SmishingCheck

Entwickler-Dokumentation | Call 19 | Projekt ID 7232

Inhalt

1 Was ist SmishingCheck?

SmishingCheck ist eine App für iOS und Android (veröffentlicht im November 2025) für österreichische Nutzer in der Sprache Deutsch, die es ermöglicht …

- … sich über Smishing und deren Ausprägungen zu informieren.
- … SMS-Nachrichten auf bekannte Betrugsfälle zu prüfen.
- … verdächtige Nachrichten an die österreichische Meldestelle (Watchlist Internet) direkt über die App zu melden.

Die App ist über die beiden Stores (iOS App-Store; Google Play-Store) kostenlos downloadbar, hier sind die Links:

- iOS: https://apps.apple.com/at/app/smishingcheck/id6741202684
- Android: https://play.google.com/store/apps/details?id=com.coastlab7.smishingCheck

Nähere Informationen zum Projekt findet man auf der Projektseite von netidee sowie der Webseite:

- netidee: https://www.netidee.at/smishingcheck
- Website: https://smishingcheck.app/

2 Für wen ist SmishingCheck?

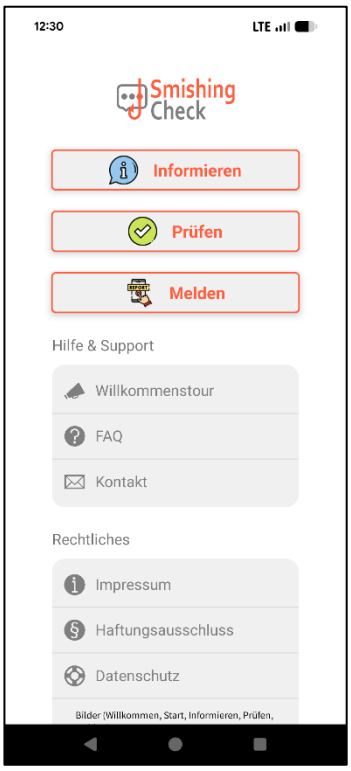
Die App SmishingCheck ist für all jene Smartphone-Nutzer (Android oder iOS) gedacht, die direkt am Handy Informationen zu Smishing erhalten wollen, SMS-Inhalte auf bekannte Smishing-Betrugsfälle prüfen wollen und Betrugsfälle direkt am Handy melden wollen. Die App kann in Österreich heruntergeladen werden, ist kostenlos und werbefrei nutzbar und ist in der Sprache Deutsch verfügbar.

Außerdem ist SmishingCheck durch die Melde-Funktion vorteilhaft für Watchlist Internet, da durch den Anreiz, Meldungen auf einfache Art und Weise über die App absenden zu können, die Dunkelziffer von bekannten Betrugsfällen reduziert wird. Es können aktuelle Trends und neue Wellen frühzeitig erkannt und dahingehend Warnungen ausgesprochen werden.

Und zu guter Letzt ist SmishingCheck auch für all jene gedacht, die das Projektergebnis weiterverwenden und mit weiteren Ideen ergänzen möchten.

3 Wie funktioniert SmishingCheck (Übersicht)?

Nachfolgend werden die Funktionen von SmishingCheck erläutert, in dem auf die einzelnen App-Screens eingegangen wird:



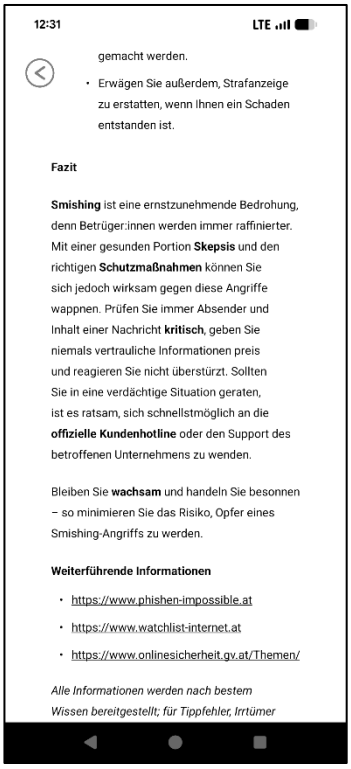
Der **Hauptschirm** beinhaltet an prominenter Stelle Schaltflächen für die drei Hauptfunktionen der App:

- Informieren,
- Prüfen
- und Melden.

Ein Klick auf einen der Schaltflächen führt zur jeweilen Funktion, die in den weiterführenden Screens erläutert wird.

Darunter befindet sich ein kleines Menü unterteilt in die Bereiche

- „Hilfe & Support“: Hier kann die Willkommenstour erneut gestartet werden, sowie sind hier die „Frequently Asked Questions (FAQ)“ und die Kontaktdaten verlinkt.
- „Rechtliches“: Mit Links zum Impressum, zum Haftungsausschluss sowie zur Datenschutzbestimmung.

