

Nome: Henrique Moura Silva

Nome: Lucas Rafael Da Cruz Oliveira

Nome: Melissa Victória Araújo

Nome: Pedro Henrique Macedo Silva

Nome: Rodolfo Chinelato Rodrigues

FACULDADE DE TECNOLOGIA DE SÃO PAULO FATEC BARUERI

Consulta CNPJ – Guia Técnico

Manual de uso e referência

Trabalho apresentado à Faculdade de Tecnologia de São Paulo – FATEC
Barueri,
como requisito parcial para obtenção de nota na disciplina Projeto
interdisciplinar III,
do curso de Gestão Da Tecnologia Da Informação.

Orientador: Vander Ribeiro Elme

BARUERI – SP
2025

1. Visão Geral do Sistema	3
2. Público-Alvo.....	3
3. Requisitos Funcionais (RF)	3
RF01 – Consulta de CNPJ.....	3
RF02 – Validação básica do CNPJ no frontend / backend	3
RF03 – Submissão da consulta e requisição ao backend	4
RF04 – Recuperação e retorno de dados completos da empresa	4
RF05 – Tratamento de erros / falhas de consulta	4
RF06 – Interface responsiva / usabilidade básica.....	5
RF07 – Modularidade do Backend / Separação de Responsabilidades.....	5
4. Requisitos Não Funcionais (RNF).....	6
RNF01 – Performance.....	6
RNF02 – Segurança	6
RNF03 – Disponibilidade	6
RNF04 – Usabilidade	6
5. Fluxo do Usuário (User Flow)	6
6. Estrutura da API.....	7
Endpoint:	7
Resposta (exemplo):	7
Resposta (exemplo de erro):	8
Outros endpoints auxiliares:	8
Página única contendo:	9
7. Modelo do Banco de Dados (simplificado).....	9
9. Tecnologias	9
Conclusão.....	10

1. Visão Geral do Sistema

O sistema tem como objetivo permitir que qualquer usuário consulte informações de empresas através do CNPJ. O usuário digita um único dado — nesse caso, o próprio CNPJ — e o backend se encarrega de buscar todas as outras informações associadas, como razão social, nome fantasia, situação cadastral, endereço completo, CEP e demais dados relevantes.

A proposta é entregar uma experiência rápida, limpa e automatizada, evitando formulários extensos ou consultas manuais.

2. Público-Alvo

- Profissionais autônomos que precisam validar empresas rapidamente;
- Pequenas e médias empresas;
- Usuários comuns que querem conferir autenticidade de um estabelecimento;
- Equipes internas de análise, compliance ou cadastro.

3. Requisitos Funcionais (RF)

RF01 – Consulta de CNPJ

- A interface deve permitir que o usuário insira um CNPJ para consulta.
- Deve aceitar CNPJ com ou sem máscara (pontuação), mas o backend deve receber apenas os dígitos numéricos.

RF02 – Validação básica do CNPJ no frontend / backend

- Ao digitar o CNPJ, o sistema deve verificar se o formato está correto (quantidade de dígitos, caracteres permitidos).

- Caso o CNPJ seja inválido, deve exibir mensagem de erro clara, sem acionar a consulta

RF03 – Submissão da consulta e requisição ao backend

- Após validação, o sistema deve enviar uma requisição (via backend, no nosso caso, está em `consulta_cnpj.py/app_optimizado.py`) para buscar os dados da empresa. Ver em [GitHub](#)
- O backend deve interpretar o CNPJ informado, fazer a lógica de consulta (seja a partir de base própria, ou seja, API externa) e retornar os dados da empresa no formato esperado.

RF04 – Recuperação e retorno de dados completos da empresa

- O backend deve retornar ao frontend os dados principais da empresa, como:
 - Razão social
 - Nome fantasia
 - Situação cadastral
 - Endereço completo (logradouro, número, bairro, cidade, UF, CEP)
 - Outras informações relevantes — que estão disponíveis no código fonte.
- O frontend deve exibir esses dados de modo organizado em forma de lista para o usuário visualizar facilmente. Isso já aparece no repositório como parte da proposta.

