# Entrega Técnica para Validação

**Projeto:** **OpenCEP**

**Empresa Parceira:** CS Prime

**Equipe Executora:** Bruno Felipe de Souza Araujo

**Data da Entrega:** 10/10/2025

**Versão da Entrega:** v1.0.1

## 1. Descrição da Entrega

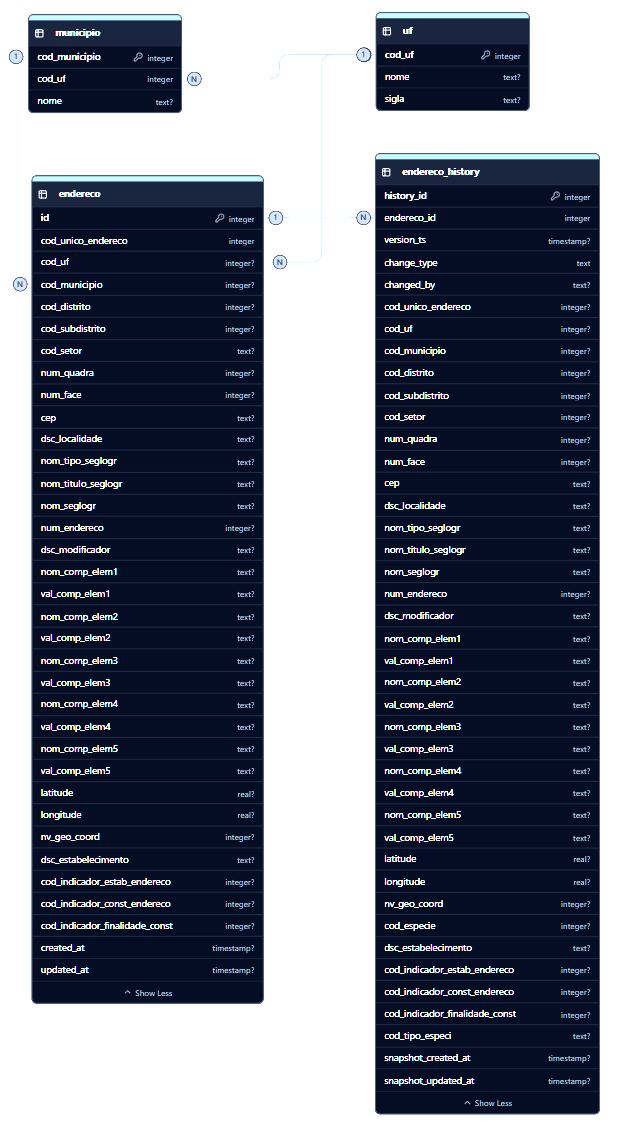
Esta entrega consiste na versão v1.0.1 da plataforma OpenCEP, uma aplicação web completa para consulta e gerenciamento de Códigos de Endereçamento Postal (CEPs) do Brasil. O funcionamento principal ocorre através de uma interface web intuitiva que permite aos usuários não apenas consultar endereços, mas também popular e atualizar a base de dados local. Adicionalmente, a plataforma expõe uma API RESTful para permitir a integração e automação de consultas por sistemas externos. A solução é construída em Python com o framework Flask e utiliza um banco de dados SQLite (OpenCEP.db), cuja estrutura é criada por script e os dados são carregados dinamicamente pela própria aplicação.

## 2. Escopo Incluído nesta Entrega

* Plataforma Web Interativa (index.html):
  + Consulta Individual de CEP: Campo para inserção de um CEP que retorna o endereço completo.
  + Consulta Reversa por Filtros: Funcionalidade de busca que permite encontrar endereços filtrando por UF, cidade e logradouro, sem a necessidade de informar o CEP.
  + Consulta de CEPs em Lote: Módulo para upload de arquivos (CSV) contendo múltiplos filtros de CEPs
  + Download do resultado consolidado da consulta em arquivo (CSV).
  + Módulo de Carga de Dados: Interface dedicada a popular o banco de dados ou realizar atualizações de endereços a serem consultados, através do upload de arquivos de dados compactados (formato .ZIP) referente a tabelas oficiais da plataforma de dados Sidra / IBGE referente ao censo 2022. (Disponível em: [Cadastro Nacional de Endereços para Fins Estatísticos | IBGE](https://www.ibge.gov.br/estatisticas/sociais/populacao/38734-cadastro-nacional-de-enderecos-para-fins-estatisticos.html?=&t=downloads)).
* API RESTful para Integração:
  + Endpoint GET /api/v1/cep/{cep} para consulta programática de um CEP específico.
  + Endpoint POST /api/v1/ceps para requisição de lotes através da passagem dos filtros utilizando JSON via API.
* Base de Dados e Estrutura:
  + Script SQL (criacaoBancoDadosEnderecos.sql) para a criação da estrutura do banco de dados SQLite.
  + Lógica de aplicação para processar e inserir os dados de endereçamento no banco a partir dos arquivos fornecidos na interface web.
* Configuração e Ambiente:
  + Arquivo config.json para parametrização da aplicação.
  + Arquivo requirements.txt com todas as dependências para garantir a replicabilidade do ambiente de execução.
* Dicionário de Dados e Diagrama da Base
  + A estrutura de dados do OpenCEP foi projetada para armazenar informações de endereçamento de forma normalizada e eficiente, garantindo a integridade dos dados e a capacidade de auditoria através de um sistema de versionamento.
* Arquivo de logs do sistema (app.log)

**Diagrama de Entidade-Relacionamento (DER)**

O diagrama abaixo ilustra as principais tabelas e seus relacionamentos.



**Dicionário de Dados Detalhado**

**Tabela: uf**

Armazena as Unidades Federativas do Brasil.

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo de Dado** | **Descrição** |
| cod\_uf | INTEGER | **Chave Primária.** Código numérico da UF, conforme padrão IBGE. |
| nome | TEXT | Nome por extenso da Unidade Federativa (ex: "São Paulo"). |
| sigla | TEXT | Sigla da Unidade Federativa (ex: "SP"). |

**Tabela: municipio**

Armazena os municípios brasileiros e sua relação com a UF.

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo de Dado** | **Descrição** |
| cod\_municipio | INTEGER | **Chave Primária.** Código numérico do município, conforme padrão IBGE. |
| cod\_uf | INTEGER | **Chave Estrangeira.** Referência uf(cod\_uf). |
| nome | TEXT | Nome por extenso do município. |

**Tabela: endereco**

Tabela principal que armazena os detalhes de cada registro de endereço.

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo de Dado** | **Descrição** |
| id | INTEGER | Chave Primária. Identificador sequencial único para cada registro na tabela. |
| cod\_unico\_endereco | INTEGER | Código de identificação único do endereço na base de dados de origem (ex: CNEFE do IBGE). |
| cod\_uf | INTEGER | Chave Estrangeira. Código da Unidade Federativa. Referencia uf(cod\_uf). |
| cod\_municipio | INTEGER | Chave Estrangeira. Código do Município. Referencia municipio(cod\_municipio). |
| cod\_distrito | INTEGER | Código do Distrito. |
| cod\_subdistrito | INTEGER | Código do Subdistrito. |
| cod\_setor | TEXT | Código do Setor Censitário. |
| num\_quadra | INTEGER | Número da quadra dentro do setor censitário. |
| num\_face | INTEGER | Número da face da quadra. |
| cep | TEXT | Código de Endereçamento Postal. |
| dsc\_localidade | TEXT | Descrição da localidade ou bairro. |
| nom\_tipo\_seglogr | TEXT | Tipo do segmento de logradouro (ex: "Rua", "Avenida", "Praça"). |
| nom\_titulo\_seglogr | TEXT | Título associado ao logradouro (ex: "Doutor", "Presidente", "General"). |
| nom\_seglogr | TEXT | Nome do logradouro (ex: "Paulista", "da Sé"). |
| num\_endereco | INTEGER | Número do endereço no logradouro. |
| dsc\_modificador | TEXT | Modificador do endereço (ex: "lado ímpar", "km 10", "antigo"). |
| nom\_comp\_elem1 | TEXT | Nome do primeiro complemento (ex: "BLOCO"). |
| val\_comp\_elem1 | TEXT | Valor do primeiro complemento (ex: "A"). |
| nom\_comp\_elem2 | TEXT | Nome do segundo complemento (ex: "ANDAR"). |
| val\_comp\_elem2 | TEXT | Valor do segundo complemento (ex: "10"). |
| nom\_comp\_elem3 | TEXT | Nome do terceiro complemento (ex: "APARTAMENTO") |
| val\_comp\_elem3 | TEXT | Valor do terceiro complemento (ex: "101"). |
| nom\_comp\_elem4 | TEXT | Nome do quarto complemento. |
| val\_comp\_elem4 | TEXT | Valor do quarto complemento. |
| nom\_comp\_elem5 | TEXT | Nome do quinto complemento. |
| val\_comp\_elem5 | TEXT | Valor do quinto complemento. |
| latitude | REAL | Coordenada geográfica de latitude. |
| longitude | REAL | Coordenada geográfica de longitude. |
| nv\_geo\_coord | INTEGER | Nível de precisão da coordenada geográfica. |
| cod\_especie | INTEGER | Código da espécie do endereço (ex: Domicílio, Estabelecimento). |
| dsc\_estabelecimento | TEXT | Nome ou descrição do estabelecimento, se aplicável. |
| cod\_indicador\_estab\_endereco | INTEGER | Código indicador se o endereço é exclusivo de um estabelecimento. |
| cod\_indicador\_const\_endereco | INTEGER | Código indicador do tipo de construção no endereço. |
| cod\_indicador\_finalidade\_const | INTEGER | Código indicador da finalidade da construção (residencial, comercial). |
| cod\_tipo\_especi | INTEGER | Código de especificação do tipo de endereço. |
| created\_at | DATETIME | Data e hora da inserção do registro no banco de dados. |
| updated\_at | DATETIME | Data e hora da última atualização do registro. |