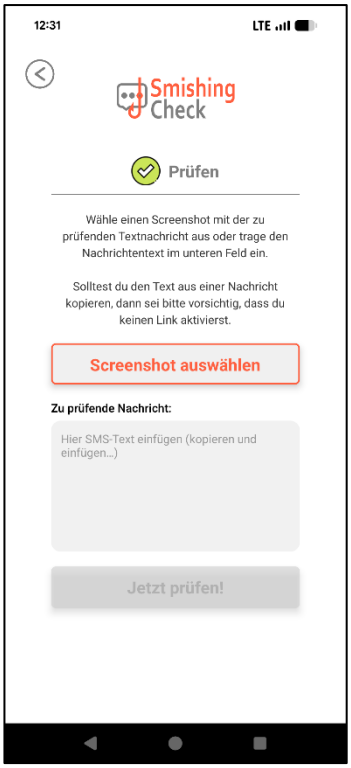


Der **Informieren**-Bereich klärt generell über Smishing auf und schildert, welche Formen von SMS-Phishing existieren. Außerdem wird darauf eingegangen, was übliche Strategien von Täter sind, auf welche Warnzeichen man achten sollte und wie man sich am besten davor schützen kann.

Es werden Smishing-Beispiele angeführt und weiterführende Informationen als Links dargestellt.

Der Informieren-Bereich ist technisch so umgesetzt, dass sehr rasch und ohne nötiges Update der App, die dargestellten Informationen erweitert oder geändert werden können. Dies ermöglicht ein rasches Reagieren auf aktuelle Trends in diesem Bereich.

Jede Unterseite der App enthält links oben eine Zurück-Schaltfläche („<“), über die man zum vorherigen Screen gelangt.



Wenn wir in den **Prüfen**-Bereich der SmishingCheck-App blicken, dann öffnet sich zuerst dieser Screen. Eine Einleitung hilf dem Nutzer, sich zurecht zu finden.

Verdächtige Texte (aus SMS- oder WhatsApp-Nachrichten) können hier als Screenshot ausgewählt werden. Das Erstellen des Screenshots muss zuvor mit den Standardfunktionen des Smartphones getätigt werden.

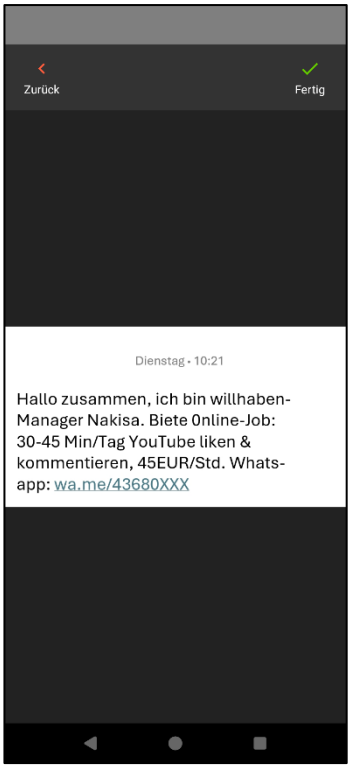
Die weitere Abfolge schildern die nachfolgenden Screens.

Sobald man ein Bild (den **Screenshot**) aus der Smartphone-internen Galerie ausgewählt hat, erscheint automatisch die Möglichkeit des Ausschneidens.

Hier kann (und sollte) dieser Teil des Screenshots ausgewählt werden, der den wesentlichen Text, welcher überprüft werden soll, beinhaltet.

Was kann und sollte so gemacht werden:

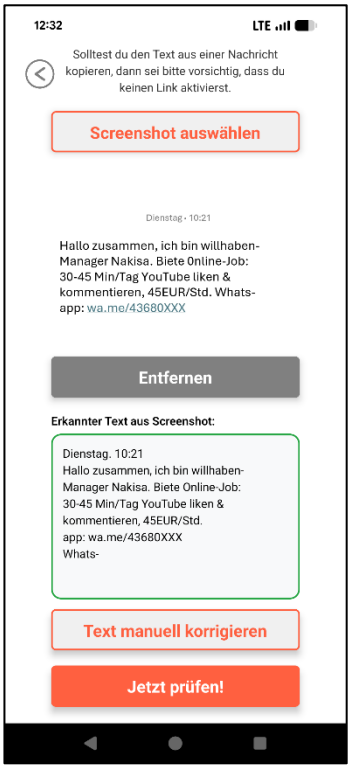
- Ein Verschieben des gesamten Rechtecks dient zum korrekten Positionieren am gewünschten Textbereich.
- Mit Hilfe der weißen Ecken des Rechtecks kann der Bereich vergrößert bzw. verkleinert werden.
- Mittels „Fertig“ (rechts oben) wird der Vorgang bestätigt.



Das ausgeschnittene **Ergebnis** des Nachrichten-Screenshots wird darauffolgend automatisch präsentiert, wie am Beispiel hier links dargestellt.

Wenn man damit zufrieden ist, ist mittels „Fertig“ (rechts oben) erneut zu bestätigen.

Möchte man den Vorgang des Ausschneidens wiederholen, so kann dies durch Betätigen von „Zurück“ (links oben) erneut durchgeführt werden.



Nach dem positiven Bestätigen des ausgeschnittenen Nachrichten-Bereichs wird automatisch der **Text aus dem Bild** ermittelt und als editierbaren Text extrahiert.

Man sieht dies hier links in der grün umrandeten Box.

