Documento de Requisitos e Arquitetura do Sistema

Projeto: Detecção de Erros em Impressão 3D

Versão: 3.0

Data: 01/03/2025

Equipe: Ana Larissa Teixeira, Antonio Everton Teixeira, Lemuel Santana, Maria Clara Pereira e Wagner Vasconcelos.

---

1. Introdução

1.1 Objetivos

Este documento tem como objetivo elencar os requisitos para o desenvolvimento do projeto de detecção de erros em impressões 3D. Este projeto visa desenvolver uma aplicação web capaz de processar imagens de impressões 3D e identificar possíveis falhas por meio de um modelo de inteligência artificial.

O sistema permitirá que usuários enviem imagens de impressões 3D, as quais serão analisadas por um modelo baseado em Máquinas de Vetores de Suporte (SVM) para detecção e classificação de erros comuns na impressão.

1.2 Escopo de validação inicial

Na primeira versão do projeto, será desenvolvida uma aplicação web estruturada para o processamento de imagens de impressões 3D. A aplicação permitirá que usuários façam upload das imagens, que serão analisadas por um modelo de IA, retornando informações sobre possíveis falhas na impressão.

2.Termos Aplicados

TERMO001: Inteligência Artificial (IA)
A Inteligência Artificial é um ramo da computação que busca criar sistemas capazes de executar tarefas que normalmente exigiriam inteligência humana. No contexto deste projeto, a IA será utilizada para classificar falhas em impressões 3D com base em um modelo treinado de Máquinas de Vetores de Suporte (SVM).

TERMO002: Impressão 3D
A impressão 3D é um processo de fabricação aditiva onde objetos tridimensionais são criados camada por camada. Defeitos comuns incluem warping (deformação das camadas), stringing (filamentos indesejados entre partes do objeto) e falhas de aderência.

TERMO002: SVM (Support Vector Machine)
A Máquina de Vetores de Suporte (SVM) é um modelo de aprendizado de máquina utilizado para classificação. Elas funcionam encontrando um hiperplano que melhor separa os dados em diferentes categorias. No contexto deste projeto, um modelo SVM é usado para classificar imagens de impressões 3D, determinando se apresentam ou não falhas com base nas características extraídas das imagens.

TERMO003: HOG (Histogram of Oriented Gradients)
O Histograma de Gradientes Orientados (HOG) é um método de extração de características que analisa a distribuição dos gradientes de intensidade em uma imagem. Ele é amplamente utilizado em visão computacional para detectar padrões, como bordas e contornos. No projeto, o HOG é aplicado para capturar informações estruturais das impressões 3D, auxiliando na identificação de falhas.

TERMO004: LBP (Local Binary Pattern)
O Padrão Binário Local (LBP) é um método de análise de textura que transforma uma imagem em um conjunto de valores binários com base na relação entre os pixels vizinhos. Essa técnica é útil para identificar padrões de textura e irregularidades na superfície de um objeto. No projeto, o LBP é utilizado para detectar variações na textura das impressões 3D, contribuindo para a identificação de possíveis defeitos.

3.Requisitos

3.1 Requisitos Funcionais e de capacidade

RF001: Upload de Imagens

- **Descrição:** A aplicação deve permitir que o usuário faça upload de imagens de impressões 3D nos formatos .jpg e .png, sem necessidade de login para maior praticidade.

- **Fluxo Principal:**
    - O usuário acessa a interface da aplicação.
    - O usuário seleciona a opção de upload de imagem.
    - O usuário escolhe a imagem a ser enviada.
    - O sistema recebe a imagem e valida o formato.
    - O sistema armazena temporariamente a imagem para processamento.

- **Critérios de Aceitação:**
    - O usuário consegue fazer upload de imagens sem precisar de login.
    - O sistema aceita somente imagens nos formatos .jpg e .png.
    - O sistema exibe uma mensagem de erro caso o formato não seja suportado.

RF002: Processamento de Imagens

- **Descrição:** O sistema deve processar a imagem enviada utilizando um modelo de IA baseado em SVM para identificar falhas.

- **Fluxo Principal:**
    - O sistema recebe a imagem enviada pelo usuário.
    - O sistema aplica pré-processamento na imagem.
    - O modelo de IA analisa a imagem e identifica possíveis falhas.
    - O sistema armazena os resultados temporariamente para exibição ao usuário.

- **Critérios de Aceitação:**
    - O modelo de IA deve processar a imagem em até 10 segundos.
    - O sistema deve retornar um resultado indicando a presença ou ausência de falhas.
    - O sistema deve notificar o usuário em caso de erro no processamento.

RF003: Exibição dos Resultados

- **Descrição:** A aplicação deve exibir a imagem enviada pelo usuário juntamente com a classificação do erro identificado. O container que exibe o resultado da análise deve mudar de cor para indicar visualmente o status da impressão.

- **Fluxo Principal:**

    - O sistema processa a imagem e obtém a classificação do erro.
    - O sistema exibe a imagem original junto com o resultado da análise.
    - O container da análise muda de cor conforme o resultado:
        - **verde:** para impressões sem erro
        - **vermelho:** para impressões com erro
    - O sistema permite que o usuário visualize os detalhes da falha detectada.

- **Critérios de Aceitação:**
    - O sistema exibe corretamente a imagem enviada.
    - O sistema exibe a classificação do erro de maneira clara e compreensível.
    - O container de exibição do resultado muda de cor de acordo com a análise realizada.
    - O usuário pode visualizar um resumo dos erros detectados na impressão.

