EPC Extraction Rewrite v4.0 - Structured HTML Approach

Date: December 10, 2025
Branch: `fix/epc-rewrite-v4-structured-html`
Commit: 0dad17d

Overview

Complete rewrite of EPC certificate extraction using reliable structured HTML parsing instead of complex SVG/fallback approaches. This implementation is simpler, more reliable, and includes intelligent certificate matching when multiple properties share the same house number.

Problem Statement

Previous EPC extraction implementations had several issues:
1. **Complex SVG parsing** with multiple fallback methods
2. **Incorrect certificate matching** when multiple properties had the same house number (e.g., “317 Wharf Road” and “Spen Lea, 317 Wharf Road”)
3. **Difficulty extracting house numbers** from addresses with property names
4. **No tie-breaking logic** when multiple valid certificates exist

Real-World Example

For postcode DN17 4JW, there are TWO different properties at 317 Wharf Road:
- **Certificate 2068-4069-6258-6561-6084**: “317, Wharf Road” - Rating F, 228 sqm
- **Certificate 0310-2606-8090-2399-6161**: “Spen Lea, 317 Wharf Road” - Rating E, 226 sqm

When searching for “317 Wharf Road”, both match the address. The algorithm needs to pick the correct one.

Solution Architecture

1. Structured HTML Parsing

New Function: `scrapeCertificateData(certificateURL)`

Directly parses reliable HTML elements from EPC certificate pages:

```
// Address from structured element
const addressElem = $('.epc-address.govuk-body, p.epc-address');
// Replace <br> with commas
certificateAddress = addressHtml.replace(/<br\s*\/?>/gi, ', ');

// Rating from plain text element (not SVG!)
const ratingElem = $('.epc-rating-result.govuk-body, p.epc-rating-result');
rating = ratingElem.text().trim().toUpperCase(); // "E", "D", etc.

// Floor area from summary list
$('dt').each((i, elem) => {
    if (label.includes('total floor area')) {
        const match = valueText.match(/(\d+(?:\.\d+)?)\s*square metres?/i);
        floorArea = parseFloat(match[1]); // 226
    }
});
```

Benefits:
- ✅ Simple and direct
- ✅ No complex SVG parsing
- ✅ No multiple fallback methods
- ✅ Extracts all three data points in one pass

---

2. Enhanced House Number Extraction

Enhanced Function: `extractHouseNumber(address)`

Now handles property names that appear before house numbers:

```
// NEW: Pattern for "Property Name, 317 Street"
const propertyNamePattern = /^[a-z\s]+,\s*(\d+[a-z]?)\b/i;

// Examples that now work:
extractHouseNumber("Spen Lea, 317 Wharf Road")       // {primary: "317"}
extractHouseNumber("Akland House, 303 Street")       // {primary: "303"}
extractHouseNumber("Flat 2a, 32 Street")             // {primary: "32", flat: "2a"}
extractHouseNumber("32a Street")                      // {primary: "32", flat: "a"}
```

Supported Patterns:
1. Flat/apartment + number: “Flat X, 32 Street”
2. Property name + number: “Spen Lea, 317 Street” ⭐ NEW
3. Letter suffix: “32a Street”
4. Number range: “32-34 Street”
5. Simple number: “32 Street”
6. Comma-separated: “Name, Name, 32 Street”

---

3. Multi-Certificate Collection & Matching

Rewritten Function: `getCertificateNumber(postcode, address, apiKey, knownFloorArea)`

NEW APPROACH:
1. Scrape all certificate numbers from postcode search page

2. For EACH certificate, fetch full data from certificate page (address, rating, floor area)
3. Extract house numbers and verify exact match
4. Verify street name similarity (≥30%)
5. Collect ALL valid matches (not just first one!)
6. Use intelligent tie-breaking when multiple matches exist

Tie-Breaking Logic:

When multiple certificates match (same house number + street):

```
// PRIORITY 1: Floor area matching (if known)
if (knownFloorArea && certData.floorArea) {
    if (floorDiff === 0) score += 10;        // Exact match
    else if (floorDiff <= 2) score += 5;     // Within 2 sqm
    else if (floorDiff <= 5) score += 2;     // Within 5 sqm
}

// PRIORITY 2: Prefer addresses without property names
if (!hasPropertyName) {
    score += 0.5;  // Slight preference
}
```

Example Decision:
- Property search: “317 Wharf Road”, known floor area: 226 sqm
- Candidate 1: “317, Wharf Road” - 228 sqm → Score: 1.0 (street) + 0.5 (no name) = **1.5**
- Candidate 2: “Spen Lea, 317 Wharf Road” - 226 sqm → Score: 1.0 (street) + 10.0 (exact floor) = **11.0** ✅

Result: **Candidate 2 selected** (exact floor area match is very strong signal)

---

4. Integration with Main Workflow

Updated: `main.js` - EPC enrichment section

```
// Extract known floor area from property data (if available)
const knownFloorArea = property.Sqm ? parseFloat(property.Sqm) : null;

// Pass to scraping function for better matching
const epcData = await scrapeEPCData(
    property.Postcode,
    property.Address,
    apiKey,
    knownFloorArea  // ⭐ NEW parameter
);
```