Dieser Screen ermöglicht folgende Funktionen:

- Screenshot auswählen: Ein neuerliches Wählen eines Screenshots inkl. der nachfolgenden, bereits erläuterten Schritte, kann getätigt werden.
- Entfernen: Der aktuell gewählte Screenshot kann entfernt werden.
- Text manuell korrigieren: Falls die Texterkennung nicht korrekt gearbeitet hat, ist hier ein manuelles Nachkorrigieren möglich. Durch Betätigen dieser Schaltfläche wird das Bearbeiten des grün umrandeten Textfeldes ermöglicht.
- Jetzt prüfen: Der ermittelte Text wird auf Smishing-Kennzeichen überprüft (mehr dazu im nachfolgenden Screen).

Die **Prüfung** des zuvor gewählten Textes erfolgt nun – was etwas Zeit in Anspruch nimmt (wenige Sekunden, in der Regel).

Was hier passiert:

Der Textinhalt wird mit folgenden Daten verglichen (mittels Ähnlichkeitsanalyse anhand einer Gewichtungsformel):

- Watchlist Internet (https://www.watchlist-internet.at) → Unseriöse Webseiten → Phishing-Alarm: Listungen mit dem Kennzeichen „SMS“.

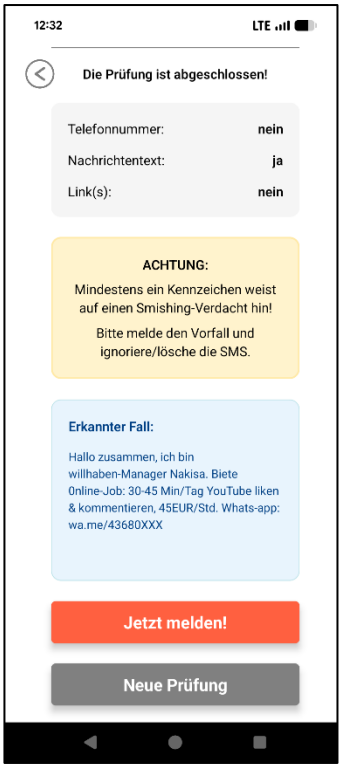
Außerdem werden Links, die im Text vorkommen, gegen folgende Listungen geprüft:

- Watchlist Internet → Unseriöse Webseiten:
  - Betrügerische Online-Shops
  - Problematische Online-Shops
  - Abo-Fallen
  - Immobilien-Agenturen
  - Urlaubsbuchung
  - Handwerksdienste
  - Speditionen
  - Jobangebote
  - Finanzbetrug
  - Automatisierte Warnungen

Außerdem werden Telefonnummern, die im Text vorkommen, gegen alle Vorkommnisse geprüft, die auf folgender Spam-Auflistung enthalten sind:

- https://spamcalls.net/de/

Wichtig: All diese Daten werden nächtlich abgerufen und zwischengespeichert, um diese in der SmishingCheck-App nutzen zu können. Daher kann es vorkommen, dass ganz aktuelle Listungen erst am nächsten Tag anschlagen.



Nachdem die **Prüfung abgeschlossen** ist, erscheint diese Hinweisseite.

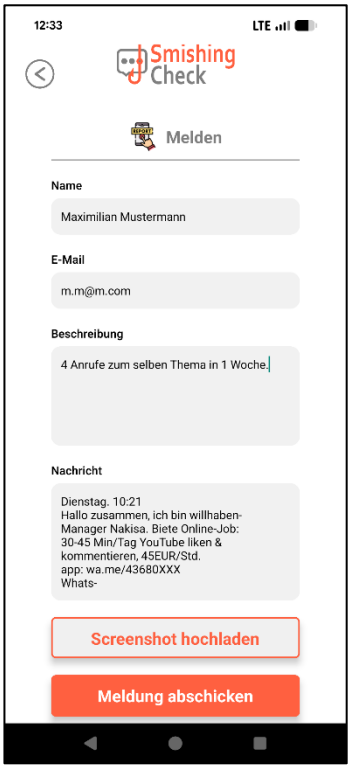
Unterteilt in die Bereiche „Telefonnummer“, „Nachrichtentext“ und „Link(s)“ werden Treffer mittels „ja/nein“ gekennzeichnet.

Wenn mindestens ein Bereich anschlägt, dann erfolgt eine Warnmeldung.

Sollte der Nachrichtentext als Verdachtsfall erkannt werden, so wird in einem blauen Bereich auch der erkannte Fall dargestellt.

Wichtig: Auch wenn kein Verdachtsfall anschlägt, heißt das nicht, dass es sich nicht trotzdem um Phishing handeln kann! Es ist daher in jedem Fall ratsam, achtsam zu sein!

Man hat nun die Möglichkeit, den Fall zu melden (mittels Schaltfläche „Jetzt melden!“), oder eine neue Prüfung durchzuführen (mittels Schaltfläche „Neue Prüfung“).



Werfen wir nun einen Blick auf die Funktion „**Melden**“.

Hier besteht die Möglichkeit, einen Betrugs- oder Verdachtsfalls an die Meldestelle von Watchlist Internet zu melden.

Wenn man direkt aus dem vorherigen Fenster „Prüfung abgeschlossen“ zum Melden gelangt, dann wird der Nachrichtentext automatisch übernommen.

Es ist nun noch notwendig, den Namen, die eigene E-Mailadresse (z.B. für Rückfragen seitens Watchlist Internet) und eine Beschreibung (bspw. ergänzende Informationen zur Nachricht) einzutragen.