**Tabela: endereco\_history**

Tabela de auditoria que armazena um snapshot do registro da tabela endereco antes de qualquer alteração (INSERT, UPDATE, DELETE).

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo de Dado** | **Descrição** |
| history\_id | INTEGER | Chave Primária. Identificador sequencial único para cada registro de histórico. |
| endereco\_id | INTEGER | Chave Estrangeira. Referencia o id do registro original na tabela endereco. |
| version\_ts | DATETIME | Data e hora exatas em que a alteração ocorreu. |
| change\_type | TEXT | Tipo da operação que gerou o histórico: 'INSERT', 'UPDATE' ou 'DELETE'. |
| changed\_by | TEXT | Campo reservado para identificar o usuário ou processo que realizou a alteração. |
| cod\_unico\_endereco | INTEGER | Snapshot do campo cod\_unico\_endereco. |
| cod\_uf | INTEGER | Snapshot do campo cod\_uf. |
| cod\_municipio | INTEGER | Snapshot do campo cod\_municipio. |
| cod\_distrito | INTEGER | Snapshot do campo cod\_distrito. |
| cod\_subdistrito | INTEGER | Snapshot do campo cod\_subdistrito. |
| cod\_setor | INTEGER | Snapshot do campo cod\_setor. |
| num\_quadra | INTEGER | Snapshot do campo num\_quadra. |
| num\_face | INTEGER | Snapshot do campo num\_face. |
| cep | TEXT | Snapshot do campo cep. |
| dsc\_localidade | TEXT | Snapshot do campo dsc\_localidade. |
| nom\_tipo\_seglogr | TEXT | Snapshot do campo nom\_tipo\_seglogr. |
| nom\_titulo\_seglogr | TEXT | Snapshot do campo nom\_titulo\_seglogr. |
| nom\_seglogr | TEXT | Snapshot do campo nom\_seglogr. |
| num\_endereco | INTEGER | Snapshot do campo num\_endereco. |
| dsc\_modificador | TEXT | Snapshot do campo dsc\_modificador. |
| nom\_comp\_elem1 | TEXT | Snapshot do campo nom\_comp\_elem1. |
| val\_comp\_elem1 | TEXT | Snapshot do campo val\_comp\_elem1. |
| nom\_comp\_elem2 | TEXT | Snapshot do campo nom\_comp\_elem2. |
| val\_comp\_elem2 | TEXT | Snapshot do campo val\_comp\_elem2. |
| nom\_comp\_elem3 | TEXT | Snapshot do campo nom\_comp\_elem3. |
| val\_comp\_elem3 | TEXT | Snapshot do campo val\_comp\_elem3. |
| nom\_comp\_elem4 | TEXT | Snapshot do campo nom\_comp\_elem4. |
| val\_comp\_elem4 | TEXT | Snapshot do campo val\_comp\_elem4. |
| nom\_comp\_elem5 | TEXT | Snapshot do campo nom\_comp\_elem5. |
| val\_comp\_elem5 | TEXT | Snapshot do campo val\_comp\_elem5. |
| latitude | REAL | Snapshot do campo latitude. |
| longitude | REAL | Snapshot do campo longitude. |
| nv\_geo\_coord | INTEGER | Snapshot do campo nv\_geo\_coord. |
| cod\_especie | INTEGER | Snapshot do campo cod\_especie. |
| dsc\_estabelecimento | TEXT | Snapshot do campo dsc\_estabelecimento. |
| cod\_indicador\_estab\_endereco | INTEGER | Snapshot do campo cod\_indicador\_estab\_endereco. |
| cod\_indicador\_const\_endereco | INTEGER | Snapshot do campo cod\_indicador\_const\_endereco. |
| cod\_indicador\_finalidade\_const | INTEGER | Snapshot do campo cod\_indicador\_finalidade\_const. |
| cod\_tipo\_especi | TEXT | Snapshot do campo cod\_tipo\_especi. |
| snapshot\_created\_at | DATETIME | Snapshot do campo created\_at do registro original. |
| snapshot\_updated\_at | DATETIME | Snapshot do campo updated\_at do registro original. |

**Visões e Triggers**

* View vw\_endereco\_full
  + É uma visão (VIEW) que une as tabelas endereco, municipio e uf para facilitar consultas, retornando um registro de endereço completo com os nomes do município e da UF, além da sigla da UF, sem a necessidade de realizar JOINs manualmente nas consultas.
* Triggers (trg\_endereco\_\*)
  + O sistema utiliza gatilhos (TRIGGERS) no banco de dados para automatizar a criação de registros de auditoria na tabela endereco\_history.
  + trg\_endereco\_after\_insert: Após um novo endereço ser inserido, uma cópia é salva no histórico com o tipo 'INSERT'.
  + trg\_endereco\_after\_update: Após um endereço ser atualizado, o estado anterior à modificação é salvo no histórico com o tipo 'UPDATE'.
  + trg\_endereco\_before\_delete: Antes de um endereço ser deletado, seu estado final é salvo no histórico com o tipo 'DELETE'.
  + trg\_endereco\_before\_update: Garante que o campo updated\_at da tabela endereco seja automaticamente atualizado com a data e hora correntes sempre que um registro for modificado.

## 3. Escopo Não Incluído / Fora desta Entrega

* Autenticação e Níveis de Acesso: A plataforma não possui sistema de login, perfis de usuário ou controle de acesso às funcionalidades.
* Sincronização Automática da Base: Não há um serviço que busca e aplica automaticamente as atualizações da base de dados do IBGE; a atualização depende da carga manual de arquivos na interface.
* Dashboard de Monitoramento: Inexistência de um painel com métricas de uso, logs de consulta ou estatísticas de acesso à API.
* Suporte a outros SGBDs: A aplicação foi desenvolvida e testada exclusivamente com SQLite.
* Containerização: A entrega não inclui arquivos de configuração para Docker, Kubernetes ou outras tecnologias de contêiner.
* Deploy em nuvem para testes em produção, pois devido ao tamanho da base geraria um custo para esse escopo.

## 4. Ambiente e Acesso

A aplicação foi projetada para ser executada em um servidor web local ou em nuvem.

* Projeto disponível em: https://github.com/Brunofsaraujo/CDN\_ProjetoIntegrador4/tree/developer/OpenCEP
* Plataforma Principal: Interface Web
* URL de Acesso: http://127.0.0.1:5000 (execução local)
* Acesso à API (para integrações):
  + Consulta Individual: GET http://127.0.0.1:5000/api/v1/cep/{cep}
  + Consulta em Lote: POST http://127.0.0.1:5000/api/v1/ceps

Não são necessárias credenciais de acesso para utilizar a plataforma ou a API.

Aplicação desenvolvida para execução online mas devido custos, a presente entrega deve ser executada localmente.