RF05 – Tratamento de erros / falhas de consulta

- Se o CNPJ não existir / empresa não for encontrada, o sistema deve informar ao usuário (“Empresa não encontrada”, “CNPJ inválido” ou similar).
- Se houver falha de comunicação com a fonte externa (se estiver usando API externa) ou erro no backend, o sistema deve capturar e exibir uma mensagem amigável de erro.

RF06 – Interface responsiva / usabilidade básica

- A página deve ser legível e funcional — HTML + CSS + JS foram as ferramentas utilizadas para estilizar, organizar e responsivo
- A experiência deve ser simples: só o campo para o CNPJ + botão de “Consultar” + área de resultado / erro. Evitar poluição visual ou campos desnecessários

RF07 – Modularidade do Backend / Separação de Responsabilidades

O backend do sistema de consulta de CNPJ foi desenvolvido com base nos princípios de **modularidade** e **separação de responsabilidades**, garantindo maior clareza, manutenção facilitada e escalabilidade da aplicação. Cada componente possui uma função específica e independente, evitando sobreposição de responsabilidades e promovendo a reutilização de código.

- **Configurações centralizadas:** parâmetros como tamanho do pool de conexões e limite de cache são definidos em um único local, permitindo ajustes rápidos sem necessidade de alterar o código principal.
- **Gerenciador de conexões (ConnectionPoolManager):** responsável exclusivamente pelo acesso ao banco de dados, utilizando pool de conexões para otimizar desempenho e reduzir custos de abertura de novas conexões.
- **Cache em memória (CNPJCache):** módulo independente que armazena resultados de consultas, aplicando política de substituição LRU (Least Recently Used) para manter eficiência e reduzir acessos repetidos ao banco.
- **Funções utilitárias:** tratam da manipulação e formatação de CNPJs, isolando regras específicas e evitando duplicação de código.
- **Funções de consulta:** concentram a lógica de busca no banco de dados e integração com o cache, mantendo a separação entre persistência e lógica de negócio.

- **Rotas Flask:** responsáveis apenas pela comunicação com o cliente, recebendo requisições e retornando respostas em formato JSON ou HTML, sem interferir na lógica interna de consulta ou cache.

Essa arquitetura modular assegura que cada parte do sistema possa ser modificada ou expandida sem impactar negativamente os demais componentes. Além disso, a separação de responsabilidades contribui para a **robustez, escalabilidade e manutenção futura** do projeto, atendendo plenamente ao requisito funcional RF07.

4. Requisitos Não Funcionais (RNF)

RNF01 – Performance

A resposta deve ocorrer em no máximo 3 segundos.

RNF02 – Segurança

- Comunicação HTTPS
- Sanitização do input
- Possível aplicação de rate limit se o sistema for público

RNF03 – Disponibilidade

O sistema deve estar acessível 24/7, exceto em períodos planejados de manutenção.

RNF04 – Usabilidade

A interface deve ser simples, com layout responsivo.

5. Fluxo do Usuário (User Flow)

1. Usuário entra no site.
2. Digita o CNPJ no campo principal.

3. O sistema valida o CNPJ.
4. O frontend envia o CNPJ limpo ao backend.
5. O backend consulta a base/API externa.
6. O backend retorna um JSON com todos os dados.
7. O frontend monta os cards com as informações.
8. Usuário vê as informações e pode consultar outro CNPJ.

6. Estrutura da API

A API desenvolvida para o sistema de consulta de CNPJ segue uma estrutura modular, com endpoints específicos para cada funcionalidade. O endpoint principal /consultar recebe um CNPJ via requisição POST e retorna os dados da empresa em formato JSON, incluindo razão social, situação cadastral e endereço. Além disso, foram implementados endpoints auxiliares para estatísticas do sistema e limpeza de cache, garantindo maior controle e manutenção da aplicação.

