Web Scraper Tool Documentation

# Overview

A comprehensive web scraping tool designed to extract company information from websites with a user-friendly interface and multiple export formats.

# Features Implemented

## Core Features (Minimal Requirements)

* ✅ Input Handling: Accept multiple URLs for scraping
* ✅ Basic Data Extraction: Extract company name, website URL, and contact information
* ✅ Error Handling: Graceful handling of network errors and missing data
* ✅ Structured Output: Export data in JSON, CSV, and Excel formats

## Enhanced Features (Optional Requirements)

* ✅ Extended Contact Information: Social media profiles, phone numbers, email addresses
* ✅ Company Overview: Description, keywords, and address extraction
* ✅ Data Cleaning: Text normalization and validation
* ✅ User Interface: Modern web interface with real-time progress tracking
* ✅ Multiple Export Formats: JSON, CSV, and Excel with professional formatting
* ✅ Statistics Dashboard: Success rates and scraping analytics

# Technology Stack

* Backend: Python Flask
* Frontend: HTML, CSS (Tailwind CSS), JavaScript
* Libraries:
* - requests - HTTP requests
* - beautifulsoup4 - HTML parsing
* - openpyxl - Excel file generation
* - selenium - Dynamic content handling (optional)

# Usage Instructions

## Web Interface

1. Enter one or more website URLs (one per line)
2. Click 'Start Scraping' to begin extraction
3. Results will appear below with detailed company information
4. Use the export buttons to download data in JSON, CSV, or Excel format

# API Endpoints

POST /api/scrape  
Content-Type: application/json  
{  
 "urls": ["https://example.com"]  
}

POST /api/export  
Content-Type: application/json  
{  
 "results": [...],  
 "format": "json|csv|excel"  
}

POST /api/analyze  
Content-Type: application/json  
{  
 "results": [...]  
}

# Data Extraction Levels

## Level 1 - Basic (Implemented)

* Company name
* Website URL
* Basic contact information (emails, phone numbers)

## Level 2 - Enhanced (Implemented)

* Social media profiles
* Company description/tagline
* Keywords and industry information
* Physical address

## Level 3 - Advanced (Partially Implemented)

* Data cleaning and normalization
* Error handling and validation
* Export capabilities
* Analytics and statistics

# Sample Output

Example JSON Output:

{  
 "results": [  
 {  
 "company\_name": "GitHub",  
 "website\_url": "https://github.com",  
 "emails": [],  
 "phones": [],  
 "description": "GitHub is where people build software...",  
 "keywords": "N/A",  
 "address": "N/A",  
 "social\_links": [  
 "https://www.linkedin.com/company/github",  
 "https://www.instagram.com/github"  
 ],  
 "status": "success",  
 "scraped\_at": "2025-07-02 10:42:30"  
 }  
 ]  
}

# Design Decisions and Assumptions

## Architecture Decisions

* Flask Backend for rapid development
* Modular Design
* RESTful API
* Responsive UI with Tailwind CSS

## Data Extraction Strategy

* Multiple Selectors
* Fallback Methods
* Text Cleaning
* Rate Limiting

## Error Handling

* Graceful Degradation
* Detailed Error Messages
* Timeout Handling
* Input Validation

## Assumptions Made

* Standard HTML structure expected
* Regex used for contact extraction
* Focus on major social platforms
* Optimized for 1-50 URLs per session

# Testing

* ✅ Basic URL input and validation
* ✅ Successful data extraction from major websites
* ✅ Error handling for invalid URLs
* ✅ Export functionality in all formats
* ✅ User interface responsiveness

## ✅ Output Screenshotsample_output