## 5. Critérios de Validação

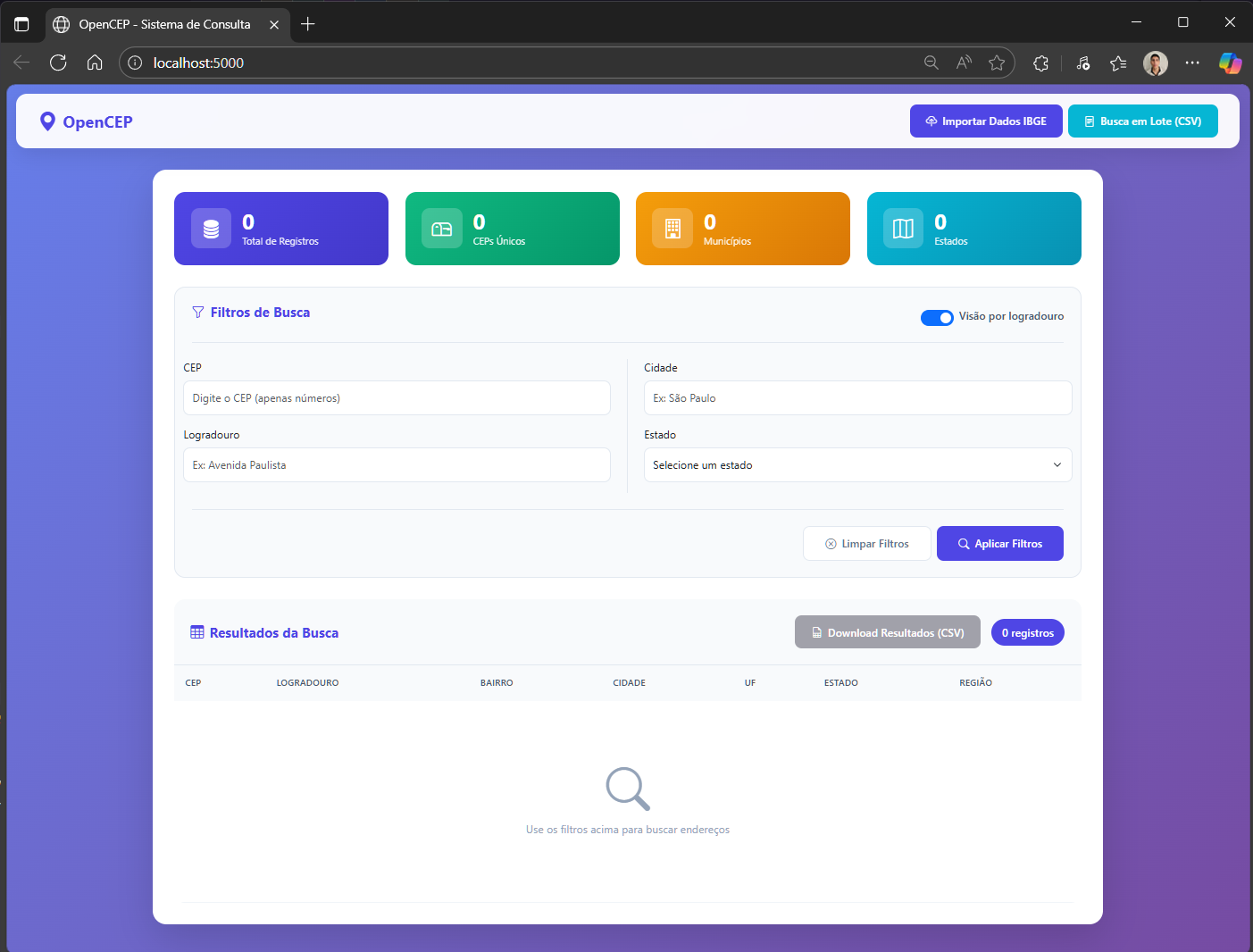
Para que a entrega seja considerada aceita, os seguintes critérios devem ser atendidos:

* Critério 1: A aplicação inicia corretamente e a interface web é carregada sem erros no navegador ao acessar a URL base.
* Critério 2: A funcionalidade de carga de dados permite o upload de um arquivo .zip (ex: 35\_SP.zip), populando o banco de dados com os respectivos endereços.
* Critério 3: A consulta individual por um CEP válido na interface web (ex: 18110008) exibe o endereço correspondente corretamente na tela.
* Critério 4: A consulta reversa, utilizando os filtros de UF "SP" e Cidade "São Paulo", retorna uma lista de endereços que correspondem aos critérios.
* Critério 5: O upload de um arquivo modelo\_busca\_lote.csv na funcionalidade de consulta em lote.
* Critério 6: Download de um arquivo export\_open\_cep.csv com os dados de endereço.
* Critério 7: Uma requisição GET direta para o endpoint da API (/api/cep/18110008) retorna um status 200 OK e a resposta em formato JSON
* Critério 8: Uma requisição GET direta para o endpoint da API (/api/bulk-search) retorna um status 200 OK e a resposta em formato JSON com a lista de filtros enviadas por body para consulta em lote.
* Critério 9: Arquivo app.log escrevendo os logs do sistema