Use Cases:
- **First-time processing:** No floor area known, uses address matching only
- **Re-processing:** Floor area available from CSV, uses for tie-breaking
- **Update runs:** Existing floor area helps pick correct certificate

---

Test Results

Test 1: Certificate Data Extraction ✅

```
Certificate: 0310-2606-8090-2399-6161
Expected: Address with "317" and "Wharf Road", Rating E, Floor Area 226 sqm

RESULTS:
  Address: "Spen Lea, 317 Wharf Road, Ealand, Scunthorpe, DN17 4JW" ✅
  Rating: E ✅
  Floor Area: 226 sqm ✅

✅ TEST 1 PASSED: All data extracted correctly
```

Test 2: Address Matching with Floor Area ✅

```
Property: "317 Wharf Road, Ealand, Scunthorpe"
Known Floor Area: 226 sqm
Expected: Certificate 0310-2606-8090-2399-6161 (Spen Lea) with rating E

MULTIPLE MATCHES FOUND:
  1. 2068-4069-6258-6561-6084
     Address: "317, Wharf Road, Ealand, SCUNTHORPE, DN17 4JW"
     Rating: F, Floor Area: 228 sqm
     Street Similarity: 100.0%

  2. 0310-2606-8090-2399-6161
     Address: "Spen Lea, 317 Wharf Road, Ealand, Scunthorpe, DN17 4JW"
     Rating: E, Floor Area: 226 sqm
     Street Similarity: 100.0%

SELECTED BEST MATCH:
  Certificate: 0310-2606-8090-2399-6161 ✅
  Rating: E ✅
  Floor Area: 226 sqm ✅
  Selection Reason: exact floor area match

✅ TEST 2 PASSED: Correct certificate matched (Spen Lea)
```

Test 3: Non-Existent Address Returns NULL ✅

```
Property: "307 Wharf Road, Ealand, Scunthorpe"
Expected: NULL (no certificate exists for this address)

Checked 12 certificates:
  - 303 Wharf Road: Different house number ❌
  - 305 Wharf Road: Different house number ❌
  - 309 Wharf Road: Different house number ❌
  - 310 Wharf Road: Different house number ❌
  - 317 Wharf Road (×2): Different house number ❌

RESULTS: NULL ✅

✅ TEST 3 PASSED: Correctly returned NULL (no certificate exists)
```

File Changes

Modified Files

1. **`src/utils/epcHandler.js`** (+198/-110 lines)
   - New `scrapeCertificateData()` function
   - Enhanced `extractHouseNumber()` with 6 patterns
   - Rewritten `getCertificateNumber()` with multi-match collection
   - Updated `fetchEPCDataViaAPI()` to accept floor area
   - Updated `scrapeEPCData()` to accept floor area

2. **`src/main.js`** (+2 lines)
   - Extract Sqm from property data
   - Pass to scrapeEPCData for matching

3. **`test-epc-rewrite-v4.js`** (NEW, 116 lines)
   - Test certificate data extraction
   - Test address matching with floor area
   - Test non-existent address handling

Technical Improvements

Reliability

- ✅ Uses structured HTML elements (not SVG parsing)
- ✅ Single data extraction path (no complex fallbacks)
- ✅ Comprehensive address verification

Accuracy

- ✅ Exact house number matching required
- ✅ Street name similarity verification (≥30%)
- ✅ Floor area tie-breaking for ambiguous cases
- ✅ Transparent match status reporting

Debugging

- ✅ Detailed logging for each certificate checked
- ✅ House number extraction logged
- ✅ Match decision reasoning logged
- ✅ All candidates shown when multiple matches exist

Performance

- ✅ Scrapes all certificates in postcode (necessary for accuracy)
- ✅ Early exit when exact match found
- ✅ Efficient cheerio parsing

Edge Cases Handled

1. **Multiple properties at same house number**
   - Uses floor area tie-breaking
   - Logs all candidates for transparency
   - Returns best match with reasoning

2. **Property names before house number**
   - "Spen Lea, 317 Street" → extracts 317
   - "Akland House, 303 Street" → extracts 303

3. **Properties without house numbers**
   - "Akland House, Street" → returns null for primary
   - Logged as "missing_house_number"

4. **No certificate exists**
   - Returns NULL (better no data than wrong data)
   - Logged with reason

5. **First-time vs re-processing**
   - First time: Uses address matching only
   - Re-processing: Uses floor area + address

Backward Compatibility

Maintained Functions

- `scrapeRatingFromCertificate()` - now calls `scrapeCertificateData()`
- `scrapeFloorAreaFromCertificate()` - now calls `scrapeCertificateData()`
- `scrapeEPCData()` - enhanced with floor area parameter (optional)

Module Exports

All existing exports maintained:

```
module.exports = {
    generateEPCSearchURL,
    scrapeEPCData,              // Enhanced
    createEPCLookupRow,
    getCertificateNumber,        // