SOFTEX PERNAMBUCO
FORMAÇÃO ACELERADA EM PROGRAMAÇÃO
BACKEND - PYTHON
TURMA 17

RELATÓRIO DE PROJETO INTEGRADOR WEB SCRAPING

Jordan Lima - UFRN - jordanflima01@gmail.com
Leonardo Jerônimo - UFRN - leonardomoraisjeronimo@gmail.com
Hugo Da Costa Gomes - UnP - thunderbite2011@gmail.com
Arthur Santana - UFRN - arthursantana304@gmail.com

NOVEMBRO– 2024
NATAL- RN

Jordan Lima - UFRN
Leonardo Jerônimo - UFRN
Hugo Da Costa Gomes - UnP
Arthur Santana - UFRN

RELATÓRIO DE PROJETO INTEGRADOR WEB SCRAPING

Relatório de projeto integrador apresentado a coordenação do curso de Formação Acelerada em Programação (FAP), curso apresentado por Softex Pernambuco sobre tutela do Prof. Dr. Jose Alfredo F. Costa na Universidade Federal do Rio Grande do Norte em cumprimento às exigências para a obtenção do certificado de conclusão do curso.

Orientador: Prof. Dr. Jose Alfredo F. Costa

NOVEMBRO– 2024

Jordan Lima - UFRN
Leonardo Jerônimo - UFRN
Hugo Da Costa Gomes - UnP
Arthur Santana - UFRN

RELATÓRIO DE PROJETO INTEGRADOR WEB SCRAPING

Natal/RN
Novembro – 2024

SUMÁRIO

1.0. INTRODUÇÃO

O mercado imobiliário em Natal, como em muitas cidades, apresenta uma dinâmica complexa, com informações dispersas em diversos portais online e uma potencial discrepância entre os valores declarados e os valores de mercado. A falta de uma fonte centralizada e atualizada de dados dificulta a análise de mercado, a tomada de decisões estratégicas e a fiscalização tributária. Este sistema de web crawler de imóveis surge como uma solução para esse problema, oferecendo um meio automatizado e eficiente para coletar, processar e disponibilizar informações relevantes do mercado imobiliário local. O sistema coleta dados de múltiplos portais imobiliários, padroniza as informações, armazena-as em um banco de dados robusto e disponibiliza uma interface web para acesso e visualização. A escolha do Supabase como provedor do banco de dados PostgreSQL garante escalabilidade e flexibilidade para o crescimento futuro do sistema. O foco inicial está na coleta e armazenamento dos dados, servindo como base para análises e integrações mais complexas em etapas posteriores do projeto.

2.0. OBJETIVO

O objetivo principal deste projeto é desenvolver um sistema completo e escalável para a coleta e análise de dados do mercado imobiliário de Natal. Este sistema deve fornecer:

- **Coleta Automatizada de Dados**: Extrair informações relevantes de imóveis em diferentes portais imobiliários de forma automatizada e eficiente, utilizando técnicas de web scraping robustas e adaptáveis.

- **Processamento e Padronização de Dados**: Limpar, validar e padronizar os dados extraídos, garantindo a consistência e a qualidade da informação. Isso inclui a conversão de formatos, a identificação e tratamento de valores faltantes ou inconsistentes, e a remoção de duplicatas.

- **Armazenamento em Banco de Dados**: Armazenar os dados processados em um banco de dados PostgreSQL, garantindo a integridade, a segurança e a disponibilidade da informação. A utilização do Supabase como provedor simplifica a implantação e a manutenção do banco de dados.

- **Interface Web Intuitiva**: Disponibilizar uma interface web amigável e eficiente para a visualização e análise dos dados, permitindo a navegação intuitiva pelos resultados e a utilização de funcionalidades como

paginação.

- **API REST**: Criar uma API REST que permita o acesso programático aos dados coletados, facilitando a integração com outras ferramentas e sistemas, incluindo a possibilidade de integração futura com sistemas de fiscalização tributária.

Em resumo, o sistema visa fornecer uma base sólida e confiável de dados para a análise de mercado imobiliário, subsidiando decisões estratégicas e o combate à sonegação fiscal. A atual implementação concentra-se nos objetivos de coleta e armazenamento de dados, mas futuras iterações focarão na implementação dos recursos de análise e integração.

3.0. ARQUITETURA DO SISTEMA

O sistema é composto por três módulos principais que interagem entre si:

Módulo de Web Scraping: Este módulo é responsável pela coleta de dados de imóveis a partir de três portais imobiliários diferentes: Abreu Imóveis, Caio Fernandes e MGF Imóveis. Ele utiliza a biblioteca Selenium para simular a interação de um usuário com o navegador web (permitindo lidar com JavaScript e conteúdo dinâmico) e a biblioteca BeautifulSoup para extrair os dados relevantes do código HTML retornado pelo servidor. Cada portal possui um script individual (main_*.py) devido às diferenças nas estruturas HTML. Cada script utiliza XPath e/ou seletores CSS para identificar os elementos de interesse na página. Os dados brutos são então tratados e formatados para garantir consistência.