RF004: Histórico de Processamento

- **Descrição:** O sistema deve armazenar as imagens processadas, juntamente com suas datas, horários e resultados, permitindo que o usuário consulte um histórico de análises anteriores.

- **Fluxo Principal:**
    - O sistema armazena a imagem enviada e seu resultado após o processamento.
    - O usuário acessa a seção de histórico da aplicação.
    - O sistema exibe a lista de imagens processadas, com data, hora e resultado da análise.
- **Critérios de Aceitação:**
    - O sistema mantém um histórico acessível sem necessidade de login.
    - O usuário pode visualizar as imagens analisadas anteriormente.
    - O histórico exibe corretamente os resultados associados a cada imagem.

3.2 Requisitos Não Funcionais

RNF001: Tempo de Resposta

- **Descrição:** O sistema deve processar as imagens e fornecer um resultado ao usuário em um tempo máximo de 10 segundos.

RNF002: Usabilidade e Navegação

- **Descrição:** A interface do sistema deve ser intuitiva e permitir fácil navegação para usuários sem conhecimento técnico avançado.

RNF003: Suporte a Formatos de Arquivo

- **Descrição:** O sistema deve suportar imagens nos formatos .jpg e .png, garantindo compatibilidade com os arquivos mais utilizados na área de impressão 3D.

RNF004: Armazenamento de Dados

- **Descrição:** O sistema deve ser capaz de armazenar imagens de até 5 MB e registrar um histórico de análises realizadas.

RNF005: Precisão do Modelo

- **Descrição:** O modelo de IA utilizado para detecção de erros deve atingir uma precisão mínima de 80% na identificação de falhas em impressões 3D.

4. Arquitetura do Sistema

4.1 Visão Geral

O sistema será composto por três componentes principais:

1. **Frontend (Interface do Usuário)**

   A interface gráfica do sistema foi desenvolvida com **React.js**, uma biblioteca Javascript de código aberto com foco em criar interfaces de usuário em páginas web. Suas principais funcionalidades são:

   - Upload de imagens para análise.
   - Exibição do resultado da análise do modelo de Machine Learning.
   - Histórico de imagens já analisadas, armazenadas no Firebase Firestore.

2. **Backend (Processamento e Análise de Imagens)**

   O backend foi implementado em Python usando o framework **Flask**. Ele é responsável por:

   - Receber a imagem enviada pelo frontend e armazená-la temporariamente.

- Extrair características da imagem utilizando o **feature_extractor**.
- Fazer a predição da imagem usando um modelo de **Máquina de Vetores de Suporte (SVM)**.
- Retornar o resultado ao frontend.

3. **Banco de Dados (Armazenamento do Histórico)**

   O armazenamento do histórico das análises é feito no **Firebase Firestore**, um banco de dados NoSQL na nuvem. Ele é utilizado para guardar as seguintes informações sobre os uploads:

   - Nome do arquivo enviado.
   - Data e hora do envio.
   - Resultado da análise do modelo.

4.2 Diagrama de Arquitetura

[Usuário]
|
[Frontend]
|
[Upload de Imagem]
|
[Backend - API Python]
|
[Modelo de IA (SVM)]
|
[Resultado Detectado]
|
[Banco de Dados]

4.3 Tecnologias Utilizadas

Componente	Tecnologia
Frontend	React
Backend	Python
Modelo de IA	SVM

Banco de dados	Firebase Firestore

4.4 Fluxo de Funcionamento

1. **Upload da Imagem:** O usuário faz o upload da imagem através da interface do sistema, observando os seguintes requisitos:
    - Apenas imagens nos formatos **.jpg** e **.png** são aceitas.
    - O tamanho do arquivo não pode ultrapassar **5MB**.

2. **Processamento Backend:** A imagem recebida é enviada ao backend via uma requisição HTTP. O backend recebe a imagem e a armazena temporariamente no servidor. Logo após, os seguintes passos são executados:
    1. **Extração de características:** A imagem passa pelo processo de extração de características que começa com o pré-processamento da imagem, onde ela é convertida para escala de cinza e redimensionada para um tamanho padrão. Em seguida, são extraídas características por meio do HOG e do LBP. Por fim, os vetores extraídos pelo HOG e LBP são concatenados, formando o vetor final de características que será usado pelo modelo.
    2. **Predição do modelo:** O vetor de características é passado como entrada para um modelo de Máquina de Vetores de Suporte (SVM) treinado previamente. O modelo retorna uma classificação:
        - 0 → Impressão sem erro
        - 1 → Impressão com erro

        O modelo SVM foi treinado utilizando um conjunto de imagens rotuladas e avaliado usando validação cruzada K-Fold (10 folds). Foram testadas duas versões do SVM:
        - SVM com C=10 e kernel RBF
        - SVM com C=100 e kernel RBF

        A versão final foi selecionada com base nas métricas Accuracy, Precision, Recall e F1 Score.
3. **Retorno do Resultado:** O sistema retorna o tipo de erro detectado ao usuário junto com a imagem.
4. **Armazenamento no Banco de Dados:** O histórico do upload é salvo no Firebase Firestore, permitindo que o usuário consulte as análises anteriores. O histórico salvo inclui o nome do arquivo, a data do upload e o resultado da predição.

5. **Histórico:** O usuário pode acessar a lista de imagens já processadas com os resultados.

5. Protótipo do Sistema

Protótipo Figma

6. Considerações Finais

Este documento descreveu os principais requisitos e a arquitetura do sistema de detecção de erros em impressões 3D. A implementação inicial focará no processamento de imagens via upload, visando simplicidade e clareza na entrega do projeto.