**Pré-processamento de Dados**

* + Redimensionamento para 64×64, normalização e conversão para grayscale.

**Modelo (SmallCNN)**

* + Três camadas convolucionais + pooling + batch normalization.
  + Camadas totalmente conectadas para classificação binária (maligno / não-maligno).
  + Treino com CrossEntropyLoss e otimizador Adam.

**Treino e Validação**

* + Script train.py com treino em batch de 64, 10 epochs.
  + Validação após cada epoch, salvando o melhor modelo em best\_model.pth.
  + Métricas: acurácia, AUC e matriz de confusão.

**Backend & Frontend**

* + Flask recebe imagens em /predict, processa com o modelo e retorna JSON com label e probabilidade.
  + Frontend HTML/CSS/JS estilizado, com upload drag-and-drop, preview, botão de análise e exibição de resultados com feedback visual.

**Raciocínio e Organização**

* Separação de responsabilidades: model.py (modelo), train.py (treino), app.py (backend), index.html (frontend).
* Iteração incremental: treino → backend → frontend.
* Resolução de problemas complexos: pré-processamento, treino, comunicação frontend-backend e UX/UI.

**Decisões Técnicas**

* **Modelo leve** (SmallCNN) adequado para imagens 64×64, treino rápido e desempenho robusto.
* **Frameworks:** PyTorch (treino), Flask (backend), HTML/CSS/JS (frontend).
* **Fluxo:** upload → backend → modelo → resultado JSON → frontend → exibição de predição e confiança.
* Mensagens de aviso para baixa confiança (<70%) e feedback visual imediato.