Neben dem Nachrichtentext kann auch ein Screenshot übermittelt werden (mittels „Screenshot hochladen“). Hier passiert wieder eine automatische Extraktion des Textes aus dem gewählten Screenshot-Bereich (gleiches Vorgehen wie zuvor bereits geschildert).

Mittels der Schaltfläche „Meldung abschicken“ wird eine E-Mail mit den eingetragenen Daten und Informationen an die Meldestelle von Watchlist Internet geschickt.

Eine Bestätigungsmeldung signalisiert den erfolgreichen Versand.

Beim allerersten App-Start (sowie beim Betätigen des Menüeintrages „Willkommenstour“) erscheint eine kleine **Welcome-Tour** mit drei Screens.

Beim ersten der drei Screens müssen außerdem der Haftungsausschluss sowie die Datenschutzbestimmungen bestätigt werden, in dem die beiden Checkboxen aktiviert werden. (Die jeweiligen rechtlichen Inhalte sind dort selbstverständlich verlinkt.)

Die Willkommenstour erläutert bild- und textlich die wesentlichen Funktionen der App.

Mittels „Weiter“ und „<“ (Zurück) sowie durch ein Swipen nach links bzw. rechts kann durch die Willkommenstour durchnavigiert werden.

Der Menüeintrag „FAQ“ (**Frequently Asked Questions**) ist mit der folgenden Unterseite der SmishingCheck-Webseite verlinkt:

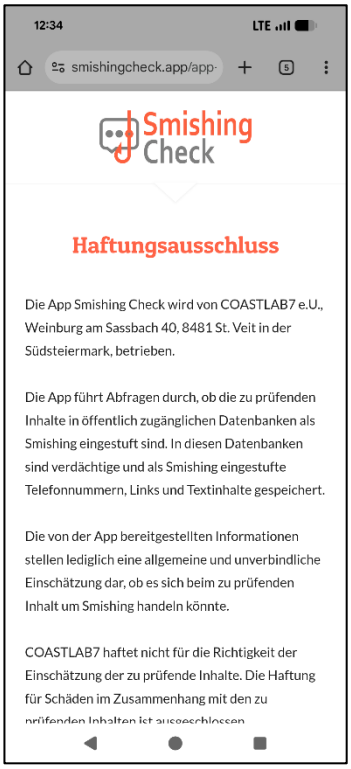
https://smishingcheck.app/faq

Hier werden wesentliche Informationen zu Smishing und natürlich zur SmishingCheck-App durch Fragen und Antworten präsentiert.

Da sich der Bereich auf der extern liegenden Webseite befindet, wirken sich Aktualisierungen direkt aus.

Der Menüeintrag „**Kontakt**“ ist mit der folgenden Unterseite der SmishingCheck-Webseite verlinkt:

https://smishingcheck.app/impressum-und-kontakt



Der Menüeintrag „**Haftungsausschluss**“ (sowie auch der Link auf der ersten Seite des Welcome-Screens) ist mit der folgenden Unterseite der SmishingCheck-Webseite verlinkt:

https://smishingcheck.app/app-haftungsausschluss

Der Menüeintrag „**Datenschutz**“ (sowie auch der Link auf der ersten Seite des Welcome-Screens) ist mit der folgenden Unterseite der SmishingCheck-Webseite verlinkt:

https://smishingcheck.app/app-datenschutzerklaerung

4 Wie funktioniert SmishingCheck (Technik)?

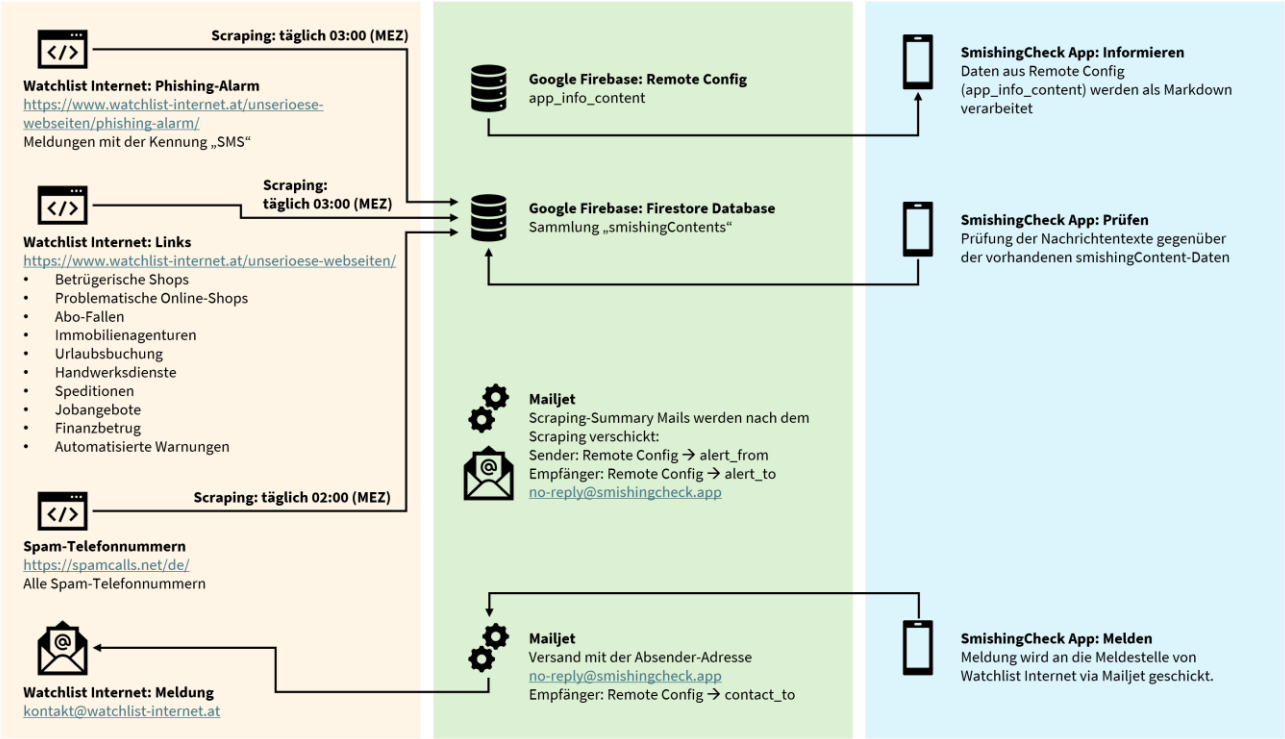
Nachfolgend wird detaillierter auf die Funktionen von SmishingCheck aus technischer Sicht eingegangen, um einen Einblick für Entwickler zu gewähren.