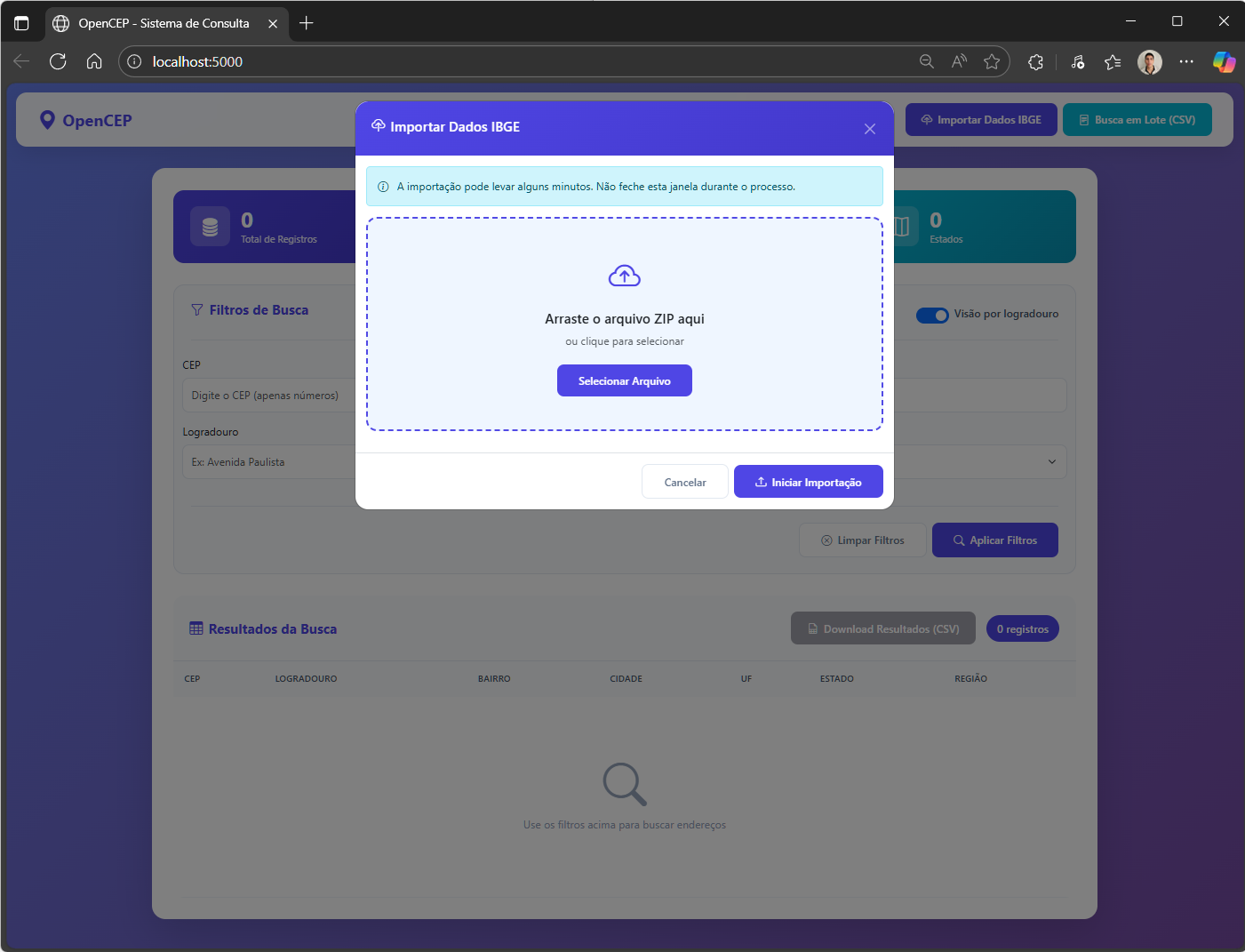
## 6. Evidências da Entrega

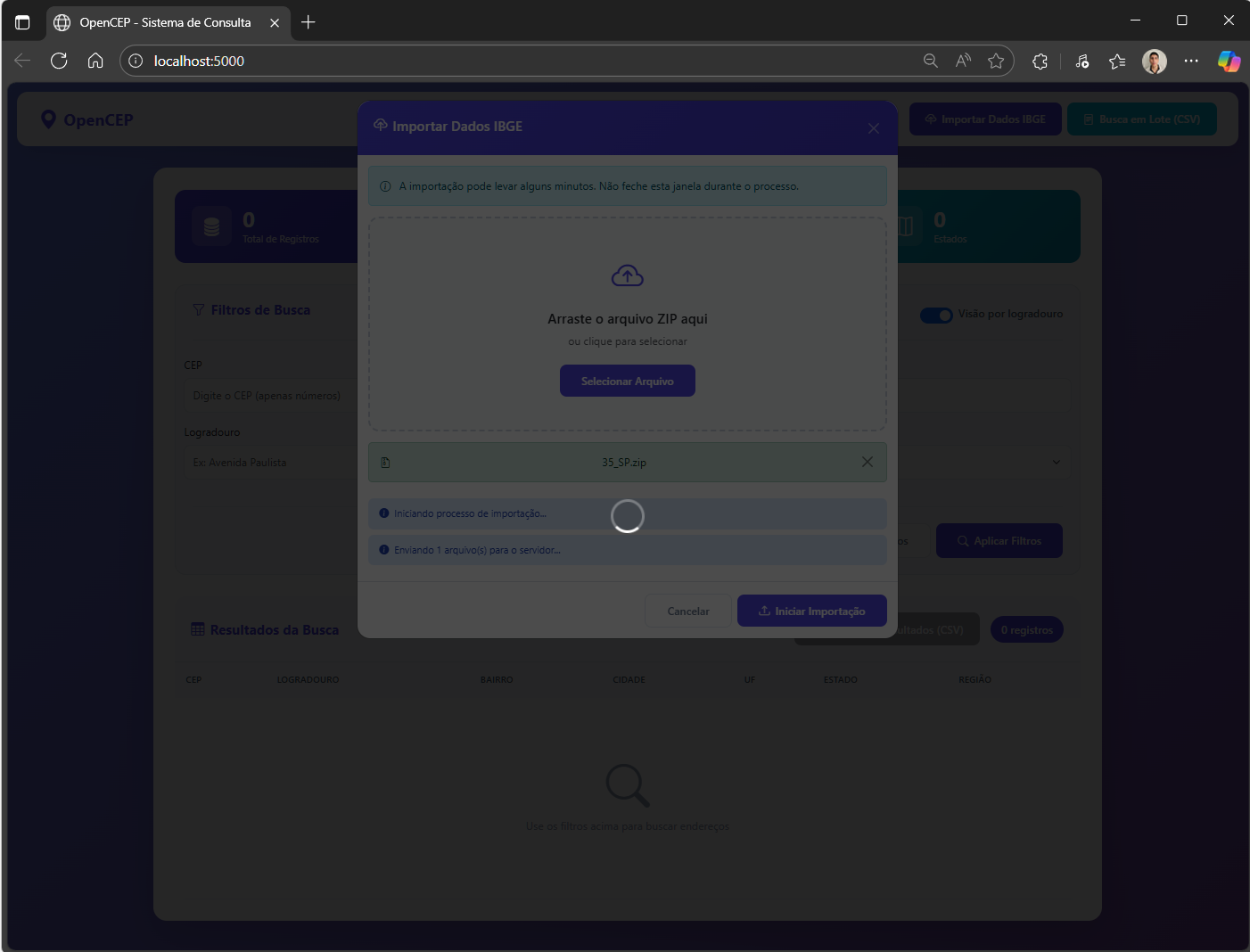
Prints demonstrando os critérios de validação.

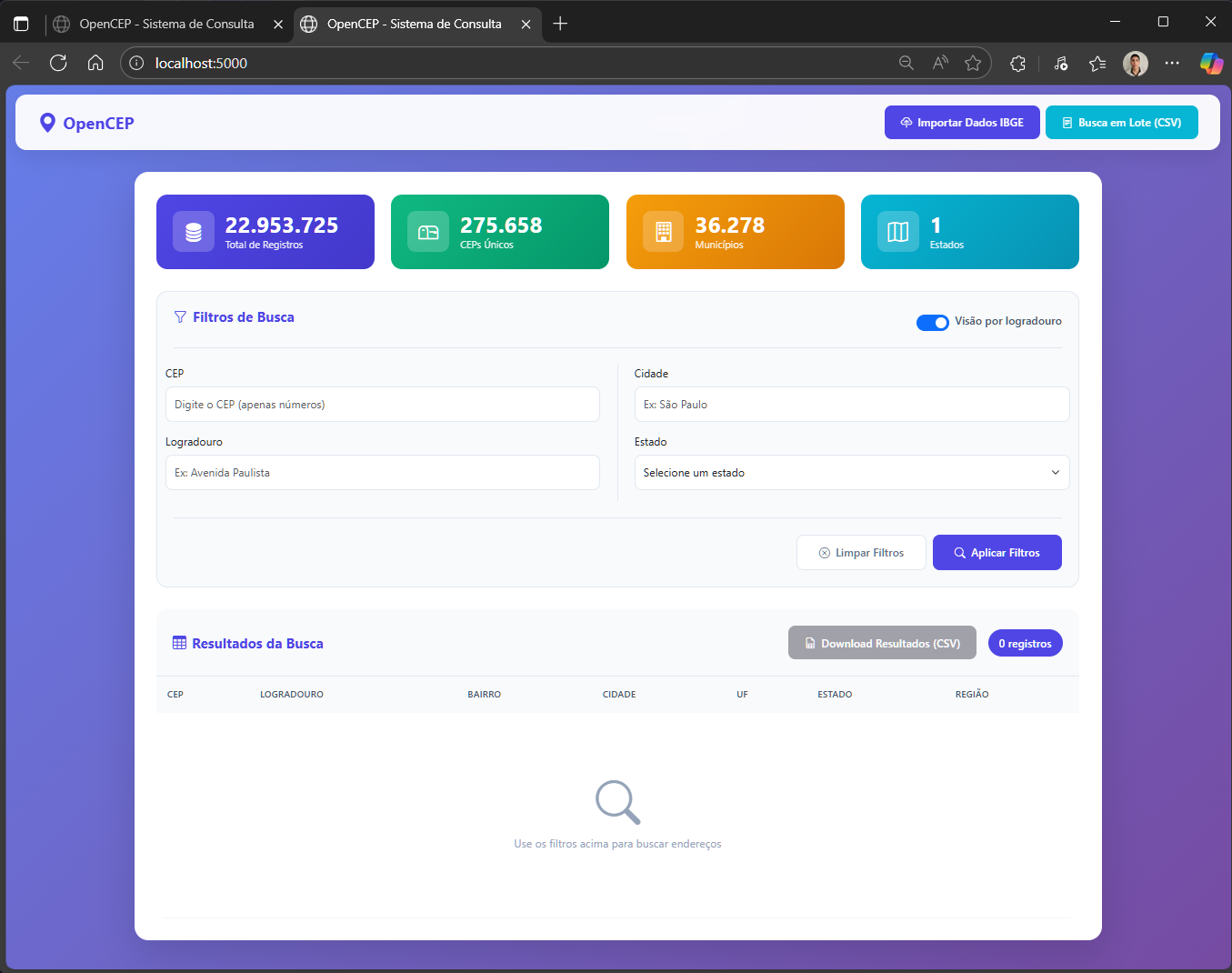
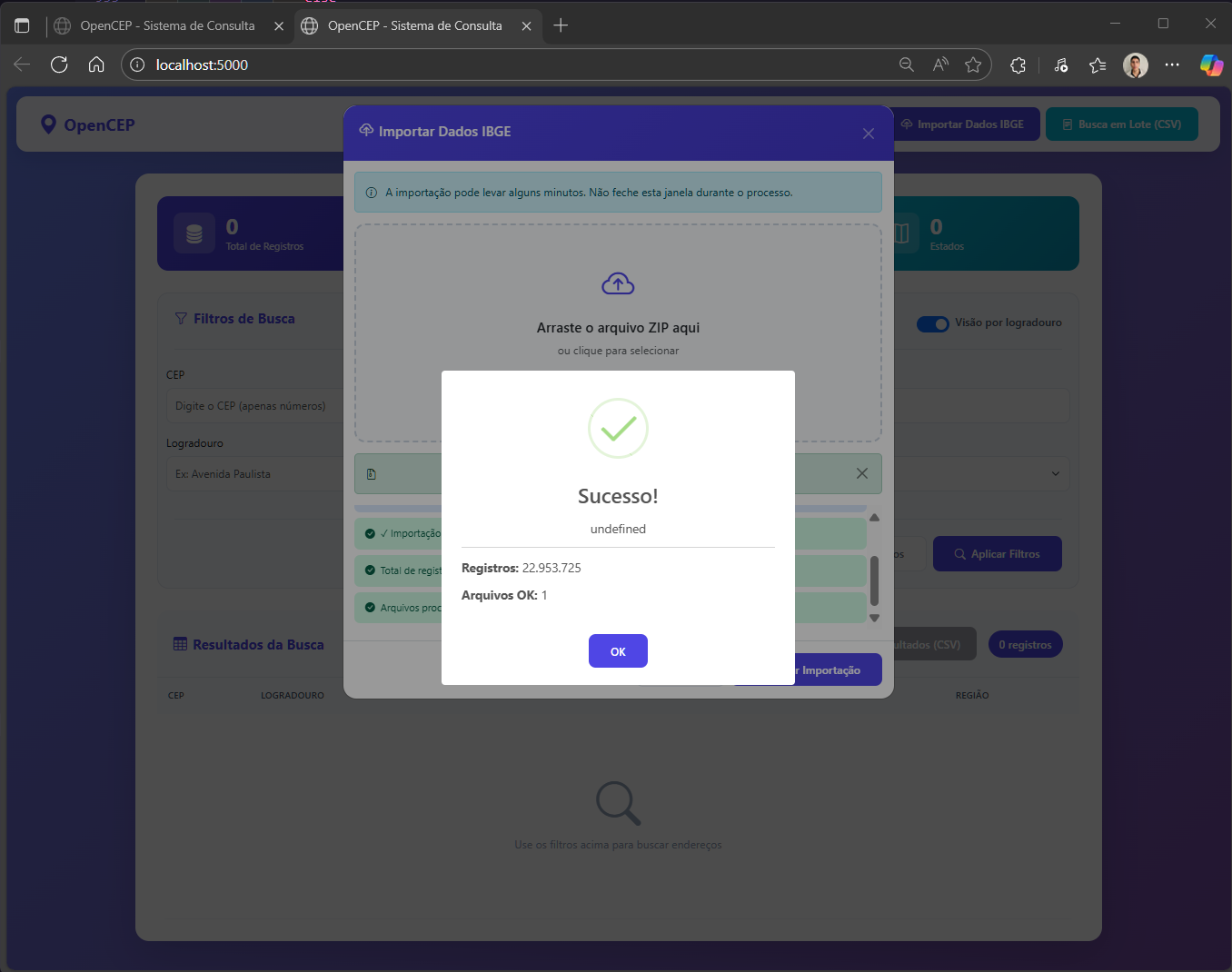
* Evidência 1 — Inicialização da Aplicação  
  Print da **tela principal da plataforma OpenCEP**, exibindo a interface web carregada corretamente no navegador ao acessar a URL base, confirmando o funcionamento inicial da aplicação.