Endpoint:

- **Método:** POST
- **URL:** /consultar
- **Descrição:** Recebe um CNPJ em formato JSON e retorna os dados da empresa consultada.
- **Parâmetros de entrada (body JSON):**

```
{  
  "cnpj": "12345678000195"  
}
```

Resposta (exemplo):

```
{  
  "cnpj": "12.345.678/0001-95",  
  "nome": "EMPRESA EXEMPLO LTDA",
```

```
"situacao": "02",  
"endereco": "Rua das Flores, 123, Centro",  
"cep": "06400000",  
"municipio": "Barueri",  
"uf": "SP",  
"ativa": true  
}
```

Resposta (exemplo de erro):

```
{  
  "erro": "CNPJ não fornecido"  
}
```

Outros endpoints auxiliares:

GET /estatisticas Retorna informações sobre o cache e pool de conexões:

```
{  
  "cache_size": 150,  
  "pool_min": 5,  
  "pool_max": 20  
}
```

POST /limpar-cache Limpa o cache de CNPJs armazenados:

```
{  
  "mensagem": "Cache limpo com sucesso"  
}
```

7. Layout Ideal do Site

Página única contendo:

- Header simples com título: *Consultar CNPJ*
- Campo único de busca
- Botão "Consultar"
- Área de carregamento (spinner)
- Seção de resultado em card
- Mensagens de erro claras

7. Modelo do Banco de Dados (simplificado)

Tabela: Empresas

Campo	Tipo	Descrição
id	INT	Chave primária
cnpj	VARCHAR(14)	CNPJ sem máscara
razao_social	VARCHAR(255)	Nome jurídico
nome_fantasia	VARCHAR(255)	Nome fantasia
situacao	VARCHAR(50)	Situação cadastral
data_abertura	DATE	Data de abertura
cnae	VARCHAR(20)	Código CNAE
logradouro	VARCHAR(255)	Endereço
numero	VARCHAR(20)	Número
bairro	VARCHAR(100)	Bairro
cidade	VARCHAR(100)	Cidade
uf	CHAR(2)	Estado
cep	VARCHAR(9)	CEP
natureza_juridica	VARCHAR(255)	Tipo jurídico

9. Tecnologias

O projeto foi desenvolvido utilizando tecnologias modernas e consolidadas. No **frontend**, empregaram-se HTML5, CSS3 e JavaScript para construção da interface e interação com o usuário. O **backend** foi implementado em Python, utilizando o framework Flask para gerenciamento das rotas e a biblioteca Psycopg2 para comunicação com o banco de dados. Por fim, o **banco de dados** escolhido foi o PostgreSQL, devido à sua robustez, suporte a consultas complexas e confiabilidade no armazenamento das informações cadastrais das empresas.

Conclusão

O desenvolvimento do sistema de consulta de CNPJ evidencia como uma solução tecnológica simples, quando planejada de forma estruturada, pode oferecer **eficiência e praticidade** tanto para usuários comuns quanto para profissionais que necessitam validar informações empresariais em suas atividades cotidianas. A proposta de permitir a consulta a partir de um único dado — o próprio CNPJ — reduz etapas, elimina retrabalhos e agiliza o processo de verificação de dados corporativos.

A implementação de um **backend modular e escalável**, aliado a um **frontend leve e responsivo**, resultou em um fluxo funcional limpo e objetivo. Os requisitos levantados garantiram não apenas a funcionalidade da consulta, mas também aspectos essenciais como **segurança, desempenho e escalabilidade**, tornando o sistema robusto e confiável.

Adicionalmente, a **estrutura modular** adotada abre espaço para futuras evoluções, como integração com novas APIs, aprimoramento do cache de consultas, criação de dashboards de estatísticas, autenticação de usuários e exportação dos resultados. Dessa forma, o projeto não apenas cumpre sua função principal, como também estabelece uma base sólida para aplicações mais complexas e completas no contexto empresarial e administrativo.

Em síntese, o sistema alcança seu objetivo com eficiência, mantendo a simplicidade na interface e a solidez na lógica de backend. Trata-se de uma **fundação consistente para expansão futura**, capaz de sustentar projetos voltados à validação e análise de dados corporativos em diferentes cenários.