4.1 Allgemeines

SmishingCheck ist ein App-Projekt, das aus folgenden technischen Bereichen besteht:

- Backend und Datenbank: **Google Firebase**
  genutzt werden die folgenden Firebase Dienste:
  - Firestore Database
  - Functions
  - Remote Config
  - Crashlytics
- Frontend: Cross-Platform Entwicklung mittels **React.Native**
- Mail-Dienste: **Mailjet**

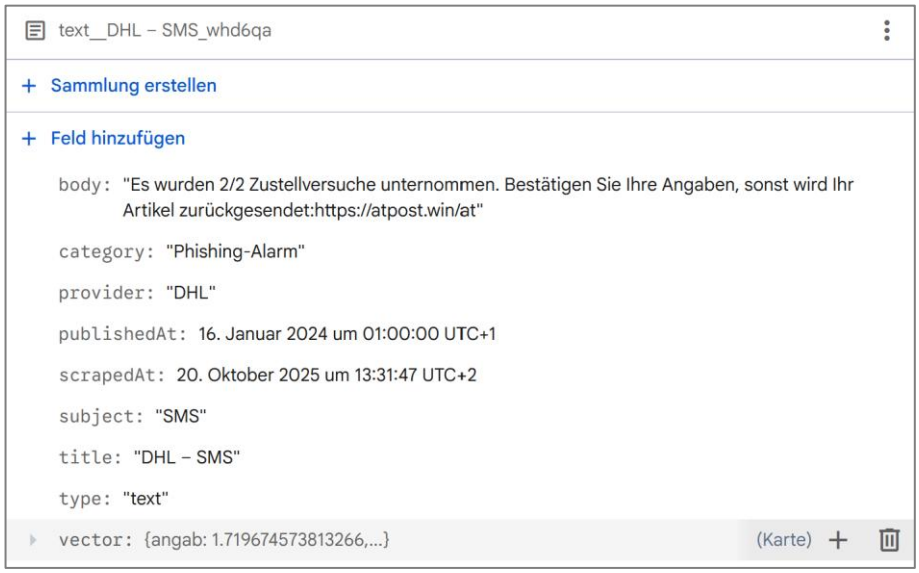
Die nachfolgende Grafik spiegelt die technischen Datenwege des Projektes wider:



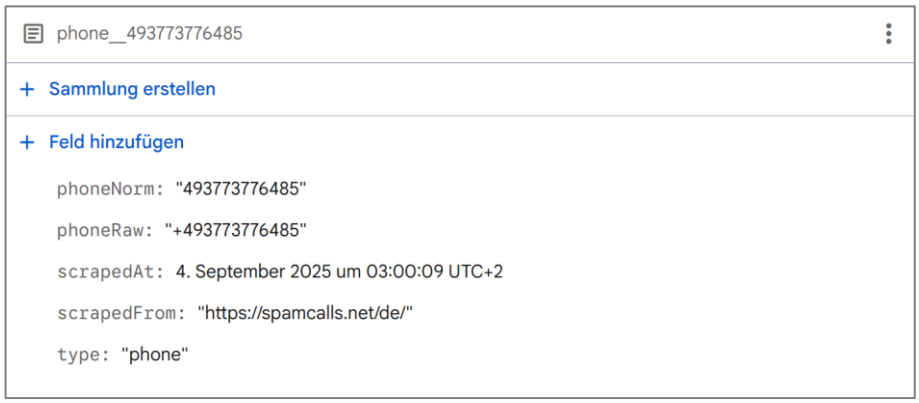
4.2 Backend

Firestore Database	Die Firestore Database besteht aus drei Sammlungen: • smishingContents • system • userEvents **smishingContents:** Hier werden die gescrapten Daten gespeichert, damit diese für Prüfungen herangezogen werden können. Unterteilt in die Kategorien:

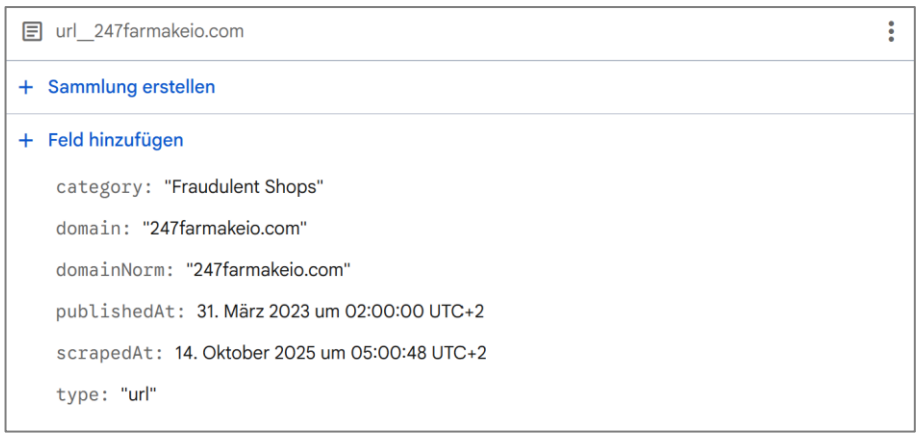
Phishing-Alarm (hier ein Beispiel):



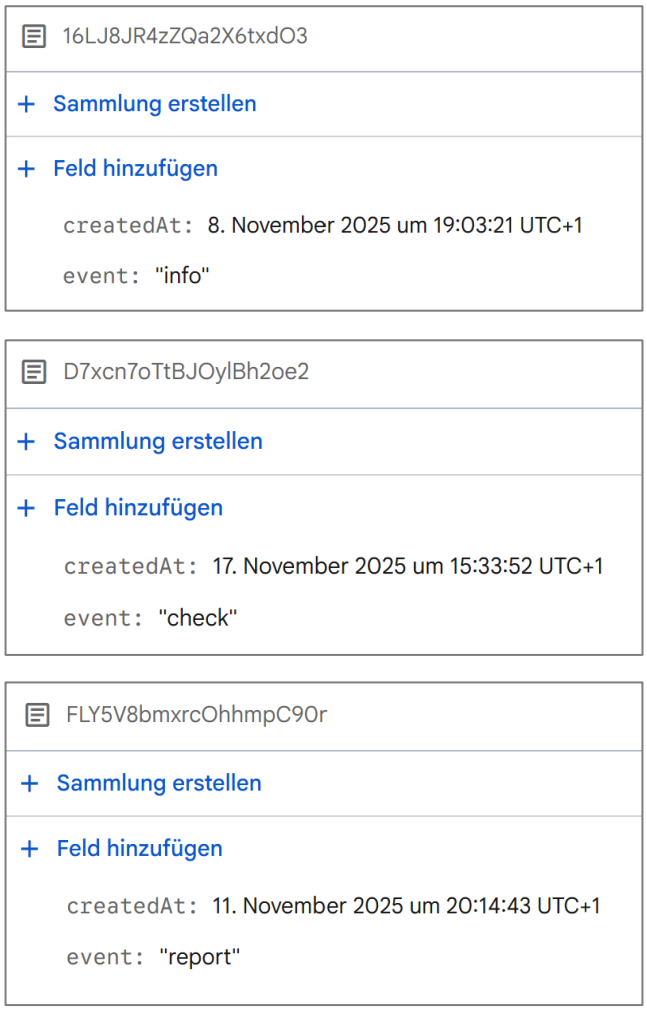
Spam-Telefonnummern (hier ein Beispiel):



Links (hier ein Beispiel):



	system: Diese Sammlung beinhaltet Informationen über den Scraping-Stand… • backfillProgress: Speichert den Fortschritt des historischen Datenimports (Backfill). Das Dokument enthält für jede Scraping-Quelle die zuletzt verarbeitete Seitennummer, sodass der Backfill-Prozess bei einem erneuten Lauf an der richtigen Stelle fortgesetzt werden kann. Struktur: ○ lastScrapedPages: Objekt mit Kategorie-Namen als Schlüssel und der zuletzt gescrapten Seitennummer als Wert (z.B. {"Fraudulent Shops": 45, "Phishing-Alarm": 12}) ○ lastUpdated: Zeitstempel der letzten Aktualisierung • importProgress: Speichert den Fortschritt des täglichen Datenimports. Wird verwendet, um den aktuellen Import-Zeitraum und -Status zu verfolgen. Enthält Metadaten zum laufenden oder zuletzt abgeschlossenen Import-Vorgang. … sowie zur Ähnlichkeitsanalyse mittels TF-IDF-Methode (Term Frequency – Inverse Document Frequency) • tfidfStats: Zentrale Statistikdaten für die TF-IDF-basierte Textähnlichkeitsanalyse. Dieses Dokument ist essenziell für die Funktion matchSmishingTfIdf und enthält: ○ docCount: Gesamtanzahl der Text-Dokumente im Korpus (Basis für IDF-Berechnung) ○ df.*: Document Frequency für jeden Term – gibt an, in wie vielen Dokumenten ein bestimmter Term vorkommt. Die Terme werden mit dem Präfix df. gespeichert (z.B. df.paket, df.zustellung, df.dhl) ○ lastUpdated: Zeitstempel der letzten Aktualisierung ○ recalculatedAt: Zeitstempel der letzten vollständigen Neuberechnung (durch fixTfIdfStats) **userEvents:** Hier werden die Daten für die Nutzungsstatistik (wird 1x jährlich manuell hier veröffentlicht: https://www.opendataportal.at/katalog/de/dataset/smishingcheck-nutzungsstatistik