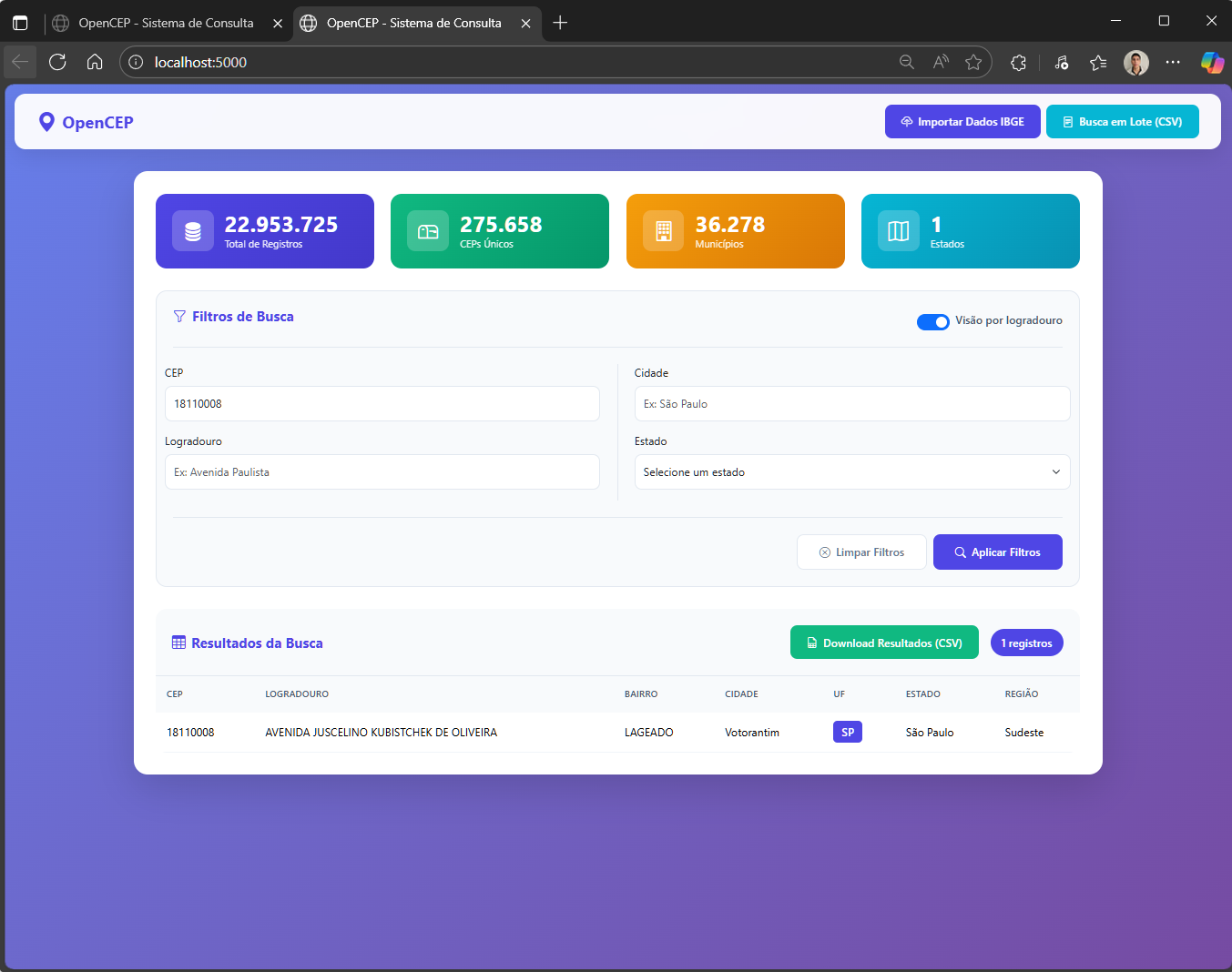
* **Evidência 2 — Carga de Dados (.ZIP)**  
  Print da **interface de carga de dados**, mostrando o upload de um arquivo .zip (ex: 35\_SP.zip) sendo realizado com sucesso e o respectivo carregamento das informações no banco de dados.

