



## **[PROTOCOLO EXPERIMENTAL : LEMBRETES ANDROID COM BACKEND (ALARMANAGER + PENDINGINTENT)]**

### **[GRUPO 10]**

#### **INFORMAÇÕES GERAIS**

- **Unidade Curricular:** Programação de Dispositivos Móveis
- **Tema:** Agendamento fiável de notificações com componentes de backend Android
- **Objetivo:** Demonstrar, testar e validar que os lembretes surgem na hora exata, mesmo com a app fechada.
- **Realizado por:**
  - Beatriz Ventura Patrício , 52100
  - Maria Benedita Dias Rodrigues , 52008
  - Maria Beatriz Rodrigues Machado, 52123
  - Fábio André Bidarra Horta , 51817
  - Daniel Lopes Almeida , 51893

# DESCRIÇÃO DA APLICAÇÃO

A aplicação desenvolvida é uma **Mini-App de Lembretes de Visitas**, cujo principal objetivo é **emitir notificações automáticas no momento agendado**, mesmo que o utilizador tenha fechado a aplicação.

- **MainActivity** – Interface de agendamento.
- **VisitSchedulerService** – Registo de alarmes no Android.
- **VisitReminderReceiver** – Execução e visualização da notificação.
- **MyApp (Application)** – Configuração de ambiente e canal de notificações.

## FUNCIONALIDADES DA MINI-APP

- Agendamento de lembretes com hora e data definidas.
- Execução de notificações automáticas.
- Garantia de funcionamento mesmo com a app encerrada.
- Criação de canal de notificações “visitas” com prioridade alta.
- Testes de robustez (Doze mode, multitarefa, ecrã desligado, reinício do dispositivo).

## PROTOCOLO DE TESTE, REALIZADO PASSO A PASSO

### 1. INSTALAÇÃO DO AMBIENTE

#### 1.1 Configuração Inicial do sistema

1. Instalar a aplicação no dispositivo Android;
2. Conceder permissões necessárias (ex.: `SCHEDULE_EXACT_ALARM`);
3. Confirmar a criação do canal de notificação “**visitas**”;
4. Verificar que a aplicação inicia corretamente.

#### 1.2 Estrutura do Projeto

pt.ubi.di.pdm.lembretes/

└─ MainActivity.java

└─ VisitSchedulerService.java

└─ VisitReminderReceiver.java

└─ MyApp.java

└─ res/layout/activity\_main.xml

└─ res/drawable/ic\_notification.xml

## TESTE DA INTERFACE (MAINACTIVITY)

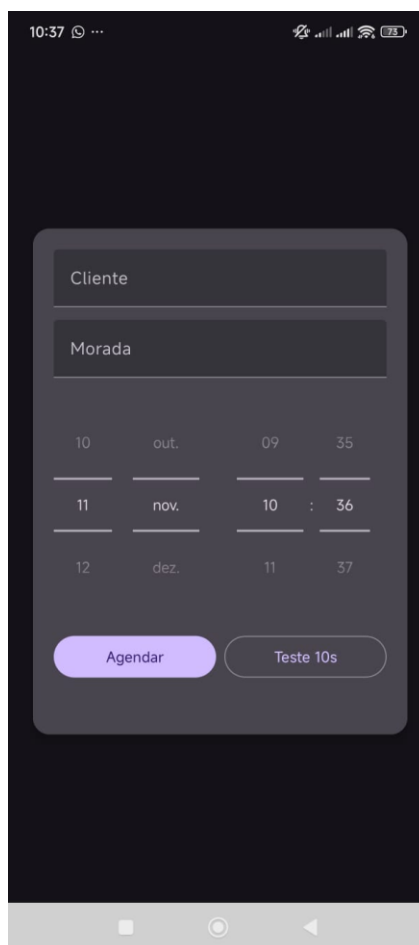
**Objetivo:** Validar que o utilizador consegue agendar um lembrete.

**Passos da execução:**

1. Abrir a aplicação.
2. Inserir data e hora.
3. Carregar em “Agendar”.

**Resultado Esperado:**

Mensagem de confirmação ou log indicando o envio do pedido para o VisitSchedulerService.



# TESTE DE AGENDAMENTO (VISITSCHEDULERSERVICE)

**Objetivo:** Garantir que o serviço agenda corretamente o lembrete.

**Passos da execução:**

1. Verificar logs da aplicação.
2. Confirmar que o método `setExactAndAllowWhileIdle()` é invocado.

**Critérios de sucesso:**

- AlarmManager regista alarme exato (RTC\_WAKEUP).
- PendingIntent criada com FLAG\_IMMUTABLE.
- Serviço termina após o agendamento (sem processos em fundo).

