

# BigCard Training - Sistema de Avaliação

---

Documentação de Desenvolvimento

Gerado em: 10/01/2026

## Arquitetura

### Visão Geral

Sistema modular com perguntas e configurações personalizáveis via arquivos de texto, eliminando a necessidade de alterar código para personalizar questionários ou visual. Inclui script .bat para facilitar execução no Windows.

### Componentes Principais

1. ConfigLoader: Classe responsável por ler e interpretar o arquivo 'config.txt' (nome da instituição e cores)
2. QuestionLoader: Classe responsável por ler e interpretar o arquivo 'perguntas.txt'
3. BigCardHandler: Handler HTTP para servir o formulário e processar respostas
4. Gerador Dinâmico de HTML: Cria formulário baseado nas perguntas e configurações carregadas
5. executar.bat: Script Windows para inicialização simplificada (verifica Python e arquivos)

### Servidor HTTP (Python)

- Framework: 'http.server' (biblioteca padrão Python)
- Porta: 3000
- Protocolo: HTTP
- Host: 0.0.0.0 (aceita conexões de toda rede local)

## Arquivo config.txt

### Estrutura Geral

Arquivo de configuração para personalizar o nome da instituição e as cores do sistema sem modificar o código Python.

```
# Comentários começam com #  
# Linhas vazias são ignoradas
```

INSTITUICAO: Nome da Empresa

# Cor em diversos formatos:

COR: blue       # Nome da cor

COR: #0066cc    # Hexadecimal

COR: 0,102,204   # RGB

### Parâmetros Aceitos

- INSTITUICAO: Define o nome exibido no cabeçalho e no certificado PDF
- COR: Define a cor principal do sistema (bordas, botões, PDF)

### Formatos de Cor Suportados

- Nome em inglês: blue, red, green, yellow, orange, purple, pink, etc
- Hexadecimal: #0066cc, #ff5733, #28a745
- RGB: 0,102,204 ou rgb(0,102,204)

## Formato do Arquivo perguntas.txt

### Estrutura Geral

# Comentários começam com #

# Linhas vazias são ignoradas

NUMERO. Texto da pergunta

# Para múltipla escolha:

NUMERO. Texto da pergunta [MULTIPLA\_ESCOLHA]

a) Primeira alternativa

b) Segunda alternativa

c) Terceira alternativa

### Regras de Formatação

- Numeração: Cada pergunta deve começar com um número seguido de ponto e espaço
- Questões Abertas: Apenas o texto da pergunta
- Múltipla Escolha: Adicione '[MULTIPLA\_ESCOLHA]' após a pergunta
- Alternativas: Liste nas linhas seguintes com formato 'letra) texto'
- Letras válidas: a-j (até 10 alternativas)
- Comentários: Linhas iniciadas com '#' são ignoradas
- Linhas Vazias: Também são ignoradas

## Fluxo de Dados

### 1. Inicialização do Servidor

- ↓ Servidor Inicia
- ↓ ConfigLoader lê config.txt
- ↓ Valida e armazena nome da instituição e cor
- ↓ QuestionLoader lê perguntas.txt
- ↓ Valida formato e organiza dados
- ↓ Servidor aguarda conexões

### 2. Processo de Carregamento de Configurações

A classe ConfigLoader:

- Lê arquivo linha por linha
- Ignora comentários e linhas vazias
- Identifica chave:valor (INSTITUICAO ou COR)
- Parse da cor para múltiplos formatos (hex, RGB, nome)
- Armazena configurações para uso no HTML/PDF

Estrutura de Dados Interna da ConfigLoader:

```
{  
  'instituicao': 'BIGCARD',  
  'cor': '#0066cc',  
  'cor_rgb': (0, 102, 204)  
}
```

### 3. Geração Dinâmica do HTML

- Cliente acessa /
- Servidor chama get\_formulario\_html()
- Carrega configurações (instituição e cor) via ConfigLoader
- Itera sobre perguntas carregadas
- Para cada questão:
  - - Se aberta: gera <textarea>
  - - Se múltipla: gera <input type="radio"> para cada alternativa
- Injeta configurações de cor no CSS
- Injeta nome da instituição no header e PDF
- Retorna HTML completo com JavaScript