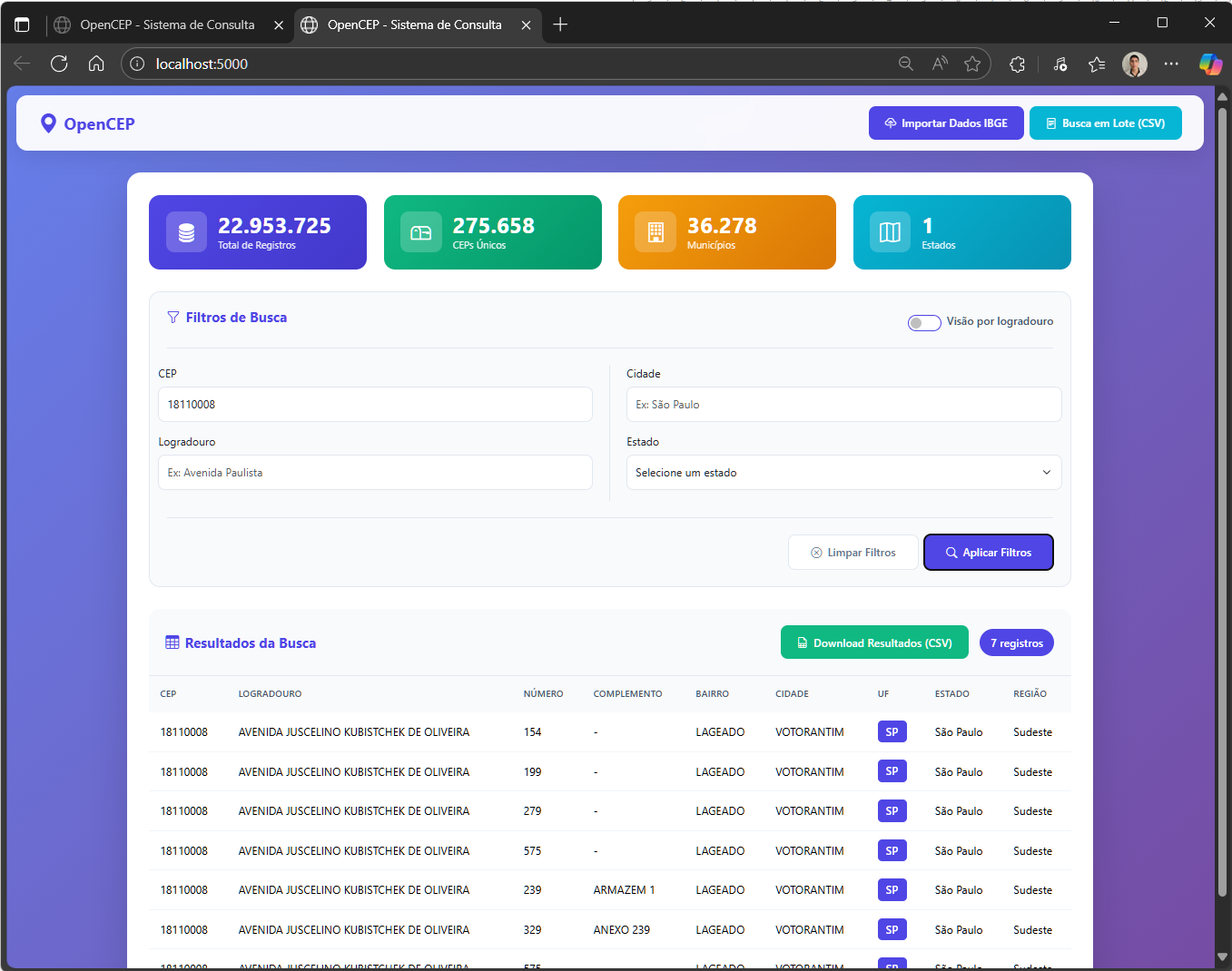




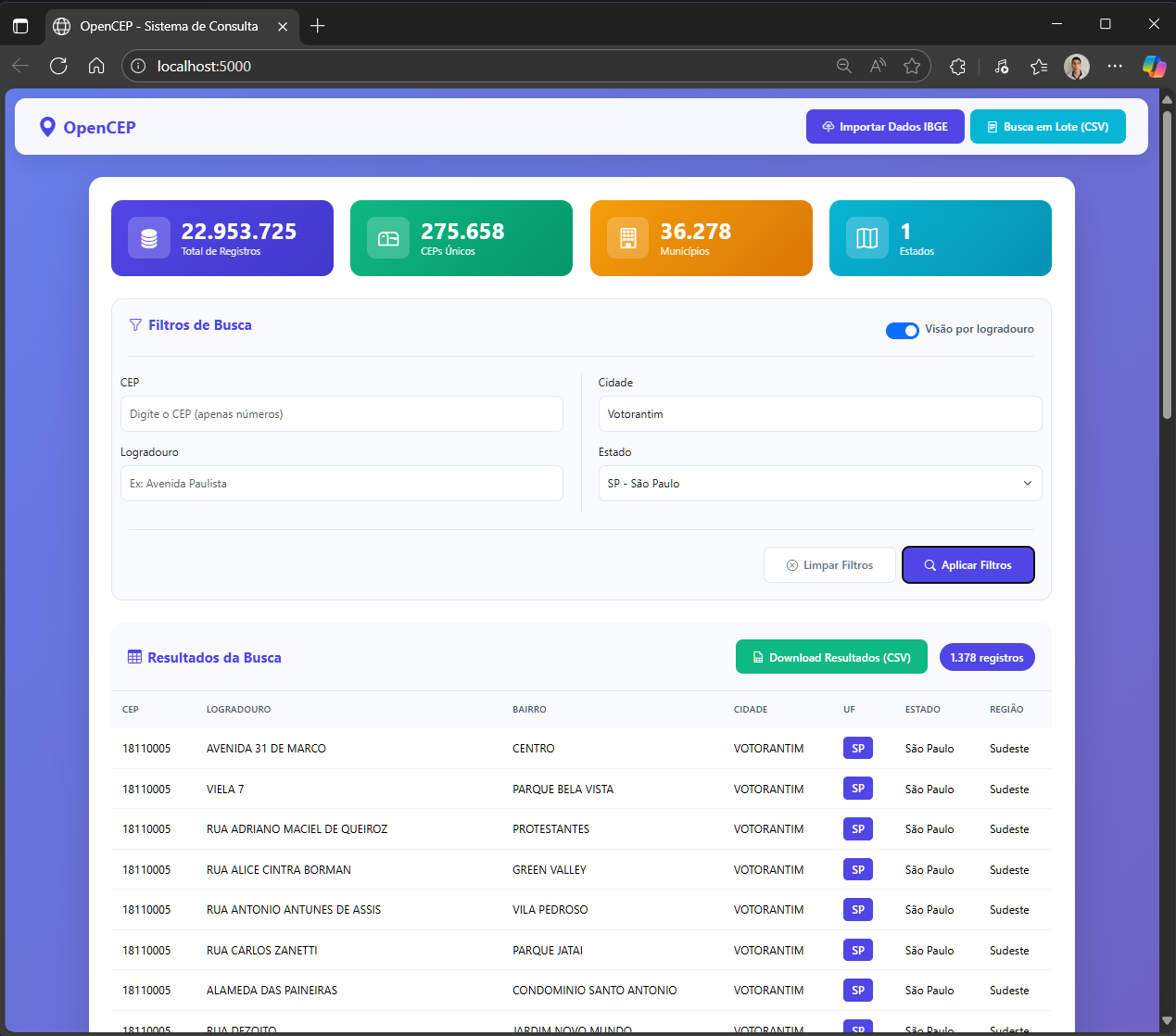


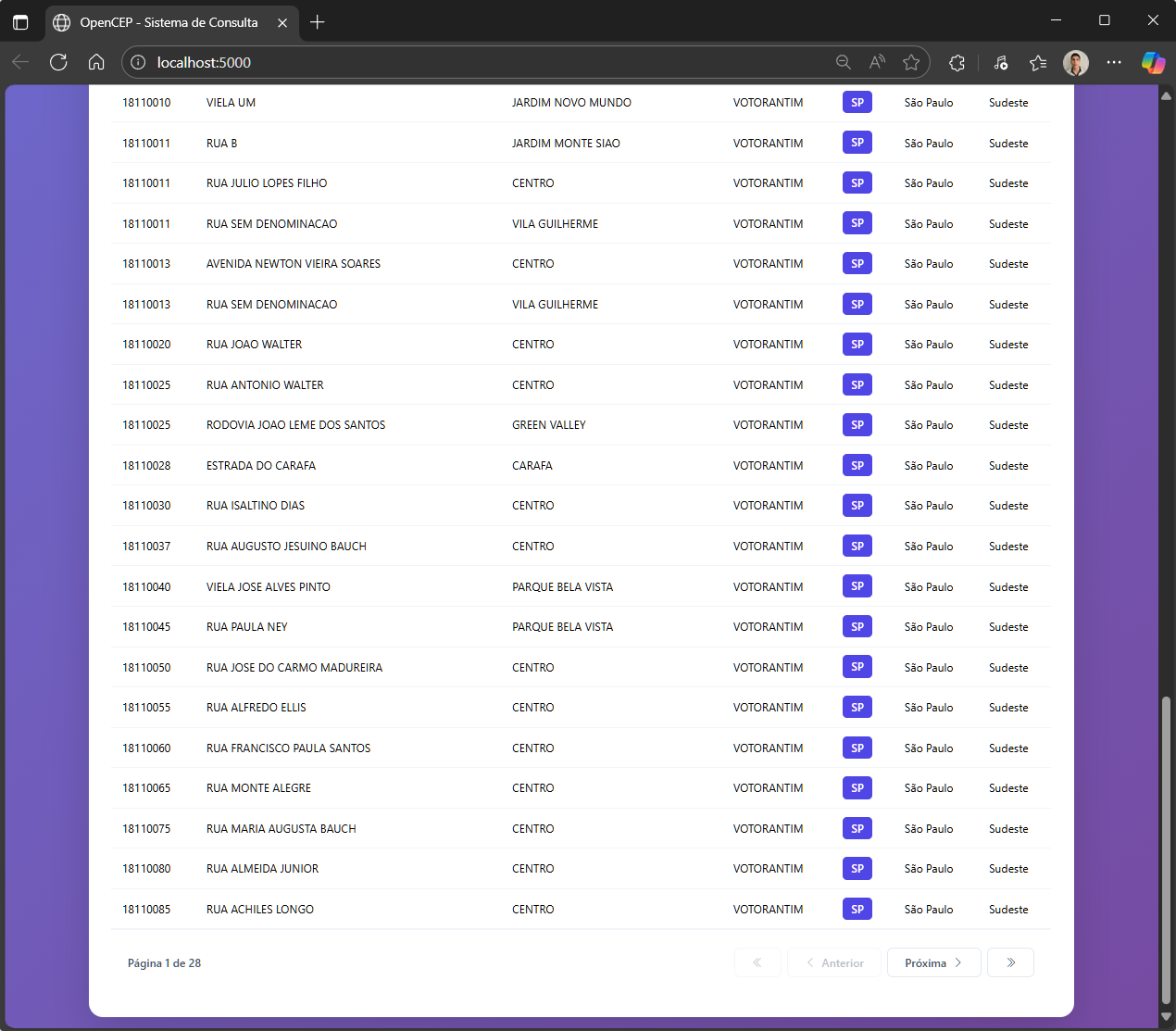
* **Evidência 3 — Consulta Individual de CEP**  
  Print do **resultado de uma consulta individual** na interface web, demonstrando a busca por um CEP válido (ex: 18110008) e o retorno correto dos dados de endereço.



