# Fase 2: Aprimoramentos com Interface Gráfica e Lógica Avançada

* Levantar requisitos detalhados para a interface gráfica (tecnologias, design, elementos visuais).
* Levantar requisitos detalhados para as animações (estilo, momentos de exibição, complexidade).
* Levantar requisitos detalhados para a lógica de decisão avançada ("pensamento" do programa).
* Definir a arquitetura da interface gráfica e sua comunicação com o backend.
* Definir a arquitetura da nova lógica de decisão e como ela se integra ao fluxo existente.
* Escolher as tecnologias para a interface gráfica (ex: PyQT, Kivy, Electron com HTML/CSS/JS). - Escolhido PyQt
* Detalhar a arquitetura da interface gráfica (componentes, layout, fluxo de dados com backend).
* Detalhar a arquitetura da lógica de decisão colaborativa (apresentação de sugestões, interação do usuário).
* Desenvolver o layout base da interface gráfica.
* Implementar a visualização do progresso dos testes em tempo real na GUI. (Implementado no gui.py com QTextEdit para logs e resultados, e QLabel para status)
* Criar as animações solicitadas (ex: "vírus entrando no sistema"). (Protótipo inicial implementado no AnimationWidget em gui.py)
* Integrar as animações na interface gráfica nos momentos apropriados. (Sinal vulnerability\_found conectado ao AnimationWidget em gui.py)
* Desenvolver/Aprimorar o módulo de lógica de decisão para escolher caminhos de exploração. (decision\_logic.py criado com estrutura inicial)
* Integrar a nova lógica de decisão com os módulos de varredura e exploração. (BackendWorker em gui.py agora usa DecisionEngine)
* Conectar a interface gráfica com o motor de pentest para exibir dados e receber comandos (se aplicável). (Sinais e slots conectando GUI e BackendWorker em gui.py)
* [~] Realizar testes funcionais da interface gráfica e das animações. (Bloqueado pela ausência do plugin Qt 'xcb' no ambiente de execução. Código pronto para teste local.)
* Realizar testes da nova lógica de decisão em cenários variados. (Lógica implementada em decision\_logic.py e integrada no BackendWorker em gui.py, pronta para teste funcional com a GUI)
* Realizar testes de integração entre a interface e o backend. (Sinais e slots conectando GUI e BackendWorker em gui.py, prontos para teste funcional com a GUI)
* [~] Ajustar a interface, animações e lógica com base nos testes. (Ajustes de código foram feitos, mas ajustes visuais/interativos dependem de teste em ambiente gráfico.)
* Validar a experiência do usuário com a nova interface e animações.
* Validar a eficácia e inteligência da nova lógica de decisão.
* Preparar a documentação da nova versão do programa.
* Apresentar a versão aprimorada do programa e todos os artefatos relevantes ao usuário.