PhishSafe SDK

Technical Documentation

Version 1.0

Brought to you by team SudoCode

Contents

Introduction and Overview

Business Goals and Objectives

PhishSafe SDK is a behavioral authentication engine designed to detect post-login fraud and phishing threats in mobile applications through continuous behavioral tracking and trust scoring. The system aims to:

- Provide real-time fraud detection using behavioral biometrics
- Reduce account takeover incidents by 85% within first year
- Maintain user privacy with on-device processing
- Integrate seamlessly with existing applications
- Provide actionable security insights through comprehensive dashboards

Target Audience

- **Mobile App Users**: End-users who need protection against phishing
- **Security Teams**: Fraud analysts and security professionals
- **Developers**: Mobile app developers integrating the SDK
- **Product Managers**: Stakeholders evaluating security solutions

Core Functionality

The PhishSafe SDK provides the following key features:

Feature	Description
Session Analytics	Monitors complete user session from login to logout
Location Tracking	Verifies geographical consistency during sessions
Screen Recording Detection	Alerts when screen capture is active
Trust Scoring	Generates real-time risk assessment (0-100 scale)
Data Export	Stores session logs in standardized JSON format
Dashboard Integration	Provides visualization tools for security teams

Installation and Setup

System Requirements

Component	Requirements
Development Environment	Flutter 3.8+, Dart 3.0+
Operating Systems	Android 10+, iOS 14+
Hardware	Minimum 2GB RAM, ARM64/x86 CPU
Permissions	Storage, Location (optional)
Dependencies	device_info_plus, shared_preferences, path_provider

Integration Steps

Adding the Dependency

Add the PhishSafe SDK to your Flutter project:

```
dependencies:
  phishsafe_sdk:
    git:
      url: https://github.com/swrjks/phishsafe_sdk.git
      ref: v2.1
```

Android Configuration

Add these permissions to `AndroidManifest.xml`:

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

iOS Configuration

Add these entries to `Info.plist`:

```
<key>NSLocationWhenInUsearchitDescription</key>
<string>For fraud detection purposes</string>
<key>UIBackgroundModes</key>
<array>
  <string>location</string>
</array>
```

Trust Score System

Risk Classification

The SDK calculates a trust score based on user behavior patterns:

Score Range	Risk Level	Description
70-100	Safe	Normal user behavior
40-70	Slightly Risky	Unusual patterns detected
Below 40	Risky	Potential fraudulent activity

Table 2: Trust Score Interpretation

Scoring Factors

The trust score is calculated using:

- Tap patterns and durations
- Swipe gestures and velocities
- Screen navigation sequences
- Session duration characteristics
- Device consistency checks
- Location verification (if enabled)

Module-by-Module Breakdown

1. SessionTracker

Tracks session start and end times.

```
// Call on user login
PhishSafeSDK.initSession();

// Call on user logout
PhishSafeSDK.endSession();
```

2. ScreenRecordingDetector

Automatically detects active screen recording during sessions. Shows warning dialog if detected.

3. DeviceInfoLogger

Collects device information (model, OS version) at session end.

4. TapTracker

Captures tap interactions:

```
PhishSafeSDK.onTap("ScreenName");
```

5. SwipeTracker

Automatically tracks swipe gestures (direction, duration, distance).

6. NavigationLogger

Logs screen visits:

```
PhishSafeSDK.onScreenVisit("Home");
```

7. InputTracker

Monitors sensitive inputs:

```
PhishSafeSDK.recordTransactionAmount("5000");
PhishSafeSDK.recordFDBroken();
PhishSafeSDK.recordLoanTaken();
```

8. LocationTracker

Captures device location once per session (if permissions granted).

9. TrustScoreEngine

Computes risk score based on behavior patterns.

10. ExportManager

Saves session data to:

```
/sdcard/Download/PhishSafe/
```

11. LocalStorage

Helper for storing temporary data using SharedPreferences.

12. PhishSafeTrackerManager

Internal manager coordinating all tracking components.

13. PhishSafeSDK

Main public interface for all SDK functionality.

14. RouteAwareWrapper

Tracks screen durations automatically:

```
RouteAwareWrapper(
  screenName: "Home",
  observer: routeObserver,
  child: HomeScreen(),
);
```

Data Collection Specification

List of Collected Features

The PhishSafe SDK collects the following behavioral and contextual data points:

Session Metrics

Feature	Description and Collection Method
session.start	Timestamp when `initSession()` is called, recorded in ISO-8601 format at millisecond precision
session.end	Timestamp when `endSession()` is called or session times out (after 30 seconds of inactivity)
session.duration_seconds	Automatically calculated as difference between start and end timestamps, rounded to nearest second

Device Information

Feature	Description and Collection Method
device.platform	"Android" or "iOS" detected via `device_info_plus` package during session initialization
device.brand	Manufacturer name (e.g., "Samsung", "Apple") collected from system APIs
device.model	Device model identifier (e.g., "iPhone14,3", "SM-G998B")
device.manufacturer	Hardware manufacturer (e.g., "Google", "Samsung Electronics")
device.version	OS version string (e.g., "Android 13", "iOS 16.4")
device.sdk_int	API level number (Android) or iOS version code (e.g., 33 for Android 13)

Location Data

Feature	Description and Collection Method
location.latitude	Last known latitude in decimal degrees (only collected if location permissions granted)
location.longitude	Last known longitude in decimal degrees (uses fused location provider on Android)

Interaction Tracking

Feature	Description and Collection Method
tap_durations_ms	Array of milliseconds between touch down and up events (sampled at 100ms intervals)
tap_events	List of objects containing: {`timestamp: DateTime, screen: String, x: double, y: double, zone: "top/bottom/left/right/middle"`}

swipe_events	Array of swipe gestures with: {direction: "left/right/up/down", velocity: px/ms, distance: px, duration: ms}

Navigation Patterns

Feature	Description and Collection Method
screens_visited	Chronological list of screen names collected via `RouteAwareWrapper`
screen_durations	Map of screen names to total visible time in seconds (rounded to 1 decimal place)

Security Anomalies

Feature	Description and Collection Method
screen_recording_detected	Boolean flag (true/false) set by `ScreenRecordingDetector` when screen capture is active

Financial Context

Feature	Description and Collection Method
session_input.transaction_amount	Numeric value set via `recordTransactionAmount()`. Represents monetary values in base currency units.
session_input.fd_broken	Boolean flag (true/false) set when `recordFDBroken()` is called. Indicates fixed deposit termination.
session_input.loan_taken	Boolean flag (true/false) set when `recordLoanTaken()` is called. Indicates loan application.
session_input.time_from_login_to_fd	Duration in seconds between session start and FD event. Precision: 1 decimal place.
session_input.time_from_login_to_loan	Duration in seconds between login and loan action. Measured from session start.
session_input.time_between_fd_and_loan	Time difference in seconds between FD and loan actions. Only recorded if both occur.
session_input.time_to_logout	Duration from last financial action to session end. Helps detect abrupt session termination.

Data Privacy Considerations

- All data is processed locally on the device
- Location data requires explicit user permission
- No personal identifiers (email, phone numbers) are collected
- Raw tap coordinates are anonymized by converting to screen zones
- Data retention period: 30 days (configurable)

Architecture Overview

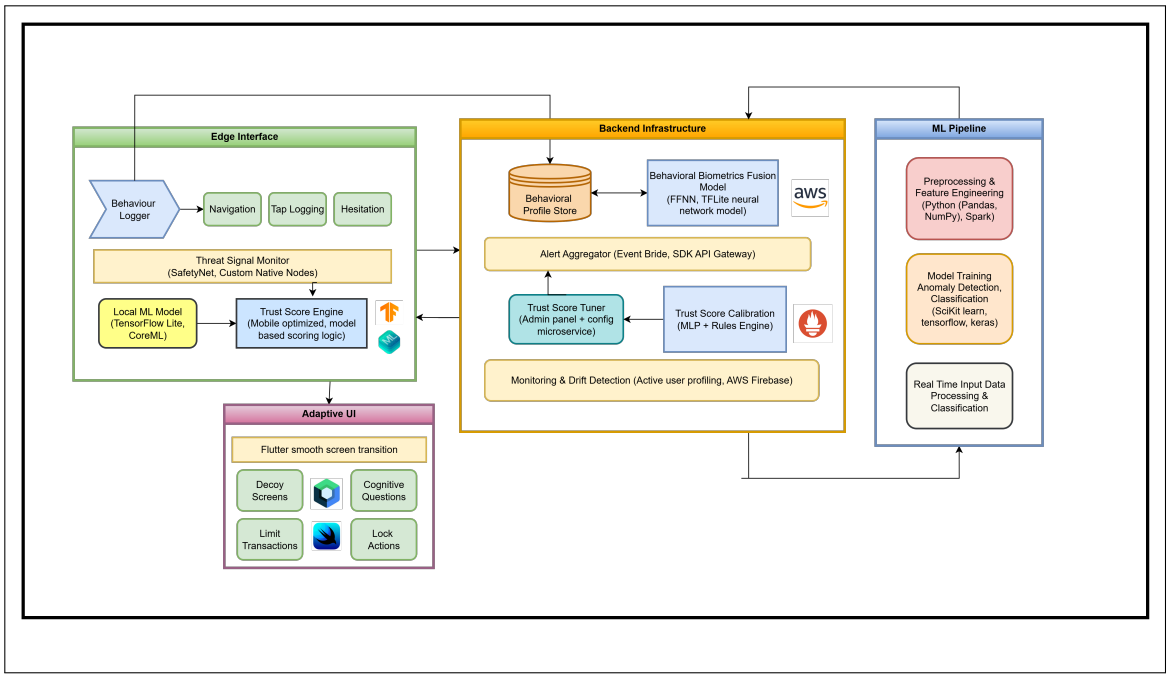

Figure 1: PhishSafe SDK Architecture Overview

The PhishSafe SDK follows a modular architecture with these key components:

- **Data Collection Layer**: Handles user interaction monitoring
- **Analysis Engine**: Processes behavioral patterns
- **Risk Assessment Module**: Calculates trust scores
- **Data Storage**: Manages session logs

Implementation Guide

Basic Integration

Initialize the SDK

Add initialization in your app's main entry point:

```dart
void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  await PhishSafeSDK.configure(
    PhishSafeConfig(
      enableLocationTracking: true,
      logLevel: LogLevel.info,
    ),
  );

  runApp(MyApp());
}
```

Session Management

Wrap your authentication flow:

```dart
void loginUser() async {
  try {
    await PhishSafeSDK.initSession();
    // Proceed with login
  } catch (e) {
    // Handle error
  }
}

void logoutUser() {
  PhishSafeSDK.endSession();
  // Clear user session
}
```

Module Integration Details

Main Interface – phishsafe_sdk.dart

What it does: This file exposes all SDK functionalities to the external Flutter app, including session management, screen visit logging, and swipe/tap events.

```dart
import 'package:phishsafe_sdk/phishsafe_sdk.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  // Start PhishSafe SDK session
  PhishSafeSDK.initSession();

  runApp(MyApp());
}
```

```dart
// When session ends (e.g., user logs out)
await PhishSafeSDK.endSession();
```

Always call `initSession()` after successful authentication and `endSession()` when user explicitly logs out.

Core Manager – PhishSafeTrackerManager

What it does: Manages all trackers, detectors, and backend communication via a singleton.

```
final tracker = PhishSafeTrackerManager();
tracker.recordTapPosition(
  screenName: "HomePage",
  tapPosition: Offset(100, 150),
  tapZone: "center",
);
```

For most use cases, use the higher-level `PhishSafeSDK` interface instead of accessing the manager directly.

RouteAwareWrapper

What it does: Wraps screens to log visits and time spent.

```
final RouteObserver<PageRoute> routeObserver = RouteObserver<PageRoute>();

MaterialApp(
  navigatorObservers: [routeObserver],
  home: RouteAwareWrapper(
    screenName: "HomePage",
    observer: routeObserver,
    child: HomeScreen(),
  ),
);
```

Use consistent screen names across your app for accurate navigation pattern analysis.

GestureWrapper

What it does: Captures taps and swipes with position and timing.

```
GestureWrapper(
  screenName: "TransferPage",
  child: Scaffold(
    body: Center(child: Text("Make a transfer")),
  ),
)
```

Wrap screens containing sensitive actions like money transfers for comprehensive gesture tracking.

TapTracker

What it does: Tracks tap positions, zones, durations.

```
PhishSafeTrackerManager().recordTapPosition(
  screenName: "LoginPage",
  tapPosition: Offset(80, 120),
  tapZone: "top_left",
);
```

The SDK automatically tracks taps in wrapped widgets - manual recording is only needed for custom gesture handlers.

SwipeTracker

What it does: Records swipe positions, speed, and duration.

```
PhishSafeTrackerManager().recordSwipeMetrics(
  screenName: "TransferPage",
  durationMs: 300,
  distance: 250.0,
  speed: 0.83,
);
```

Swipe velocity patterns are strong indicators of authentic vs. fraudulent behavior.

InputTracker

What it does: Logs login, FD break, loan taken, transaction timing.

```
PhishSafeTrackerManager().recordFDBroken();
PhishSafeTrackerManager().recordLoanTaken();
PhishSafeTrackerManager().recordWithinBankTransferAmount("100000");
```

Call these methods immediately after financial transactions occur for most accurate timing data.

LocationTracker

What it does: Requests and returns geolocation.

```
Position? position = await LocationTracker().getCurrentLocation();
print("Latitude: ${position?.latitude}, Longitude: ${position?.longitude}");
```

Location tracking requires explicit user permission - handle permission requests gracefully.

NavigationLogger

What it does: Logs screen visits with timestamps.

```
PhishSafeTrackerManager().onScreenVisited("LoanSummaryScreen");
```

For Flutter apps, prefer `RouteAwareWrapper` over manual screen visit logging.

SessionTracker

What it does: Tracks session start/end and duration.

```
PhishSafeTrackerManager().startSession();
// ...user actions...
await PhishSafeTrackerManager().endSessionAndExport();
```

Sessions automatically time out after 30 minutes of inactivity.

ScreenRecordingDetector

What it does: Detects if screen recording is active.

```
final isRecording = await ScreenRecordingDetector().isScreenRecording();

if (isRecording) {
  print("WARNING: Screen recording is active!");
}
```

Consider showing a warning to users when screen recording is detected during sensitive operations.

DeviceInfoLogger

What it does: Gets platform, model, brand info.

```
final info = await DeviceInfoLogger().getDeviceInfo();
info.forEach((k, v) => print("$k -> $v"));
```

Device fingerprinting helps detect suspicious device changes between sessions.

LocalStorage

What it does: Simple key-value storage using SharedPreferences.

```
final storage = LocalStorage();

await storage.saveString("user_id", "user123");
final userId = await storage.readString("user_id");
await storage.deleteKey("user_id");
```

Use for temporary session data - not for sensitive information.

ExportManager

What it does: Exports session logs to local + cloud.

```
final manager = ExportManager();
final dummySession = {
  "session": {"start": "2025-07-27T10:00:00Z", "end": "2025-07-27T10:10:00Z"},
  "tap_events": [],
};

await manager.exportToJson(dummySession, "session_log");
```

Default Log Storage Location Exported logs are stored in `/sdcard/Download/PhishSafe/` by default.

ApiService

What it does: Sends events to the backend Flask API.

```
await ApiService.sendTapEvent(
  screenName: "Dashboard",
  position: Offset(150, 200),
  tapZone: "bottom_right",
);
```

```
await ApiService.sendScreenVisit("LoginPage");
await ApiService.sendSessionEnd(DateTime.now());
```

The SDK handles API communication automatically - manual calls are rarely needed.

Full Integration Sample

```
final RouteObserver<PageRoute> observer = RouteObserver<PageRoute>();

MaterialApp(
  navigatorObservers: [observer],
  home: RouteAwareWrapper(
    observer: observer,
    screenName: "LoginScreen",
    child: GestureWrapper(
      screenName: "LoginScreen",
```

```
      child: LoginScreen(),
    ),
  ),
);
```

This comprehensive wrapping provides the most complete behavioral data collection.

Data Storage

All session logs are saved in JSON format at:

```
/sdcard/Download/PhishSafe/
```

Sample log structure:

```
{
  "session_id": "abc123",
  "start_time": "2025-07-20T10:00:00Z",
  "device_info": {
    "model": "Pixel 6",
    "os": "Android 14"
  },
  "trust_score": 85,
  "anomalies": []
}
```