Módulo de Backend (Django): Este módulo é o coração do sistema, sendo responsável pelo processamento, armazenamento e exposição dos dados via API REST. Ele é construído utilizando o framework Django. As principais funcionalidades incluem:

- **Recebimento de Dados**: Recebe os dados tratados e formatados do módulo de Web Scraping.

- **Validação e Limpeza de Dados**: Realiza validações para garantir a integridade e consistência dos dados recebidos, limpando-os de possíveis erros ou inconsistências.
- **Armazenamento de Dados**: Armazena os dados validados e limpos em um banco de dados PostgreSQL utilizando o ORM do

Django. A função update_or_create garante a atualização eficiente de registros existentes e a criação de novos registros. Inclui a lógica para remover imóveis que não foram encontrados em novas iterações de scraping.

- **API REST**: Expõe os dados através de uma API REST desenvolvida com o Django REST Framework. A API fornece endpoints para acesso aos dados, permitindo a consulta, inclusão, atualização e deleção de registros de imóveis.

- **Gerenciamento de Sessões**: Apesar de não implementada, é necessário considerar a adição de um sistema de autenticação e gerenciamento de sessões para a API.

Módulo de Frontend (Django): Este módulo é responsável pela interface com o usuário, permitindo a visualização dos dados coletados. Construído com Django, ele utiliza templates para renderizar as páginas HTML. A interface atual é básica e fornece apenas paginação para navegar pelas listas de imóveis. Funcionalidades adicionais, como filtros, busca e ordenação, ainda precisam ser implementadas.

3.1.ESPECIFICAÇÕES TÉCNICAS

Linguagens de Programação: Python (backend e scraping), HTML, CSS, JavaScript (frontend).

Frameworks: Django (backend e frontend), Django REST Framework (API REST).

Bibliotecas: Selenium, BeautifulSoup (scraping), requests (possivelmente, dependendo da implementação), Django ORM (para interação com o banco de dados).

Banco de Dados: PostgreSQL (atualmente hospedado no Supabase, mas com configuração facilmente alterável para outros servidores PostgreSQL).

Sistema Operacional: Independente (Python é multiplataforma, o Django também).

Ambiente de Desenvolvimento: Recomendado o uso de um ambiente virtual (venv ou conda) para gerenciar as dependências.

Controle de Versão: Git

3.2.MODELO DE DADOS

O modelo de dados é baseado em dois modelos Django: Imovel e Imovel_Caio. Eles foram definidos separadamente devido a diferenças no formato e conteúdo dos dados coletados de cada site. Considerando a similaridade, a unificação em um modelo único com campos opcionais é uma melhoria que poderia ser implementada.

Modelo Imovel: Este modelo representa os dados coletados do site Abreu Imóveis, e possivelmente outros sites, no futuro:

- **id**: Campo inteiro auto-incremental, chave primária.
- **imovel_tipo**: Campo texto (CharField) que representa o tipo de imóvel (casa, apartamento, etc.).
- **imovel_codigo**: Campo inteiro (IntegerField) representando o código único do imóvel no site. Pode ser nulo.
- **imovel_m2**: Campo inteiro (IntegerField) representando a área do imóvel em metros quadrados. Pode ser nulo.
- **imovel_endereco**: Campo texto (CharField) com o endereço completo do imóvel.
- **imovel_valor**: Campo numérico (FloatField) representando o valor do imóvel.
- **imovel_rua**, **imovel_bairro**, **imovel_cidade**: Campos adicionados para melhorar a organização e granularidade dos dados de endereço. Podem ser nulos.
- **imovel_site**: Campo texto para indicar o site de origem. Pode ser nulo.
- **created_at**: Timestamp de criação do registro.
- **updated_at**: Timestamp de atualização do registro.

3.3. API REST

A API REST expõe os dados do modelo **Imovel** via endpoints CRUD (Create, Read, Update, Delete). Utiliza o Django REST Framework e a serialização é definida em ImovelSerializer. A autenticação e autorização ainda não estão implementadas, mas são cruciais para versões futuras.

3.4. CÓDIGO-FONTE

https://github.com/Grupo-dos-casas-Softex-FAP

4.0. CONSIDERAÇÕES FINAIS / CONCLUSÕES

Esta versão estabelece uma base sólida para a coleta e armazenamento de dados. Melhorias significativas são planejadas para futuras iterações:

Primárias:

- **Melhoria da Robustez do Web Scraping**: Implementar mecanismos mais robustos para lidar com mudanças nos sites imobiliários e tratamento de erros.

- **Integração com IA**: Implementar algoritmos de IA para identificar possíveis inconsistências entre os valores de mercado e os valores declarados para a fiscalização.

- **Otimização do Banco de Dados**: Refatoração do modelo de dados para melhor normalização e otimização do desempenho do banco de dados.

Secundárias:

- **Dashboards e Visualização**: Implementar dashboards interativos para melhor visualização dos dados.
- **Funcionalidades Avançadas na Interface Web**: Adicionar recursos de filtro, busca e ordenação avançados à interface web.
- **Integração com Mais Sites**: Expandir a coleta para abranger mais portais imobiliários.
- **Alertas e Notificações**: Implementar sistema de alertas para notificar usuários sobre atualizações importantes.
- **Segurança na API**: Implementar autenticação e autorização robustas na API REST.

NOVEMBRO– 2024
NATAL- RN