Automated Supplier Data Extraction Guide

For the Replit Development Agent

Version: 1.0
Date: 2025-07-24
Author: Replit Coach Too
Status: Draft

1. Objective

This document provides a detailed technical guide for the Replit Agent to build an **automated data extraction feature** for the supplier onboarding process. This feature will allow suppliers to provide a URL to a product page, from which the system will attempt to scrape and pre-fill the required product attribute data, significantly reducing manual entry and improving the user experience.

2. Part 1: Technology & Backend Enhancements

2.1. New Technology: Web Scraping Libraries

- **Requirement:** The Python backend must incorporate libraries designed for web scraping.
- **Technology:**
  - **requests:** To fetch the HTML content of the provided URL.
  - **BeautifulSoup4:** To parse the HTML content and make it easy to search and extract data from.

2.2. New Backend Service: WebScrapingService

- **Logic:** A new service will be created in the Flask backend to encapsulate the scraping logic.
- **Functionality:** This service will have a primary function, scrape_product_data(url), which will:
  1. Take a URL as input.
  2. Use requests to get the page's HTML.
  3. Use BeautifulSoup to parse the HTML.
  4. Search the parsed HTML for common keywords and patterns associated with product specifications (e.g., "Material:", "Weight:", "Recycled Content:", "g", "kg", "%").
  5. Attempt to extract the values associated with these keywords.
  6. Return a JSON object containing the extracted, structured data (e.g., {"material_type": "Glass", "weight_grams": 540}).

3. Part 2: Updated Supplier Onboarding Flow

This new feature will be integrated directly into the **Supplier Onboarding Portal** (Workflow 2 from the supplier-management-guide).

3.1. New User Interface Component

- **Location:** On the "Product Data" step of the supplier onboarding form.
- **Interface:** Before the manual entry form for a new product, a new section will appear:
  - **Headline:** "Want to speed things up? Let us try to import your product data automatically."
  - **Component:** URL_Input_Field with a button labeled **"Import from URL"**.

3.2. The Automated Onboarding Workflow

1. **Supplier Action:** The supplier enters the URL of their product page (e.g., https://supplier-x.com/products/700ml-flint-bottle) and clicks "Import from URL".
2. **Frontend Action:** The frontend displays a loading indicator and sends an asynchronous API request to a new backend endpoint, passing the URL.
3. **Backend Action:**
   - The new endpoint (POST /api/suppliers/scrape-product) is called.
   - It invokes the WebScrapingService to perform the data extraction.
   - The service returns the extracted data as a JSON object.
4. **Frontend Action:**
   - The loading indicator is removed.
   - The manual entry form for the product is now **pre-filled** with the data that was successfully extracted.
   - **Crucially:** The user is prompted to review the imported data. A message will appear: **"We've imported the following data. Please review it for accuracy and fill in any missing fields."**
5. **Supplier Action:** The supplier reviews the pre-filled form, corrects any errors, fills in any data the scraper missed, and then submits the form as usual.

4. Part 3: Backend Logic & Considerations

- **New API Endpoint: POST /api/suppliers/scrape-product**
  - **Request Body:** { "url": "..." }
  - **Response Body (Success):** { "extracted_data": { "material_type": "Glass", "weight_grams": 540, ... } }
  - **Response Body (Failure):** { "error": "Could not retrieve or parse data from the provided URL." }
- **Robustness & Error Handling:**
  - The WebScrapingService must be built defensively. It needs to handle cases

where a website is down, blocks scraping attempts, or has an unusual HTML structure.
    - The scraper should not be expected to be 100% accurate. It is a "best-effort" tool designed to assist the user, not to replace them entirely. The user must always have the final say in reviewing and confirming the data.
- **Security:**
    - The backend must validate all incoming URLs to ensure they are legitimate HTTP/HTTPS URLs to prevent potential security risks.

This feature, when implemented, will dramatically improve the efficiency and user-friendliness of the supplier onboarding process, making it a key differentiator for the platform.