Hot for Security. Accessed: April 16, 2025. [Online]. Available at: https://www.bitdefender.com/en-us/blog/hotforsecurity/parents-need-know-whatsapp

[76] « Why you need parental control software – and 5 features to look for ». Accessed: April 16, 2025. [Online]. Available at: https://www.welivesecurity.com/2023/05/12/why-need-parental-control-software-5-features-look-for/

[77] « Best Parental Control App for WhatsApp 2025: Top Guide! - Impulsec ». Accessed: April 16, 2025. [Online]. Available at: https://impulsec.com/parental-control-software/best-parental-control-app-for-whatsapp/

[78] « 7 Best Parental Controls for WhatsApp in 2025 ». Accessed: April 16, 2025. [Online]. Available at: https://www.safetydetectives.com/blog/best-parental-controls-for-whatsapp/

[79] M. Eriksson, « 10 Best Parental Control Apps for iPhone, iOS & Android 2025 », Private Proxy Guide. Accessed: April 16, 2025. [Online]. Available at: https://www.privateproxyguide.com/best-parental-control-apps-for-iphone/

[80] « The Future of Parenting: How WhatsApp Tracker App Enhance Child Safety ». Accessed: April 16, 2025. [Online]. Available at: https://www.outrightcrm.com/blog/future-of-parenting-whatsapp-tracker-app/

[81] « A “stalkerware” app leaked phone data from thousands of victims ». Accessed: April 16, 2025. [Online]. Available at: https://techcrunch.com/2020/02/20/kidsguard-spyware-app-phones/

[82] « Behavior changes: Apps targeting Android 13 or higher », Android Developers. Accessed: June 17, 2025. [Online]. Available at: https://developer.android.com/about/versions/13/behavior-changes-13

[83] amolbh, « What is the location of WhatsApp’s encryption key file? », r/DataHoarder. Accessed: June 17, 2025. [Online]. Available at: https://www.reddit.com/r/DataHoarder/comments/11jpf49/what_is_the_location_of_whatsapps_encryption_key/

[84] « Restricted Settings in Android 13 and 14 ». Accessed: June 17, 2025. [Online]. Available at: https://usa.kaspersky.com/blog/android-restricted-settings/29485/

[85] « tesi.pdf ». Accessed: June 17, 2025. [Online]. Available at: https://webthesis.biblio.polito.it/33137/1/tesi.pdf

[86] « Analyse forensique de WhatsApp sur Android. Partie I : Acquisition ». Accessed: June 17, 2025. [Online]. Available at: https://belkasoft.com/android-whatsapp-acquisition

[87] « What is Rooting and Jailbreaking – HWG Sababa ». Accessed: June 17, 2025. [Online]. Available at: https://www.hwgsababa.com/en/what-is-rooting-and-jailbreaking/

[88] « Appareils rootés : définition, avantages et risques de sécurité | Okta ». Accessed: June 17, 2025. [Online]. Available at: https://www.okta.com/identity-101/rooted-device/

[89] « Should you root your Android device? Pros and cons | McAfee ». Accessed: June 17, 2025. [Online]. Available at: https://www.mcafee.com/learn/what-is-a-rooted-android-device/

[90] « What Is Rooting? Rooted Devices & Android Root Access ». Accessed: June 17, 2025. [Online]. Available at: https://www.avast.com/c-rooting-android

[91] jiggunjer, « Answer to “What does « to root a phone » mean?” », Android Enthusiasts Stack Exchange. Accessed: June 17, 2025. [Online]. Available at: https://android.stackexchange.com/a/129384

[92] « Root Detection | Glossary ». Accessed: June 17, 2025. [Online]. Available at: https://docs.talsec.app/glossary/root-detection

[93] « Rooted Device ». Accessed: June 17, 2025. [Online]. Available at: https://zimperium.com/glossary/rooted-device

[94] « 3 Effective Methods to Read Encrypted WhatsApp Messages ». Accessed: June 17, 2025. [Online]. Available at: https://www.airdroid.com/parent-control/read-encrypted-whatsapp-messages/

[95] « À propos des fonctionnalités d'accessibilité sur WhatsApp | Pages d'aide WhatsApp ». Accessed: June 17, 2025. [Online]. Available at: https://faq.whatsapp.com/3614672068767202/?cms_platform=android

[96] « Kotlin overview », Android Developers. Accessed: June 17, 2025. [Online]. Available at: https://developer.android.com/kotlin/overview

[97] « What Is Python? (Definition, Uses, Difficulty) », Built In. Accessed: June 17, 2025. [Online]. Available at: https://builtin.com/software-engineering-perspectives/python

[98] « What is JavaScript? - Learn web development | MDN ». Accessed: June 17, 2025. [Online]. Available at: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript

[99] « Introduction à Django - Apprendre le développement web | MDN ». Accessed: June 17, 2025. [Online]. Available at: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Django/Introduction

[100] « Canaux Django — Documentation des canaux 4.2.0 ». Accessed: June 17, 2025. [Online]. Available at: https://channels.readthedocs.io/en/latest/

[101] « Accueil - Framework Django REST ». Accessed: June 17, 2025. [Online]. Available at: https://www.django-rest-framework.org/

[102] « Introduction • Docs • Svelte ». Accessed: June 17, 2025. [Online]. Available at: https://svelte.dev/docs/kit/introduction

[103] « PostgreSQL : À propos ». Accessed: June 17, 2025. [Online]. Available at: https://www.postgresql.org/about/

[104] « What is Redis Explained? | IBM ». Accessed: June 17, 2025. [Online]. Available at: https://www.ibm.com/think/topics/redis

[105] « What is Android Studio? », GeeksforGeeks. Accessed: June 17, 2025. [Online]. Available at: https://www.geeksforgeeks.org/overview-of-android-studio/

[106] « What is Visual Studio Code? Microsoft's extensible code editor », InfoWorld. Accessed: June 17, 2025. [Online]. Available at: https://www.infoworld.com/article/2335960/what-is-visual-studio-code-microsofts-extensible-code-editor.html

[107] « Git ». Accessed: June 17, 2025. [Online]. Available at: https://git-scm.com/

[108] « Google Reveals the Dessert Name Android Q Was Most Likely to Have », Gadgets 360. Accessed: February 14, 2025. [Online]. Available at: https://www.gadgets360.com/mobiles/news/android-q-queen-cake-quince-tart-android-10-dessert-name-revealed-2093103

[109] « Improving app security and performance on Google Play for years to come », Android Developers Blog. Accessed: February 14, 2025. [Online]. Available at: https://android-developers.googleblog.com/2017/12/improving-app-security-and-performance.html

[110] « Lucky number Android 13: The latest features and updates », Google. Accessed: February 14, 2025. [Online]. Available at: https://blog.google/products/android/android-13/

[111] « Android 14: More customization, control and accessibility features », Google. Accessed: February 14, 2025. [Online]. Available at: https://blog.google/products/android/android-14/

[112] « What's new in Android 15, plus more updates », Google. Accessed: February 14, 2025. [Online]. Available at: https://blog.google/products/android/android-15/

[113] « The First Beta of Android 16 », Android Developers Blog. Accessed: February 14, 2025. [Online]. Available at: https://android-developers.googleblog.com/2025/01/first-beta-android16.html

[114] « Android 16: Confirmed features, codename, leaks, release date, and everything else we know so far », Android Authority. Accessed: February 14, 2025. [Online]. Available at: https://www.androidauthority.com/android-16-features-3484159/