P.PORTO

ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO

LICENCIATURA EM SEGURANÇA INFORMÁTICA EM REDES DE COMPUTADORES

Relatório de Análise Forense Comportamental da Aplicação Pacer Health

Realizado por: Hugo Leite Martins (8230273) e Rúben Uth (8210481)

Unidade Curricular: Análise Forense Digital

Novembro 2024

Índice

Índice de Figuras

1. Introdução

Este relatório consiste na realização de uma análise forense comportamental à aplicação Pacer Health, a qual utiliza a localização do utilizador (GPS) para monitorizar as atividades físicas enquanto este as realiza e regista, automaticamente, os passos, distâncias, calorias queimadas e tempo ativo.

Iremos começar por explicar a preparação que fizemos com vista a proceder à análise da aplicação e, de seguida, iremos identificar as permissões as exigidas pela mesma, as alterações verificadas nas suas bases de dados aquando da sua utilização e de que modo é que informação de localização (GPS) é utilizada e armazenada.

As principais ferramentas para análise da aplicação foram as seguintes:

- Android Studio – para extração das bases de dados da aplicação;
- Jadx-gui – para a análise do apk da aplicação;
- Data Grip – para análise das bases de dados da aplicação;
- DB Browser (SQlite) – para análise das bases de dados da aplicação.

2. Preparação para a análise

Para dar início à investigação forense, foi selecionado o telemóvel/emulador a ser utilizado na análise, bem como o tipo de imagem a ser criado para o mesmo.

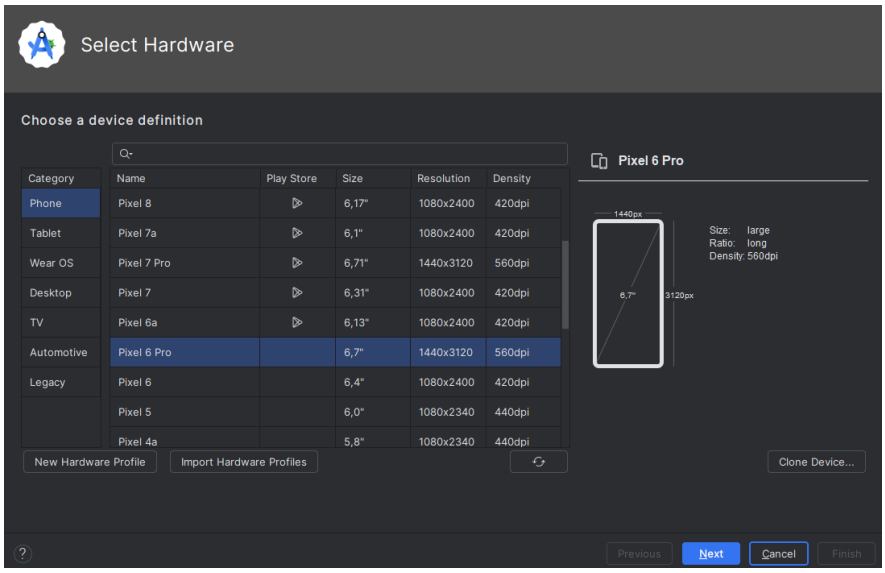

Figura 1 - Selecionar tipo de telemóvel

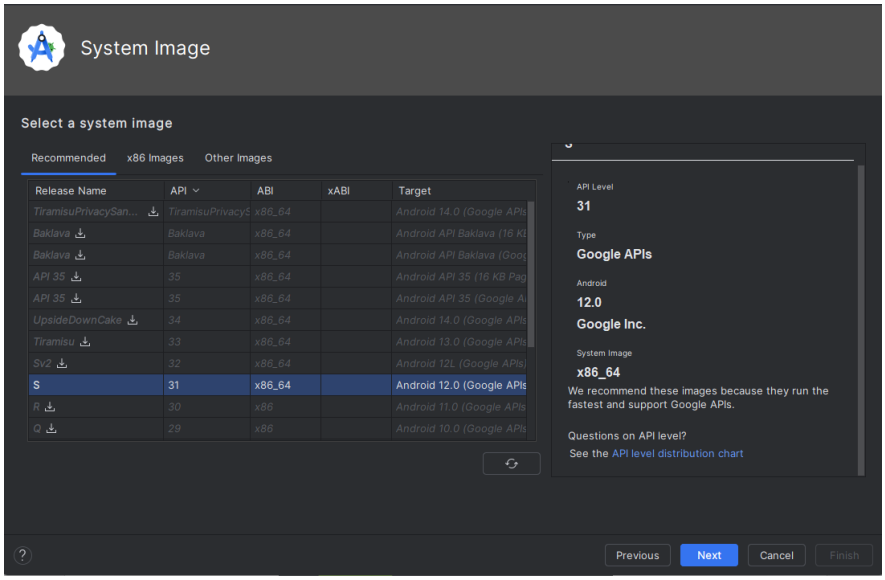

Figura 2 - Selecionar a imagem do telemóvel

Após a instalação do emulador, avançamos para a instalação da aplicação que será analisada. Para esse processo, utilizamos o Aptoide, uma aplicação destinada ao download de APKs.

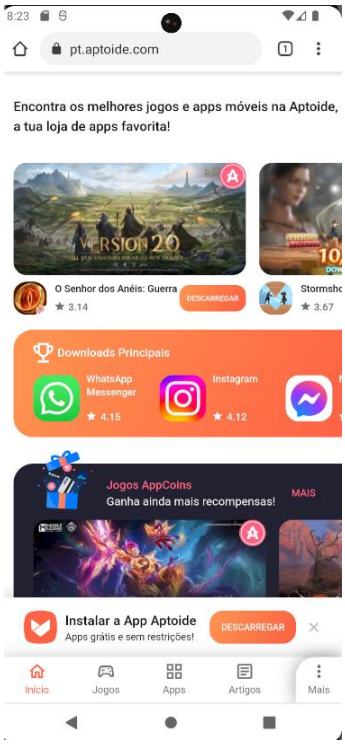

Figura 3 - Instalação da aplicação Aptoide

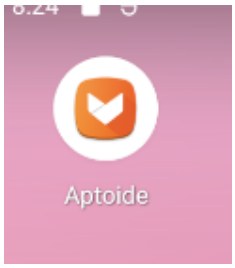

Figura 4 - Aplicação Aptoide

Após a instalação do Aptoide, procedemos à instalação da aplicação Pacer Health.

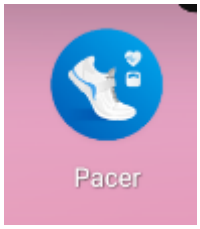

Figura 5 - Aplicação Pacer

Com a aplicação instalada, procedemos à extração da aplicação, que foi realizada através do Android Studio, utilizando uma série de comandos específicos para garantir o acesso a todas as pastas da mesma.

O primeiro passo foi aceder à pasta onde se encontra o adb (Android Debug Bridge), uma ferramenta essencial para estabelecer a comunicação entre o computador e o dispositivo Android, facilitando o teste e a manipulação de aplicações.

```
PS C:\Users\hugol\AndroidStudioProjects\MyApplication> cd C:\Android\Sdk\platform-tools
PS C:\Android\Sdk\platform-tools>
```

Figura 6 - Aceder ao caminho onde se encontra o adb

Posteriormente, executámos o comando exibido na imagem abaixo para listar todos os dispositivos conectados e ativos, permitindo avançar com os próximos passos.

```
PS C:\Android\Sdk\platform-tools> .\adb devices
List of devices attached
emulator-5554   device
```

Figura 7 - Lista de dispositos que estão disponíveis

Para continuar, foi necessário realizar o root do dispositivo, obtendo assim permissões de administrador e acesso total ao sistema.

```
PS C:\Android\Sdk\platform-tools> .\adb root
adbd is already running as root
```

Figura 8 - Realização do root ao dispositivo

Com as permissões adequadas, utilizámos o comando Shell, que nos permitiu entrar no sistema do dispositivo e navegar até a aplicação em questão.

```
PS C:\Android\Sdk\platform-tools> .\adb shell
emulator64_x86_64_arm64:/ #
```

Figura 9 - Entrar dentro do dispositivo

Uma vez dentro do dispositivo, localizámos a pasta onde está armazenada a aplicação em análise.

```
emulator64_x86_64_arm64:/ # cd /data/data
emulator64_x86_64_arm64:/data/data #
```

Figura 10 - Aceder a pasta principal do dispositivo

Após encontrar a pasta principal, utilizámos o comando ls (listar pastas e ficheiros) para identificar a localização exata da aplicação desejada.

```
emulator64_x86_64_arm64:/data/data # ls
android                                  com.android.sharedstoragebackup
android.auto_generated_rro_product__     com.android.shell
android.auto_generated_rro_vendor__      com.android.simappdialog
cc.pacer.androidapp                      com.android.simappdialog.auto_generated_rro_product__
```

Figura 11 - Aceder a pasta principal do android

Com a aplicação identificada, saímos do modo administrador e utilizámos o comando pull (serve para transferir ficheiros ou pastas do dispositivo android para o computador) para transferir os ficheiros da aplicação do dispositivo Android para o computador, permitindo iniciar o processo de análise.

```
PS C:\Android\Sdk\platform-tools> .\adb pull /data/data/cc.pacer.androidapp
```

Figura 12 - Extração dos ficheiros da aplicação

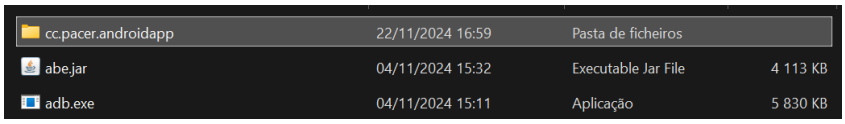

Figura 13 - Aplicação extraída para o computador

Na etapa seguinte, foi realizada a extração do arquivo APK da aplicação.

O processo começou com a execução do código mostrado na próxima imagem, utilizado para identificar o *package* da aplicação.

```
PS C:\Android\Sdk\platform-tools> .\adb shell pm list packages
```

```
package:cc.pacer.androidapp
```

Figura 14 - Encontrar o package da aplicação

Com o package identificado, localizámos o caminho completo do APK correspondente.

```
PS C:\Android\Sdk\platform-tools> .\adb shell pm path cc.pacer.androidapp
package:/data/app/~~0wIwASgCrT_7EfkDMppe9Q==/cc.pacer.androidapp-aF1ICDJe8KC2-BCLyIgs7Q==/base.apk
```

Figura 15 - Localização do apk

Finalmente, procedemos à extração do APK utilizando o comando ilustrado na última imagem.

```
PS C:\Android\Sdk\platform-tools> .\adb pull /data/app/~~0wIwASgCrT_7EfkDMppe9Q==/cc.pacer.androidapp-aF1ICDJe8KC2-BCLyIgs7Q==/base.apk
/data/app/~~0wIwASgCrT_7EfkDMppe9Q==/cc.pacer.androidapp-aF1ICDJe8KC2-BC...base.apk: 1 file pulled, 0 skipped. 24.3 MB/s (68397475 bytes in 2.685s)
```

Figura 16 - Extração do apk

3. Análise das permissões

Após a extração do ficheiro base.apk, podemos avançar para a sua análise.

Primeiramente, é necessário abrir o ficheiro base.apk da aplicação.

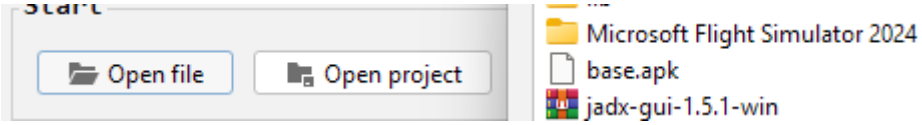

Figura 17 - Interface do JADX com o ficheiro base.apk por abrir

Após abrir o ficheiro APK no JADX, é possível visualizar os ficheiros contidos na aplicação.

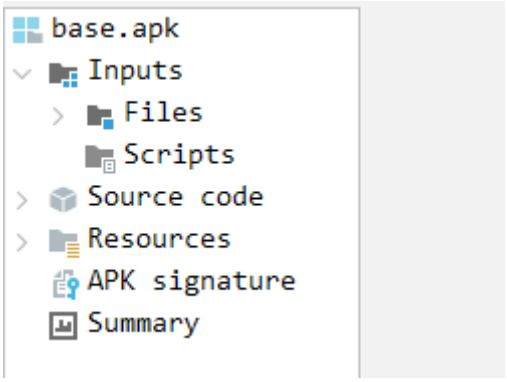

Figura 18 - Visualização da estrutura de ficheiros do APK na ferramenta JADX

Durante a análise do APK, identificámos, na pasta resources, o ficheiro AndroidManifest.*xml*, que contém as permissões utilizadas pela aplicação.

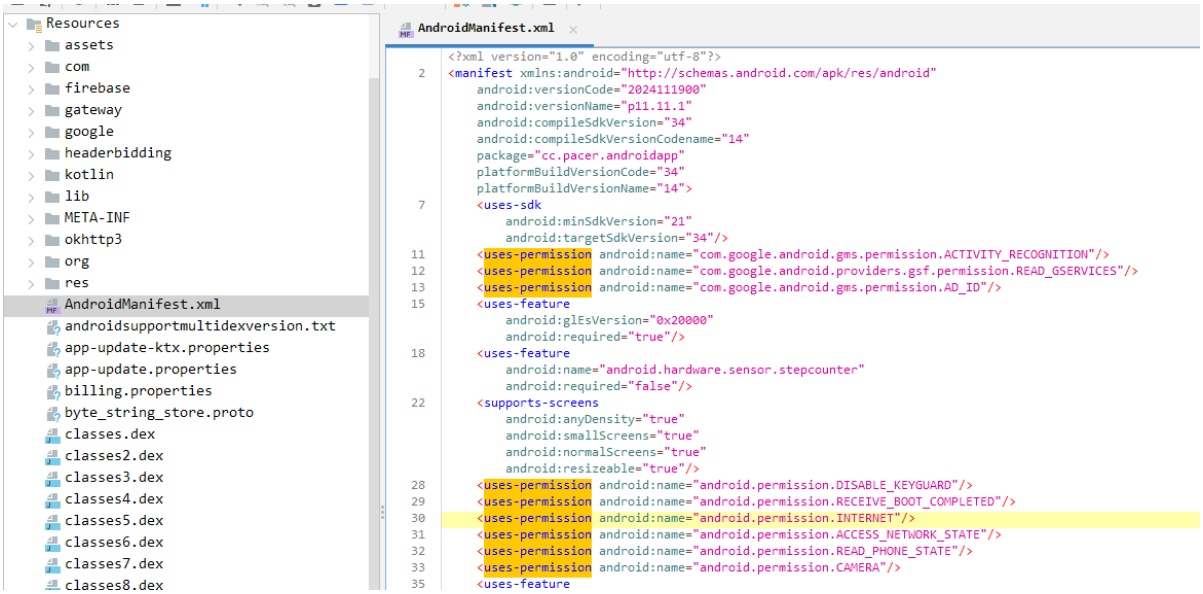

Figura 19 - Exibição do ficheiro AndroidManifest.xml e suas permissões

Na figura a baixo, estão elencadas todas as permissões da aplicação e a sua descrição.

Permissão	Descrição
com.google.android.gms.permission.ACTIVITY_RECOGNITION	Permissão para reconhecer atividades do utilizador.
com.google.android.providers.gsf.permission.READ_GSERVICES	Permissão para ler configurações do Google Services Framework.
com.google.android.gms.permission.AD_ID	Permissão para acessar identificadores de anúncios.
android.permission.DISABLE_KEYGUARD	Permissão para desativar o bloqueio de tela.
android.permission.RECEIVE_BOOT_COMPLETED	Permissão para receber eventos de inicialização do sistema.
android.permission.INTERNET	Permissão para acessar a Internet.
android.permission.ACCESS_NETWORK_STATE	Permissão para verificar o estado da rede.
android.permission.READ_PHONE_STATE	Permissão para ler o estado do telefone.
android.permission.CAMERA	Permissão para acessar a câmara.
android.permission.READ_MEDIA_IMAGES	Permissão para ler imagens de mídia.
android.permission.READ_MEDIA_VIDEO	Permissão para ler vídeos de mídia.
android.permission.READ_MEDIA_AUDIO	Permissão para ler áudio de mídia.
android.permission.WRITE_EXTERNAL_STORAGE	Permissão para gravar no armazenamento externo.
android.permission.WAKE_LOCK	Permissão para manter o processador ativo.
android.permission.ACCESS_FINE_LOCATION	Permissão para acessar a localização precisa.
android.permission.ACCESS_COARSE_LOCATION	Permissão para acessar a localização aproximada.
android.permission.ACCESS_WIFI_STATE	Permissão para verificar o estado do Wi-Fi.
android.permission.READ_CONTACTS	Permissão para ler contatos do utilizador.
android.permission.SYSTEM_ALERT_WINDOW	Permissão para sobrepor janelas ao sistema.
android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS	Permissão para ignorar otimizações de bateria.
android.permission.FOREGROUND_SERVICE	Permissão para usar serviços em primeiro plano.
android.permission.POST_NOTIFICATIONS	Permissão para enviar notificações.
android.permission.FOREGROUND_SERVICE_LOCATION	Permissão para serviços em primeiro plano relacionados à localização.
android.permission.FOREGROUND_SERVICE_HEALTH	Permissão para serviços em primeiro plano relacionados à saúde.
android.permission.READ_MEDIA_VISUAL_USER_SELECTED	Permissão para ler mídia selecionada pelo utilizador.
com.google.android.c2dm.permission.RECEIVE	Permissão para receber mensagens do Google Cloud Messaging.
android.permission.ACCESS_ADSERVICES_ATTRIBUTION	Permissão para acessar atribuição de serviços de anúncios.
android.permission.ACCESS_ADSERVICES_AD_ID	Permissão para acessar identificadores de serviços de anúncios.
com.android.vending.BILLING	Permissão para realizar transações de faturamento.
com.applovin.array.apphub.permission.BIND_APPHUB_SERVICE	Permissão para vincular o serviço AppHub da AppLovin.
com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE	Permissão para vincular o serviço de referência de instalação.
android.permission.ACCESS_ADSERVICES_TOPICS	Permissão para acessar tópicos de serviços de anúncios.
cc.pacer.androidapp.DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION	Permissão personalizada para receptores dinâmicos não exportados.
com.samsung.android.mapsagent.permission.READ_APP_INFO	Permissão para ler informações de aplicativos (Samsung).
com.huawei.appmarket.service.commondata.permission.GET_COMMON_DATA	Permissão para acessar dados comuns do Huawei AppGallery.

Figura 20 - Tabela de permissões da app

Destas permissões, podemos destacar as seguintes:

- Permissões essenciais da aplicação

  android.permission.INTERNET

  Esta permissão é necessária caso a aplicação precisa de se conectar à internet para transferir dados, como aceder a APIs ou carregar conteúdos online (ex: Anúncios).

  android.permission.ACCESS_NETWORK_STATE

  A permissão é útil para verificar se o dispositivo tem conexão com a internet.

  android.permission.FOREGROUND_SERVICE

  É necessária para executar serviços em primeiro plano.

- Permissões Relacionadas à Localização

  android.permission.ACCESS_FINE_LOCATION

  É necessária para obter a localização precisa do utilizador.

  android.permission.ACCESS_COARSE_LOCATION

  Permite o acesso à localização aproximada, geralmente suficiente para apps que não precisam de alta precisão.

- Permissões de Notificações

  android.permission.POST_NOTIFICATIONS

O requisito para enviar notificações nesta aplicação é necessário ter o Android 13 ou superior.

- Permissões Específicas de Funcionalidade

  android.permission.READ_PHONE_STATE

  É necessário para verificar o estado do telefone (como ID de dispositivo ou estado de ligação).

  com.android.vending.BILLING

  É essencial para implementar compras no aplicativo.

4. Análise à aplicação Pacer Health

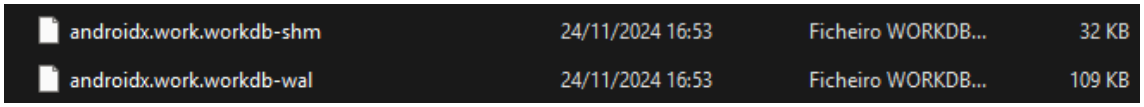

Figura 21 - Ficheiros db-wal e db-shm

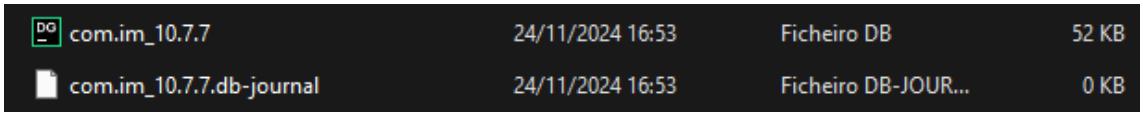

Figura 22 - Ficheiro db-journal

Nas pastas Data Bases e no_backup, encontramos bases de dados e ficheiros da aplicação. Podemos dizer que a aplicação utiliza dois formatos de gravação das bases de dados, nas quais são WAL e SHM e também vendo um ficheiro journal.

O formato SHM é um ficheiro temporário criado pelo SQLite, utilizado pela base de dados para armazenamento de memória partilhada, criado e gerido automaticamente pelo SQLite também se tratando de um recurso utilizado em sistemas informáticos para permitir que múltiplos processos acedam à mesma região de memória, promovendo uma comunicação eficiente entre eles.

O ficheiro .wal regista as alterações antes de serem aplicadas ao ficheiro principal, garantindo eficiência, integridade dos dados e suporte a múltiplos acessos simultâneos.

O formato journal (journal file ou ficheiro de registo) é um mecanismo tradicional usado pelo SQLite para garantir a integridade dos dados em caso de falha durante uma transação.

4.1. Análise às bases de dados

4.1.1. Pasta: cc.pacer.androidapp > databases > com.im_10.7.7.db

Tabela config

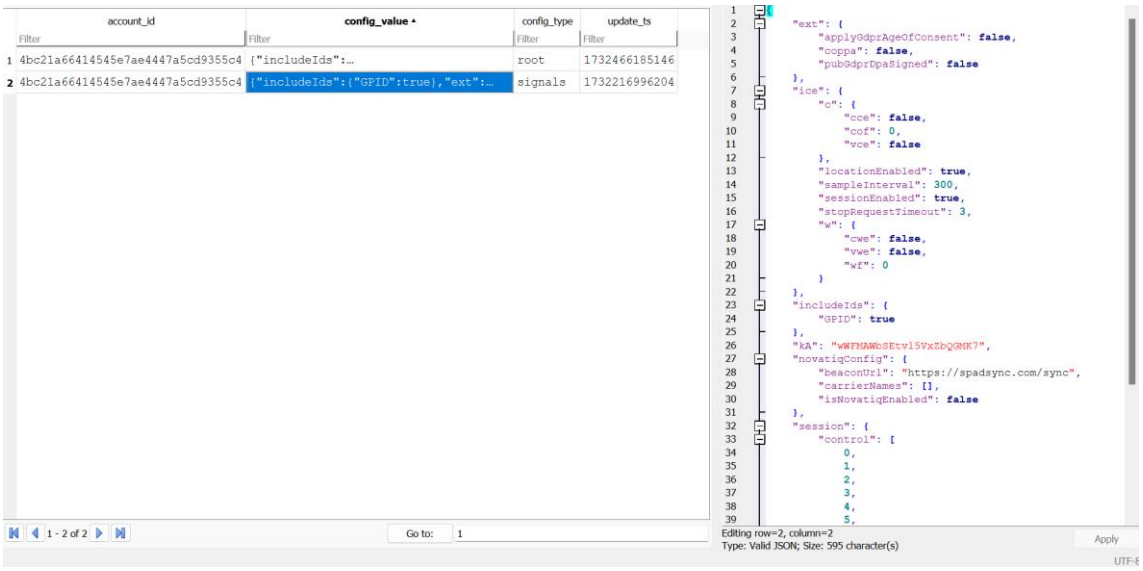

Figura 23 - Tabela config

Esta tabela contém configurações da aplicação, incluindo endpoints (URL's) de comunicação externa e identificadores que podem ser utilizados para rastrear utilizadores. Algumas configurações sugerem atenção a privacidade (coppa, gdpr)*.

As mudanças são rastreadas por timestamps, mas nenhuma alteração significativa foi identificada no estado atual.

No campo config_value temos dados json que nos indicam a estruturação de configurações de contas. Neste caso havendo duas root e signals, sendo que a root será de uma conta que define a base do funcionamento do sistema e a signals que serve para a monitorização da reação a eventos e outros comportamentos.

- COPPA – lei dos EUA que protege a privacidade de menores de 13 anos de idade.
- GDPR – General Data Protection Regulation.

Tabela telemetry

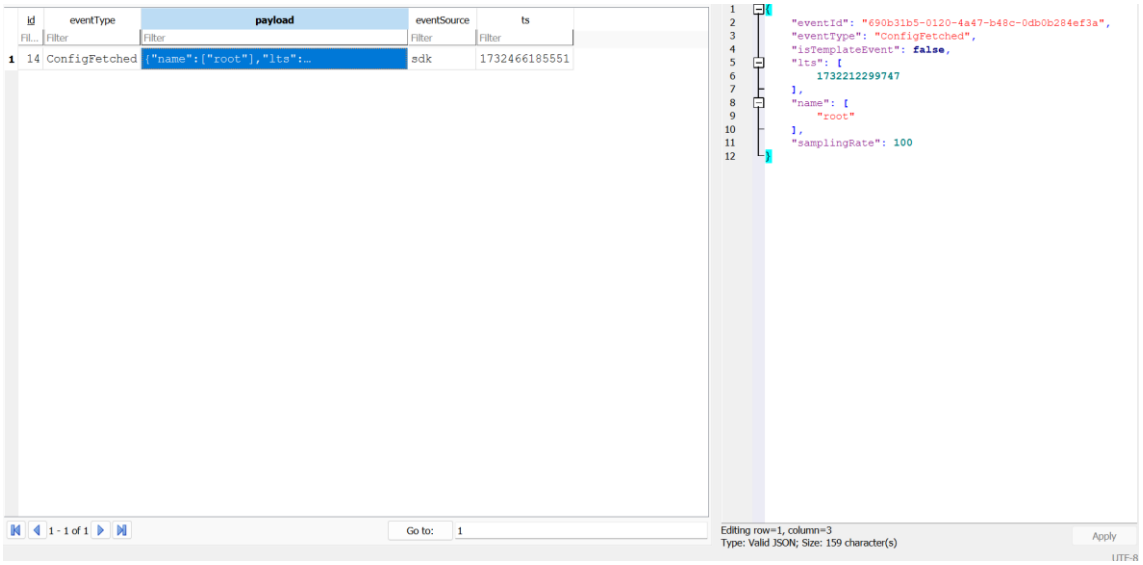

Figura 24 - Tabela telemetry

Esta tabela regista eventos do tipo sessionStarted, que indicam o início de sessões de utilizadores. Cada evento tem um identificador único (eventId) e é monitorizado com uma taxa de amostragem 100%, sugerindo monitorização constante.

Também vendo que no payload temos uns dados adicionais em formato json que indica que a configuração deste tipo de evento que é em modo root.

4.1.2. Cc.pacer.androidapp > databases > MDData.db

Tabela dailyActivityLog

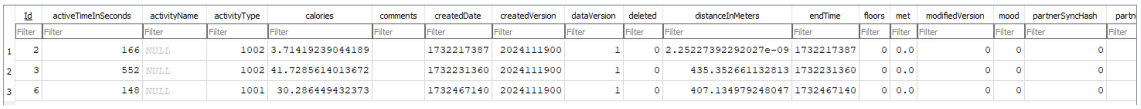

Figura 25 - Primeira parte da tabela dailyActivityLog

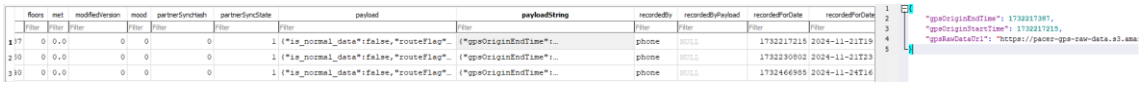

Figura 26 - Segunda parte da tabela dailyActivityLog

A tabela regista atividades físicas detalhadas, como passos, tempo ativo e calorias, associadas a um utilizador único através de identificadores (user_id).

Os dados indicam monitorização constante e sincronização com servidores. No entanto, alguns registos estão incompletos, sugerindo possíveis falhas de rastreamento.

Fazendo uma análise do campo payloadString, em conjunto com os dados obtidos nos campos json, podemos ver que:

- gpsOriginEndTime: pela informação que aparece no json indica o fim do rastreamento GPS da atividade que corresponde a 29 de novembro de 2024, 06:49:47 UTC

- gpsOriginSartTime: pela informação que aparece no json indica o início do rastreamento GPS da atividade que corresponde a 29 de novembro de 2024, 06:46:55 UTC
- gpsRawDataUrl: no caso do RawDataUrl corresponde ao rastreamento do waypoints do usuário na atividade.

Tabela trackpoints

	accuracy	activity_type	altitude	bearing	id	latitude	longitude	path_id	speed	steps	time	trackId
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	5.0		0.0	0.0	10	37.4219983	-122.084	3	0.00692041078582406	0	1732217220054	3
2	5.0		0.0	0.0	11	37.4219983	-122.084	3	2.58537163899675e-10	0	1732217225056	3
3	5.0		0.0	0.0	12	37.4219983	-122.084	3	1.21221381457065e-12	0	1732217229137	3
4	5.0		0.0	0.0	13	37.4219983	-122.084	3	5.56218372616403e-15	0	1732217233180	3
5	5.00099992752075		0.0	0.0	14	37.4219983	-122.084	3	2.78722126314923e-19	0	1732217238100	3
6	5.00500011444092		0.0	0.0	15	37.4219983	-122.084	3	1.68245611181505e-24	0	1732217243075	3
7	5.00400018692017		0.0	0.0	16	37.4219983	-122.084	3	1.70684563310019e-30	0	1732217248067	3
8	5.01399993896484		0.0	0.0	17	37.4219983	-122.084	3	1.74682951861926e-36	0	1732217252072	3
9	5.01200008392334		0.0	0.0	18	37.4219983	-122.084	3	1.78105034815684e-42	0	1732217256074	3
10	5.01700019836426		0.0	0.0	19	37.4219983	-122.084	3	0.0	0	1732217260078	3
11	5.01800012588501		0.0	0.0	20	37.4219983	-122.084	3	0.0	0	1732217265085	3
12	5.02099990844727		0.0	0.0	21	37.4219983	-122.084	3	0.0	0	1732217270097	3
13	5.02500009536743		0.0	0.0	22	37.4219983	-122.084	3	0.0	0	1732217275087	3
14	5.0310001373291		0.0	0.0	23	37.4219983	-122.084	3	0.0	0	1732217279089	3
15	5.0		0.0	0.0	24	37.4219983	-122.084	3	0.0	0	1732217283114	3
16	5.03900003433228		0.0	0.0	25	37.4219983	-122.084	3	0.0	0	1732217287122	3
17	5.0		0.0	0.0	26	37.4219983	-122.084	3	0.0	0	1732217292099	3
18	5.0		0.0	0.0	27	37.4219983	-122.084	3	0.0	0	1732217296102	3
19	5.0		0.0	0.0	28	37.4219983	-122.084	3	0.0	0	1732217300108	3
20	5.0		0.0	0.0	29	37.4219983	-122.084	3	0.0	0	1732217304110	3
21	5.0		0.0	0.0	30	37.4219983	-122.084	3	0.0	0	1732217309182	3

Figura 27 - Tabela trackPoints

A tabela regista dados de localização geográfica e movimento dos utilizadores, incluindo, latitude, longitude, altitude e velocidade. Esta tabela é altamente sensível, pois pode ser usada para rastrear atividades e rotinas dos utilizadores, levantando preocupações significativas de privacidade.

Tabela tracks.db

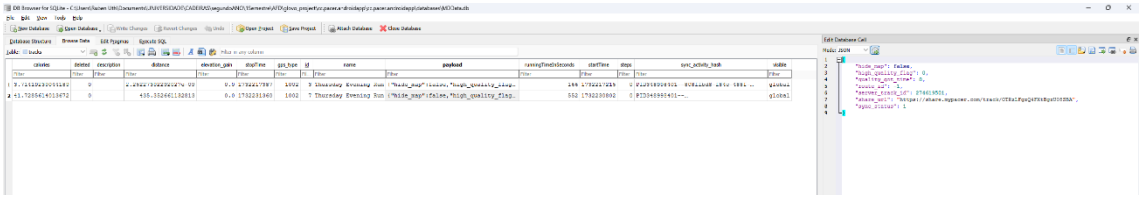

Figura 28 - Tabela tracks

A tabela regista trajetos realizados pelos utilizadores, incluindo detalhes como calorias queimadas, distância e passos. Cada trajeto pode ser identificado por um URL público (share_url), levantando preocupações de privacidade. A sincronização com servidores externos é indicada por campos como server_track_id e async_status, sugerindo que os dados podem ser acessíveis por terceiros.

4.2. Análise às alterações nos dados

4.2.1. Antes da Corrida Ter Sido Iniciada

Na primeira imagem, a **linha 3** (atividade track_1732466985) aparece com valores incompletos:

- **calories**: 0.0
- **distance**: 0.0
- **steps**: 0
- **payload**: NULL

Estes valores sugerem que a atividade foi criada, mas ainda não tinha dados associados porque a corrida ainda estava em andamento ou apenas tinha sido iniciada.

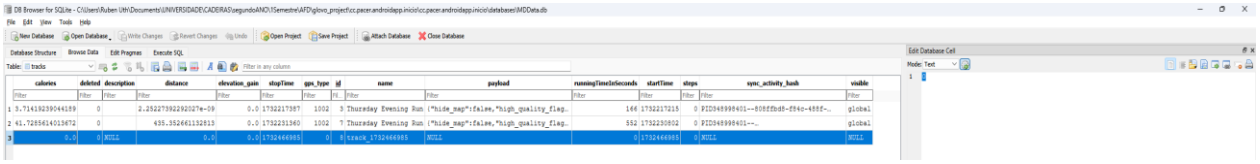

Figura 29 - Tabela tracks antes da corrida

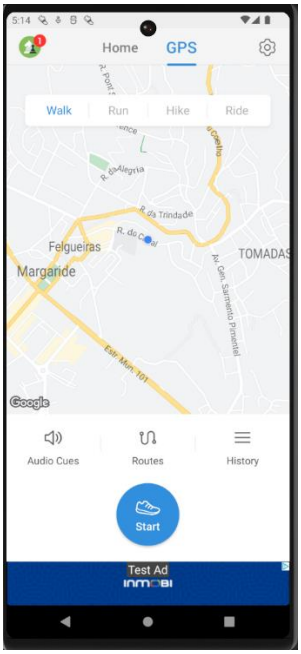

Figura 30 - Ecrã de começo da atividade

Ao pressionar o botão START para iniciar a corrida, a aplicação começa a exibir movimento, e os campos da base de dados, anteriormente preenchidos com zero ou NULL, passam a ser atualizados com os dados correspondentes à corrida.

4.2.2. Depois da Corrida Ter Sido Terminada

Na segunda imagem, a mesma **linha 3** (track_1732466985) foi atualizada:

- **calories**: 30.28

- **distance**: 407.13
- **steps**: 0 (pode indicar um bug ou que passos não foram registados para esta atividade específica).
- **payload**: Inclui informações como hide_map e high_quality_flag, indicando que os detalhes da corrida foram processados e armazenados.

Relativamente à tabela tracks, verificamos que o nome da atividade (track_1732466985) manteve-se genérico, sugerindo que não foi personalizada.

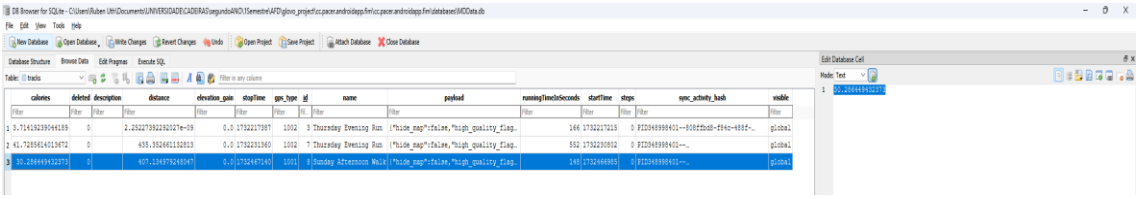

Figura 31 - Tabela tracks depois da corrida

Já quanto à tabela dailyActivityLog, verificamos que foram adicionados novos registos e foram atualizados os seguintes campos para refletir a atividade realizada:

- DistanceInMeters: a distância percorrida foi calculada e também registada.
- RecordedBy: indica que os dados foram registados por "phone", o que confirma que o dispositivo utilizado foi um smartphone.
- Payload: contêm detalhes adicionais sobre a corrida (como origem e destino ou configuração GPS usada).

	payload	payloadString	recordedBy	recordedByPayload	recordedForDate	recordedForDatetimeIso8601	recordedForDay
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	{"is_normal_data":false,"routeFlag":…	{"gpsOriginEndTime":…	phone	NULL	1732217215	2024-11-21T19:29:47.673Z	20241121
2	{"is_normal_data":false,"routeFlag":…	{"gpsOriginEndTime":…	phone	NULL	1732230802	2024-11-21T23:22:40.636Z	20241121
3	{"is_normal_data":false,"routeFlag":…	{"gpsOriginEndTime":…	phone	NULL	1732466985	2024-11-24T16:52:20.299Z	20241124

Figura 32 - Tabela dailyActivityLog depois da corrida

Assim, verificou-se o comportamento esperado para uma tabela comportamental como a dailyActivityLog. Sempre que uma atividade é realizada na aplicação, é automaticamente gerado um registo na base de dados, no qual são detalhados aspetos específicos dessa atividade, como a data e a distância percorrida. Esta estrutura demonstra que a aplicação é capaz de armazenar registos contínuos das atividades, permitindo assim a criação de um histórico comportamental.

Este comportamento é particularmente relevante porque confirma a ligação direta entre as ações realizadas na aplicação e os dados armazenados. Por exemplo, ao criar uma corrida, é gerado automaticamente um registo detalhado na base de dados, o que assegura que cada ação é documentada de forma precisa. Além disso, o nível de detalhe registado, como a distância percorrida ou a origem dos dados, ilustra claramente como a aplicação monitoriza as atividades dos utilizadores.

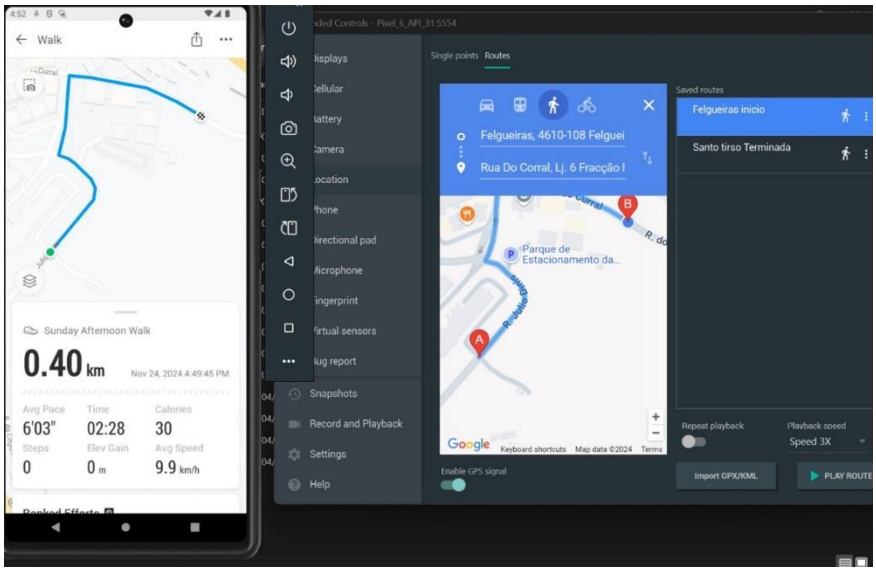

Figura 33 - Screenshot dos dados da corrida

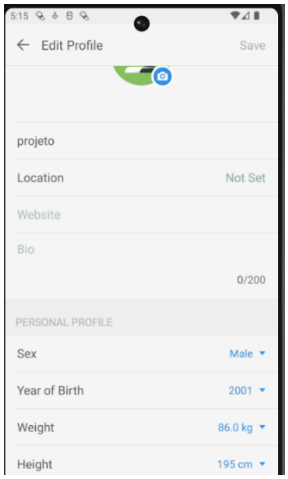

Figura 34 - Ecrã para editar o perfil

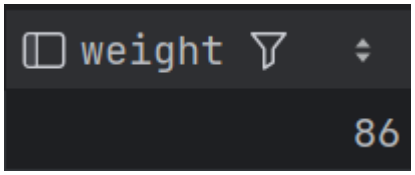

Figura 35 - Tabela heightLog

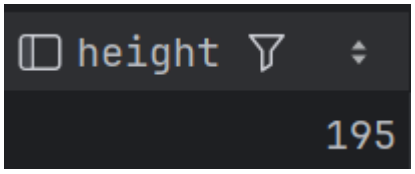

Figura 36 - Tabela weightLog

Após a análise das duas imagens acima, concluímos que, ao inserir o valor do campo height na aplicação, este é devidamente atualizado na base de dados, sendo registado na tabela heightLog. Da mesma forma, verificamos que, ao inserir o valor do campo weight, este é corretamente registado na tabela weightLog.

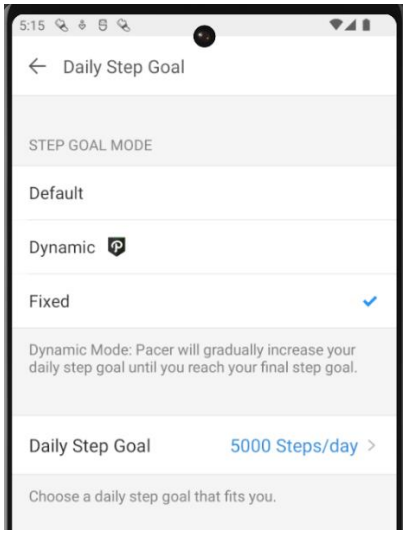

Figura 37 - Daily step goal

Nesta seção da aplicação no telemóvel, podemos observar que, após alterar a meta de passos diários inicialmente definida como default (10.000 passos) para o modo manual, ajustámos o valor para 5.000 passos. Esta alteração reflete-se corretamente na tabela dailyGoal.

default	10000
manual	5000

Figura 38 - Tabela dailyGoal

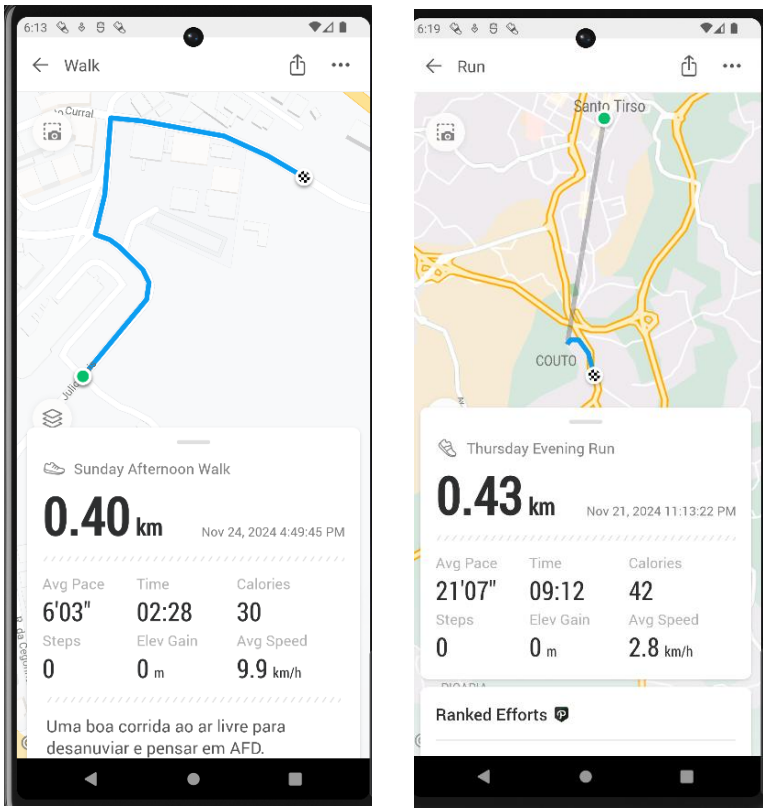

Figura 39 - Exercícios Walk e Run

	calories	deleted	description	distance
1	41.72856140136719	0		435.3526611328125
2	30.286449432373047	0	Uma boa corrida ao ar livre para desan…	407.1349792480469

Figura 40 - Tabela sem eliminar o Walk

Após a análise das imagens acima, verificamos que, de acordo com os dados inseridos na aplicação após o término da corrida, os valores na tabela *tracks* são registados corretamente. Além disso, caso a tabela seja eliminada, a situação é apresentada conforme mostrado na imagem abaixo.

	calories	deleted	description	distance
1	41.72856140136719	0		435.3526611328125

Figura 41 - Tabela Depois de eliminar o Walk

4.2.3. Comportamento da Aplicação

Inicialmente, a atividade é criada como um registo placeholder, com valores padrões (zeros ou NULL). Após o término da corrida, a aplicação processa e atualiza os campos com dados reais, como calorias, distância e outros detalhes no payload.

Este comportamento sugere um fluxo sequencial de registo e processamento de dados.

4.2.4. Dados Atualizados Após o Processamento

O registo reflete métricas da corrida (calorias e distância), enquanto outros campos, como steps, podem não ser atualizados dependendo de como a aplicação lida com diferentes tipos de atividades.

4.2.5. Impacto na Privacidade

Os dados processados, como os valores no payload e o identificador único da atividade, indicam que as informações são armazenadas de forma estruturada e potencialmente sincronizáveis com servidores externos, dependendo do estado do campo sync_activity_hash.

4.2.6. Diferenciação de Estados

A comparação entre os estados "antes" e "depois" do término da corrida evidencia como a aplicação lida com dados em tempo real, registando placeholder e atualizando os valores assim que a atividade é concluída.

4.2.7. Ofuscação das pastas no Jadx-gui

Durante a compilação, o Android Studio pode usar ferramentas como o ProGuard (ou o mais moderno R8) para reduzir o tamanho do APK e ofuscar os nomes das classes, métodos, variáveis e pacotes como por exemplo, um pacote chamado com.exemplo.aplicacao pode ser transformado em algo como ac ou df para que seja difícil do outro lado perceberem em que ficheiro estão a mexer, isto claro em níveis de segurança.

A ofuscação do código dificulta a identificação lógica do funcionamento interno da aplicação, protegendo a lógica do negócio e assim evitando os ataques maliciosos as aplicações, pois sem a ofuscação, alguém com ferramentas como o *jadx-gui* ou *APKTool* pode facilmente entender e explorar o código para atividades como injeção de código malicioso dentro da aplicação tirando proveito das falhas de segurança.

5. Desafios

Inicialmente, o projeto era sobre a análise das bases de dados da aplicação Glovo e sobre o comportamento das mesmas após ações efetuadas na app.

No entanto, durante a realização do mesmo, fomos desafiados por algumas barreiras significativas que, por consequência dificultou o progresso do projeto.

As bases de dados principais da Glovo, como glovo.db e runtime-carts.db, estavam encriptadas, tornando impossível o acesso aos dados sem as chaves de encriptação.

A aplicação apresentava também dependências do Google Play, o que limitava a capacidade de replicar o comportamento num emulador sem Google Play. Criaram-se diferentes emuladores para testar o ambiente (com e sem Google Play), mas os resultados permaneceram limitados e sem nenhum aproveitamento dos mesmos.

Mesmo com permissões de root, algumas áreas da aplicação permaneciam inacessíveis devido à forma como os dados estavam armazenados.

Após vários dias de tentativas e várias maneiras de tentar obter algo útil para análise, o tempo disponível para o projeto levou à decisão de mudar o foco para outra aplicação mais acessível, a Pacer Health. A mudança foi importante, principalmente no ponto de cumprimento dentro do prazo estabelecido de realização do projeto.

Os desafios iniciais com a Glovo ajudaram a definir estratégias mais eficientes para a análise da Pacer Health, como um melhor entendimento das ferramentas necessárias e a configuração adequada do ambiente de trabalho. Essas aprendizagens demonstram a importância de se adaptar a barreiras técnicas e de redirecionar esforços de forma estratégica.

6. Conclusão

6.1. Resumo

A análise forense realizada sobre a aplicação Pacer Health permitiu compreender, em detalhe, o comportamento da aplicação em termos de registo de dados do utilizador. Foi observado que a aplicação monitoriza constantemente as atividades, armazenando informações como passos, distância percorrida, calorias queimadas, localização e permissões solicitadas. Através do cruzamento dessas informações com os registos da base de dados, foi possível confirmar a correta associação entre as ações realizadas pelo utilizador e os dados registados, destacando ainda algumas vulnerabilidades, como possíveis impactos na privacidade dos utilizadores devido à sincronização com servidores externos.

Esta análise evidenciou também como as tabelas de base de dados refletem mudanças em tempo real e processam informações, demonstrando o fluxo funcional da aplicação desde o registo inicial de atividades até à sua conclusão.

6.2. Considerações finais

A investigação sublinhou a relevância de análises forenses regulares em aplicações móveis para garantir conformidade com regulamentos de privacidade, como o RGPD, e identificar potenciais riscos à segurança e privacidade dos utilizadores. Além disso, a análise detalhada das permissões requisitadas pela aplicação revelou a necessidade de avaliar criticamente os acessos concedidos para limitar potenciais vulnerabilidades.