	) gesammelt. Sobald ein User in der App auf einen der drei Bereiche „Informieren“, „Prüfen“ oder „Melden“ wechselt, wird hier ein Eintrag geschrieben. Es folgen drei Beispiel-Einträge für die drei App-Bereiche: 
Functions	**sendContactEmail:** Callable Function, die Kontaktformular-Anfragen aus der App verarbeitet und per E-Mail versendet. Die Funktion validiert die Eingabedaten (Name, E-Mail, Beschreibung, Nachricht) und unterstützt optionale Screenshot-Anhänge als Base64-kodierte Bilder. Der Versand erfolgt über die Mailjet API an die in Remote Config hinterlegte Empfängeradresse (contact_to). Bei erfolgreichem Versand wird eine Bestätigung an die App zurückgegeben. **fixTfIdfStats:** Callable Function zur Reparatur und Neuberechnung der TF-IDF-Statistiken. Diese Wartungsfunktion durchläuft alle Text-Dokumente in der

	smishingContents-Sammlung, bereinigt HTML-Entities aus den Body-Feldern, berechnet die Document Frequency (DF) für alle Terme neu und aktualisiert anschließend die TF-IDF-Vektoren aller Dokumente. Wird manuell ausgeführt, wenn Inkonsistenzen zwischen docCount und der tatsächlichen Dokumentanzahl auftreten. **getAppInfo:** Callable Function, die den Markdown-Inhalt für den „Informieren"-Bereich der App bereitstellt. Der Inhalt wird aus dem Remote Config Parameter „app_info_content" geladen, wodurch Aktualisierungen ohne App-Update möglich sind. **backfillSmsTexts:** Callable Function für den einmaligen Import historischer SMS-Phishing-Texte. Die Funktion scraped alle SMS-Einträge aus dem Phishing-Alarm-Bereich von Watchlist Internet ab dem 01.01.2023 und aktualisiert die TF-IDF-Statistiken entsprechend. Wird manuell über einen HTTP-Aufruf gestartet. **scrapeWatchlistBackfill:** Scheduled Function für den progressiven Import historischer Watchlist-Daten. Läuft täglich um 05:00 Uhr (Europe/Vienna), eine Stunde nach dem Daily-Scraping. Die Funktion arbeitet budgetbewusst mit konfigurierbarer Fenstergröße und importiert schrittweise ältere Einträge aus allen URL- und Text-Quellen. Der Fortschritt wird in der system-Sammlung gespeichert. Der Backfill kann über Remote Config (progressive_import_enabled) aktiviert/deaktiviert werden. **logUserEvent:** Callable Function zur Erfassung von Nutzungsstatistiken. Wird von der App aufgerufen, wenn ein User einen der drei Hauptbereiche („Informieren", „Prüfen", „Melden") betritt. Der Event-Name („info", „check", „report") wird zusammen mit einem Zeitstempel in der userEvents-Sammlung gespeichert. Diese Daten werden jährlich aggregiert und im Open Data Portal veröffentlicht.

	matchSmishingTfIdf: Callable Function und Kernstück der SMS-Analyse. Nimmt einen Nachrichtentext entgegen und prüft diesen auf Smishing-Indikatoren: • **URL-Analyse:** Extrahiert URLs aus dem Text, löst URL-Shortener auf und vergleicht die normalisierten Domains gegen die gespeicherten betrügerischen URLs. • **Telefonnummern-Analyse:** Extrahiert und normalisiert Telefonnummern und prüft diese gegen die Spam-Telefonnummern-Datenbank. • **Text-Ähnlichkeit:** Berechnet TF-IDF-Vektoren und ermittelt mittels Cosinus-Ähnlichkeit Übereinstimmungen mit bekannten Phishing-SMS-Texten (Schwellenwert: 0.45). Die Funktion gibt ein strukturiertes Ergebnis mit allen gefundenen Treffern zurück. **scrapeWatchlistDaily:** Scheduled Function für den täglichen Import aktueller Bedrohungsdaten. Läuft täglich um 04:00 Uhr (Europe/Vienna) und scraped Einträge der letzten 14 Tage aus folgenden Quellen: • Watchlist Internet: Alle URL-Listen (Betrügerische Shops, Abo-Fallen, etc.) • Watchlist Internet: Phishing-Alarm (SMS-Einträge) Nach dem Scraping werden die TF-IDF-Statistiken aktualisiert und eine Zusammenfassung per E-Mail an die konfigurierten Empfänger versendet. **scrapeSpamcallPhones:** Scheduled Function für den täglichen Import von Spam-Telefonnummern. Läuft täglich um 03:00 Uhr (Europe/Vienna) und scraped alle auf https://spamcalls.net/de/ gelisteten Telefonnummern. Die Nummern werden normalisiert und in der smishingContents-Sammlung gespeichert. Nach Abschluss wird eine Zusammenfassung per E-Mail versendet.
Remote Config	**alert_from:** Definiert die Sender-Mailadresse für System-Nachrichten wie beispielsweise der Scraping-Summary Mails.

alert_to:

Definiert die Empfänger-Mailadresse(n) für System-Nachrichten wie beispielsweise der Scraping-Summary Mails. Mehrere Empfänger werden mittels Semikolons (ohne Leerzeichen dazwischen) getrennt.

app_info_content:

Definiert den Inhalt des „Informieren"-Bereichs, welcher in der App angezeigt wird. Der Inhalt kann hier mittels Markdown-Attributen gestaltet werden.

contact_to:

Definiert die Empfänger-Mailadresse(n), welche eine aus der App abgeschickte Meldung empfangen. Mehrere Empfänger werden mittels Semikolons (ohne Leerzeichen dazwischen) getrennt.