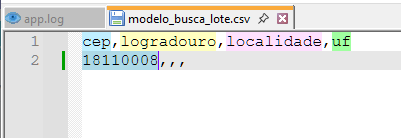
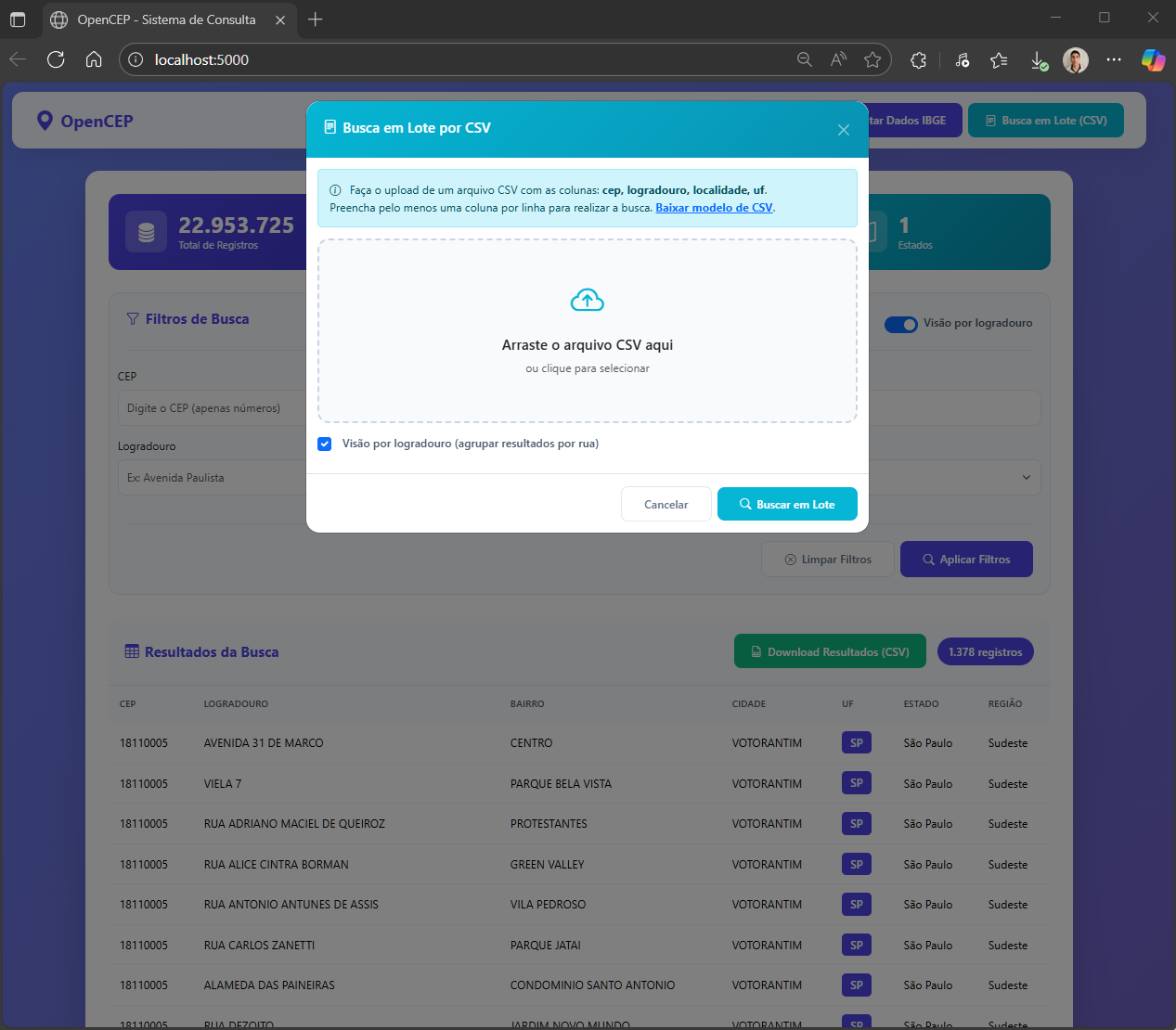


* **Evidência 4 — Consulta Reversa (UF + Cidade)**  
  Print da **consulta reversa** realizada na interface web, utilizando os filtros de **UF = “SP”** e **Cidade = “São Paulo”**, exibindo a lista de endereços correspondentes.