Cliente  
beatriz

---

Morada  
rua da liberdade

---

Oct	10	15	53
Nov	11	16	: 54
Dec	12	17	55

AgendarTeste 10s

Visita agendada para 11/11/2025 16:54

# TESTE DE EXECUÇÃO (VISITREMINDERRECEIVER)

**Objetivo:** Confirmar que a notificação é mostrada no momento certo.

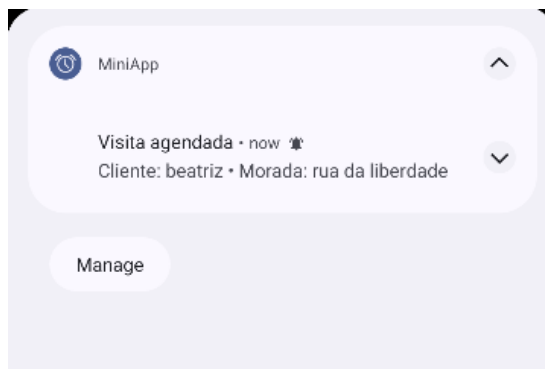
**Passos da execução:**

1. Agendar notificação para 10 segundos.
2. Minimizar ou fechar a app.
3. Aguardar disparo da notificação.

**Resultado Esperado:**

Notificação com:

- Título: "Lembrete de visita"
- Texto: "Está na hora da visita agendada!"
- Ícone visível
- AutoCancel ativo



## TESTES DE CENÁRIOS DE FALHA

Cenário	Ação	Resultado Esperado
App fechada	Fechar app após agendar	Notificação aparece normalmente
Ecrã desligado	Bloquear ecrã	Notificação é mostrada
Doze Mode	Emulador em modo de poupança	Notificação sem atraso
Canal inexistente	Apagar canal manualmente	App recria canal automaticamente
Reinício do telefone	Reiniciar dispositivo	Reagendamento futuro através de BOOT_COMPLETED (planeado)

# VALIDAÇÃO FINAL

## Funcionalidades Principais:

- ✓ App inicia sem erros e carrega corretamente.
- ✓ Ecrã principal funcional, permitindo definir data e hora para o lembrete.
- ✓ Agendamento de notificações realizado com sucesso através do VisitSchedulerService.
- ✓ Execução automática da notificação confirmada através do VisitReminderReceiver.
- ✓ Canal de notificações "visitas" criado e configurado na classe MyApp.
- ✓ Notificação exibida mesmo com a aplicação encerrada.
- ✓ Testes de Doze Mode e multitarefa concluídos com sucesso, sem atrasos ou falhas.
- ✓ Cancelamento e reagendamento possíveis através da mesma PendingIntent.

## EXPERIÊNCIA DO UTILIZADOR:

- ✓ Interface simples, funcional e intuitiva.
- ✓ Campos de data/hora e botão "Agendar" com feedback visual adequado.
- ✓ Notificação apresentada com ícone, som e destaque no ecrã de bloqueio.
- ✓ Interação fluida entre o ecrã de agendamento e o sistema de notificações.
- ✓ Feedback imediato ao utilizador após o agendamento do lembrete.
- ✓ Aplicação responsiva e estável durante toda a utilização.

## INTEGRAÇÃO TÉCNICA:

- ✓ Permissão SCHEDULE\_EXACT\_ALARM corretamente configurada no AndroidManifest.
- ✓ Dependências e classes Android (AlarmManager, PendingIntent, NotificationCompat) integradas sem conflitos.
- ✓ Canal de notificações criado com IMPORTANCE\_HIGH, garantindo prioridade e visibilidade.
- ✓ Utilização do método setExactAndAllowWhileIdle() para precisão máxima.
- ✓ PendingIntent configurada com FLAG\_IMMUTABLE para segurança e compatibilidade com Android 12+.
- ✓ Ausência de crashes ou exceções durante a execução normal e nos testes realizados.
- ✓ Arquitetura modular e escalável, com clara separação entre interface e backend.

# CONCLUSÕES E RESULTADOS

## RESULTADOS OBTIDOS:

- A aplicação demonstrou **arquitetura backend robusta**, separando claramente interface e lógica de sistema.
- O uso de **AlarmManager + PendingIntent** garantiu precisão e fiabilidade no agendamento.
- O canal de notificações criado em **MyApp** assegurou compatibilidade com versões recentes do Android.
- Os testes confirmaram que a **notificação dispara com a app fechada**, validando a correta integração entre componentes de backend.

## VALOR EDUCACIONAL:

O projeto demonstra boas práticas de Android:

- Separação de responsabilidades (UI vs backend).
- Uso de componentes do sistema operativo.
- Consideração por gestão de energia e permissões.
- Preparação para escalabilidade (base de dados, BOOT\_COMPLETED, múltiplos canais).

## DEMONSTRAÇÃO PRÁTICA:

Cenário	Ação Executada	Resultado Observado
Teste rápido (10 segundos)	Agendar notificação para 10s e minimizar a app	A notificação foi exibida exatamente no tempo previsto ("Está na hora da visita agendada!").
Ecrã desligado / Doze Mode	Bloquear o dispositivo após agendar	O alarme disparou normalmente, confirmando o funcionamento de AllowWhileIdle.
Fecho da app / multitarefa	Encerrar completamente a app após agendar	O lembrete foi mostrado mesmo com a app encerrada — validação de backend independente da UI.