## Classe ConfigLoader

```
class ConfigLoader:
    def __init__(self, filename):
        self.filename = filename
        self.instituicao = "BIGCARD"
        self.cor = "#0066cc"
        self.cor_rgb = (0, 102, 204)
        self.load_config()

    def get_instituicao(self) -> str
    def get_cor(self) -> str
    def get_cor_rgb(self) -> tuple
```

## Endpoints

### GET `/` ou `/formulario`

- Descrição: Retorna o HTML do formulário de avaliação
- Content-Type: 'text/html; charset=utf-8'
- Resposta: HTML completo gerado dinamicamente com cores e instituição personalizadas

### POST `/enviar`

- Descrição: Recebe e salva as respostas do formulário
- Content-Type: 'application/json'
- Resposta sucesso: '{"success": true}'

## Configurações

### Arquivo de Configuração

CONFIG\_FILE = "config.txt" # Linha 18

### Arquivo de Perguntas

QUESTIONS\_FILE = "perguntas.txt" # Linha 17

### Porta do Servidor

PORT = 3000 # Linha 15

### Arquivo de Respostas

DATA\_FILE = "respostas.txt" # Linha 16

## Personalização Visual

### Como Funciona

O sistema utiliza as configurações do config.txt para aplicar cores dinamicamente em todo o sistema:

- CSS - Borda do cabeçalho: border-bottom: 3px solid {cor}
- CSS - Logo: color: {cor}
- CSS - Hover de questões: border-color: {cor}
- CSS - Focus de inputs: border-color: {cor}
- CSS - Botão primário: background: {cor}
- PDF - Cabeçalho: pdf.setFillColor(r, g, b)
- PDF - Linha de rodapé: pdf.setDrawColor(r, g, b)
- HTML - Nome da instituição no header e PDF

### Conversão de Cores

O método `_parse_color()` da `ConfigLoader` converte qualquer formato de cor (nome, hex, RGB) para uma tupla RGB usada no PDF:

Nome → RGB: "blue" → (0, 102, 204)

Hex → RGB: "#ff5733" → (255, 87, 51)

RGB → RGB: "255,0,0" → (255, 0, 0)

## Melhorias Futuras

6. Validação de Formato: Checar sintaxe do config.txt e perguntas.txt na inicialização
7. Hot Reload: Recarregar configurações e perguntas sem reiniciar servidor
8. Múltiplos Questionários: Suporte a vários arquivos de perguntas
9. Mais Opções de Personalização: Subtítulo, descrições, rodapé customizável
10. Banco de Dados: Migrar de .txt para SQLite
11. Dashboard Web: Interface para visualizar respostas
12. Exportação: CSV/Excel das respostas
13. Gabarito: Sistema de correção automática
14. Tipos Adicionais: Caixas de seleção, escala Likert, etc

## Tratamento de Erros

### Arquivo config.txt Ausente

Sistema usa valores padrão: BIGCARD e cor azul (#0066cc)

Mensagem de aviso no terminal

### Arquivo perguntas.txt Ausente

Servidor não inicia

Mensagem clara no terminal

### Formato Inválido em config.txt

Sistema ignora linhas malformadas

Usa valores padrão para campos não encontrados

## Validações Cliente

- Nome obrigatório
- Respostas abertas: mínimo 10 caracteres
- Múltipla escolha: seleção obrigatória
- Scroll automático para campo com erro



## Script executar.bat (Windows)

### Funcionalidade

O arquivo 'executar.bat' é um atalho para usuários Windows que:

15. Verifica se Python está instalado no sistema
16. Verifica se 'server.py' existe no diretório
17. Verifica se 'perguntas.txt' existe no diretório
18. Verifica se 'config.txt' existe no diretório
19. Inicia o servidor automaticamente
20. Exibe mensagens de erro claras se algo estiver errado

### Vantagens

- Não precisa abrir terminal manualmente
- Validações automáticas de pré-requisitos
- Mensagens de erro amigáveis para usuários não-técnicos
- Experiência "plug and play"
- Ideal para distribuição interna