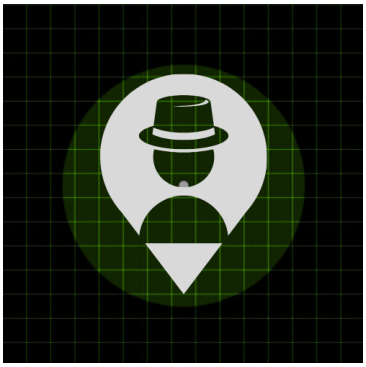
Principles of Software Development - Report MiniMap

MAHIEU Corentin - 1240038553

29 May 2025

Abstract

MiniMap is an Android application designed to assess and classify the security level of surrounding WiFi networks using a custom-built machine learning model. The application automatically scans the environment, collects WiFi network data, and stores the observations in a structured CSV file. Users can additionally save scan results in JSON format for later analysis. All collected data can be conveniently exported and shared with other applications, providing flexibility for further processing or security analysis.



1 Introduction

Along the semester, we have learned principles of software development through the course. The project consists of creating an android application in android studio IDE using as many concepts as possible we have seen related to software development.

2 MiniMap

Being really interested into cybersecurity, I chose to create an android application related to this field of application.

2.1 Main idea

MiniMap is an android application designed to scan and classify the security level of surrounding WiFi networks. Its name comes from the free and open source network tool *nmap* allowing to perform discovery and security auditing. MiniMap performs a periodic scan and displays information related to the WiFi of each network found. Observed WiFi networks are stored locally, so users can keep the information of the WiFi and can also access the location of the latter on Google Maps. This application is a passive network scan, meaning that it only collects information over a network without actively interacting with the target. It also includes different easy-to-use features presented in Section 2.2.

2.2 Features

The application includes different features allowing to easily perform a scan, store, save, consult and share data related to WiFi network.

2.2.1 WiFi Detection

The WiFi scan is perform using built-in android studio library. Every 3 seconds, a scan is perform and WiFi dots are displayed in the user interface in a sonar-like canvas. The related information are displayed in the bottom. As a scan is performed at a periodic time, we can *pause* the periodic scan to watch related data and then *resume* the scan.

2.2.2 Classification

Each observed WiFi are classified using a custom-build machine learning model. There are 3 security level: Safe, Medium and Dangerous. The model uses different characteristics such as capabilities, frequency and SSID to classify WiFi network.

2.2.3 Storing and Exporting Data

The user can stored data from a scan in a JSON file named by his own and then consult them directly in the application. He is able to consult every observed WiFi found. He can also share his JSON file and share observed WiFi as CSV file.

2.2.4 Access Location

The location (latitude and longitude) is retrieved for each WiFi network found with the application, which means that the user can know where a WiFi comes from. A button associated with each WiFi can redirect the user to the location in Google Maps.

2.2.5 Persistent options

Some options are available and saved in a data store to remember them at each application's launch. For instance, the user can enable the *automatic save* meaning that each observed WiFi will be stored locally.

2.2.6 Notification Manager

The user can enable an option to receive notifications from the application. If enabled and if the *automatic scan* option is enabled, a scan will run every 15 minutes. If a dangerous and non-stored WiFi network is observed, it will send a notification.

2.2.7 Others

Some other small features are included in the application. There are some animations when launching the application and when scanning WiFi. There is also a search bar to look for a specific JSON file or a specific observed WiFi in the stored data. Some filters have been added to look for a specific security observed WiFi, either Safe, Medium, or Dangerous when consulting stored data.

2.3 Principles of Software Development

MiniMap includes many principles of software development concepts. Those are necessary in order to create a robust, efficient, and maintainable application while ensuring scalability, good user experience, and clean code architecture.

2.3.1 Programming Language

The project is entirely coded in Kotlin, which is a modern statically typed programming language used by more than 60% professional Android users. Kotlin is an object-oriented language. It supports fundamental object-oriented programming (OOP) concepts such as classes, inheritance, encapsulation, and polymorphism.

2.3.2 Class architecture

The project is split into well-organized classes, making the code more modular and maintainable. Some components leverage Kotlin and Android-specific features, such as Jetpack Compose for UI and Coroutines for asynchronous tasks. Each class is properly commented and follows clean code principles to ensure readability and ease of future updates.

2.3.3 AI Implementation

MiniMap includes AI, as it uses a machine learning model to classify the security level of the WiFi network. It allows to get a more precise result for the classification part.

2.3.4 Versioning

A versioning system is included in the application. At each commit, I ensured to modify the application's version according to the modification and by following the *semantic versioning*. Either a huge modification in API or modification in the code or correction.

2.3.5 CI/CD Pipeline

A CI/CD set up pipeline allows to get a correct usable android application in a APK file for each push.

2.3.6 Tests

The application includes unit tests to verify the correctness of key functionalities. The test suite follows convenient pattern and uses JUnit for test execution. Mock objects are employed where needed to isolate components during testing. Continuous integration ensures tests are automatically run on each commit.

3 Conclusion

MiniMap makes sure to get closer to the application of modern software development principles in an Android context. The project combines Kotlin's expressive syntax with Jetpack Compose for responsive UI, Coroutines for efficient asynchronous operations, and clean architecture for maintainability. Key achievements include the implementation of a WiFi scanner, a machine learning classifier for WiFi security assessment, data persistence mechanisms, and intuitive user interactions and interface. The modular class structure, documentation, and CI/CD pipeline ensure the application remains scalable and easy to maintain. MiniMap serves as both a practical security tool and showcase how software engineering best practices can create reliable mobile applications. Future enhancements could expand the machine learning capabilities and refine the user experience further.
GitHub Repository: https://github.com/Fir3n0x/MiniMap