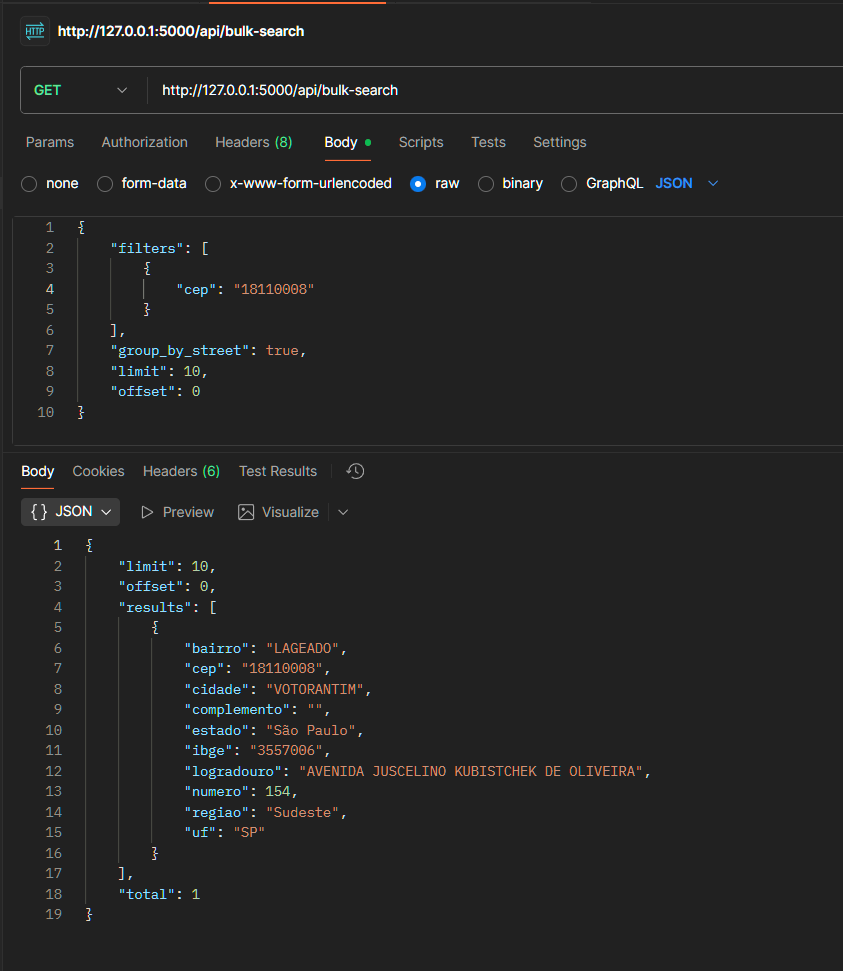


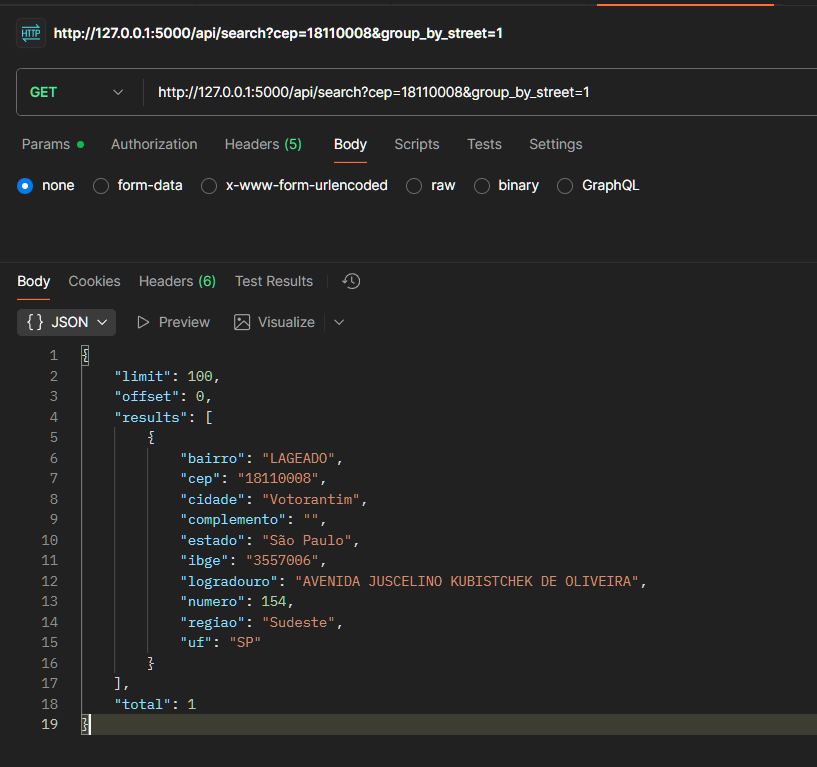


* **Evidência 5 — Consulta em Lote e Exportação**  
  Print demonstrando o **processo completo de consulta em lote**, incluindo:
  + Upload de um arquivo modelo\_busca\_lote.csv;
  + Processamento da busca;
  + Download automático do arquivo de resultados export\_open\_cep.csv.

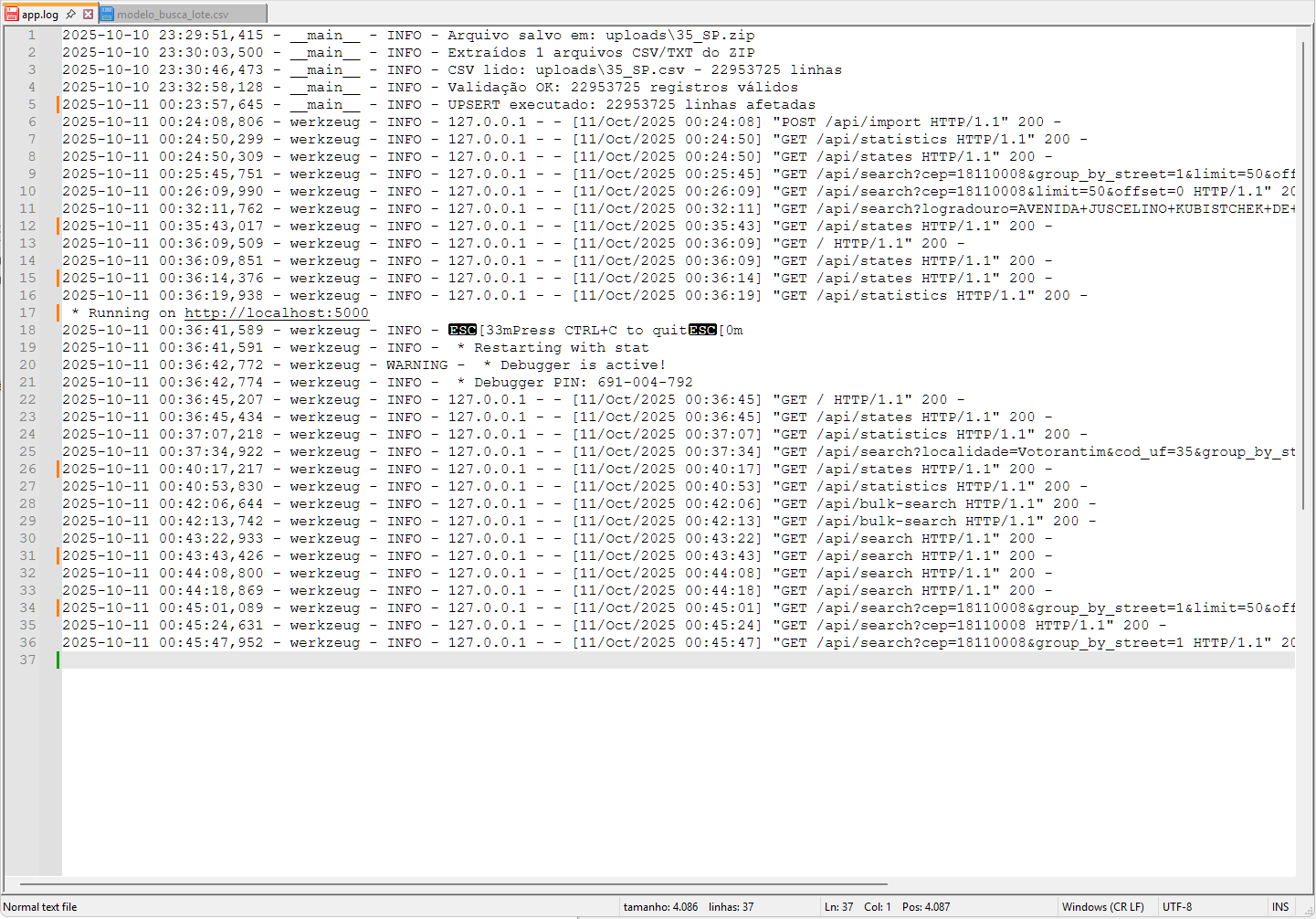


* **Evidência 6 — Testes de API (GET e POST)**  
  Prints do **Postman** (ou ferramenta similar) demonstrando:
  + Requisição **GET** para o endpoint /api/search/18110008, com resposta **HTTP 200 OK** e corpo JSON contendo o endereço.
  + Requisição **POST** para o endpoint /api/bulk-search, com resposta **HTTP 200 OK** e retorno JSON contendo a lista de endereços correspondentes aos filtros enviados.





* Evidência 7 **—** Arquivo app.log escrevendo o log do sistema



## 8. Aprovação da Validação

**Responsável da Empresa Parceira:** Lukas Reis de Oliveira

**Assinatura:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Data:** 10/10/2025