Lt. Projektdefinition werden Meldungen an kontakt@watchlist-internet.at übermittelt. Diese landen dann im Ticketsystem von Watchlist Internet und werden redaktionell geprüft und weiterverarbeitet.

mailjet_api_key_private:

Hier kann der „Private Key" der Mailjet API hinterlegt werden. (Wird fürs Versenden von Mails genutzt, um keinen eigenen Mailserver betreiben zu müssen.)

mailjet_api_key_public:

Hier kann der „Public Key" der Mailjet API hinterlegt werden. (Wird fürs Versenden von Mails genutzt, um keinen eigenen Mailserver betreiben zu müssen.)

max_operations_per_batch:

Maximale Anzahl an Firestore-Schreiboperationen pro Scraping-Durchlauf. Dient zur Kostenkontrolle und verhindert übermäßige Datenbankzugriffe bei großen Datenmengen. Wenn das Limit erreicht wird, stoppt der Scraper und setzt beim nächsten Lauf fort.

	max_pages_per_source: Maximale Anzahl an Seiten, die pro Datenquelle während eines Scraping-Durchlaufs verarbeitet werden. Begrenzt die Laufzeit und Kosten einzelner Scraping-Jobs. Das Limit greift erst, nachdem relevante Einträge im Zielzeitraum gefunden wurden. **progressive_import_enabled:** Aktiviert oder deaktiviert den progressiven Backfill-Import historischer Daten. Wenn aktiviert (true), läuft die Funktion scrapeWatchlistBackfill täglich um 05:00 Uhr und importiert schrittweise ältere Einträge. Nach Abschluss des historischen Imports sollte dieser Parameter auf false gesetzt werden. **progressive_import_start_date:** Startdatum für den progressiven Import im Format YYYY-MM-DD. Definiert, ab welchem Datum historische Daten importiert werden sollen. Wird in Kombination mit progressive_import_enabled verwendet. **progressive_import_window_months:** Zeitfenster in Monaten für den progressiven Import. Bestimmt, wie viele Monate an Daten pro Backfill-Durchlauf verarbeitet werden. Kleinere Werte reduzieren die Last pro Durchlauf, benötigen aber mehr Tage für den vollständigen Import. **rate_limit_delay_ms:** Wartezeit in Millisekunden zwischen dem Scraping einzelner Seiten. Verhindert eine Überlastung der externen Datenquellen (Watchlist Internet, Spamcalls.net) und reduziert das Risiko von IP-Sperren oder Rate-Limiting durch die Quellserver. **scraper_timeout_ms:** Timeout in Millisekunden für HTTP-Anfragen an externe Datenquellen. Wenn eine Seite nicht innerhalb dieser Zeit antwortet, wird die Anfrage abgebrochen

	und ein Fehler protokolliert. Zu niedrige Werte können bei langsamen Quellservern zu unnötigen Fehlern führen.
Crashlytics	Crashlytics wurde technisch in die App integriert, sodass im Crashlytics-Dashboard von Firebase Probleme und Abstürze getrackt und eingesehen werden können.

4.3 Frontend

Das Frontend ist als Cross-Platform Mobile App mit React Native und Expo umgesetzt und läuft auf iOS und Android.

Technologie Stack:

Komponente	Technologie
Framework	React Native
Build-System	Expo SDK
Sprache	TypeScript
Navigation	expo-router (file-based routing)
Crash-Reporting	Firebase Crashlytics

OCR-Texterkennung:

Die App nutzt ***expo-text-extractor*** für die automatische Texterkennung aus Screenshots. Mit ***expo-dynamic-image-crop*** können Nutzer den relevanten Bereich eines Screenshots zuschneiden, bevor die OCR-Erkennung startet. Alternativ kann der SMS-Text manuell eingegeben werden.

Backend-Kommunikation:

Die Kommunikation mit dem Backend erfolgt über Firebase Cloud Functions (Region: europe-west3).

Folgende Funktionen werden aufgerufen:

Funktion	Erläuterung
matchSmishingTfIdf	Smishing-Analyse des Nachrichtentexts
sendContactEmail	Meldung an Watchlist Internet absenden
getAppInfo	Inhalte für den Informieren-Bereich laden
logUserEvent	Nutzungsstatistik erfassen

Lokale Datenspeicherung:

Mit ***AsyncStorage*** wird lokal gespeichert, ob der Nutzer die Willkommenstour bereits gesehen hat.

4.4 Quellcode

Der bereinigte und dokumentierte Quellcode (unter der Lizenz GPL 3.0) der Cloud Functions sowie der App selbst befindet sich hier:

Backend: https://github.com/CL7-abreznik/smishingcheck_backend

Frontend: https://github.com/CL7-abreznik/smishingcheck_frontend