

## Ir Além 2 - Automação Inteligente com RPA, IA e Dados Híbridos

### Objetivo

Simular um robô (RPA) que monitora periodicamente dados clínicos estruturados em um banco relacional, detecta anomalias com regras simples (IA “leve”) e registra logs/alertas de forma rastreável em um armazenamento não relacional.

### Implementação

#### Código:

Relacional (SQLite): ``automation/database_setup.py``

Robô de monitoramento: ``automation/rpa_monitor.py``

#### Armazenamento:

Banco relacional: ``automation/data/patients.db`` (SQLite)

Banco não relacional (logs): ``automation/data/logs.json`` (JSON)

#### Fluxo do Robô (RPA)

##### 1. **\*\*Inicialização do banco\*\***

``database_setup.py`` cria tabelas ``patients`` e ``monitoring`` e popula com dados fictícios.

##### 2. **\*\*Leitura periódica\*\***

``rpa_monitor.py`` lê as últimas medições do banco relacional.

##### 3. **\*\*Detecção de anomalias\*\***

Regra: PA > 140/90 ou FC > 100 bpm.

##### 4. **\*\*IA Generativa (opcional)\*\***

Se ``GEMINI_API_KEY`` estiver disponível, gera uma frase de log clínico com recomendação breve.

##### 5. **\*\*Registro rastreável\*\***

Cria entradas com ``timestamp``, paciente, vitais, status e ação sugerida.

Salva no ``logs.json`` (NoSQL).

#### Por que Dados Híbridos?

**\*\*Relacional (SQLite)\*\*:** adequado para dados estruturados com relacionamento paciente -> medições, integridade e consultas.

**\*\*Não relacional (JSON)\*\*:** adequado para logs de execução, auditoria e eventos, com escrita simples e leitura rápida.

#### Como Executar

```
```bash
```

```
cd automation
```

```
python database_setup.py
```

```
python rpa_monitor.py
```

...

Após rodar, valide:

`automation/data/patients.db` criado

`automation/data/logs.json` atualizado com os alertas

Conclusão

O fluxo implementa automação de ponta a ponta com dados híbridos, IA aplicada de forma coerente (regras + GenAI opcional) e rastreabilidade por logs estruturados.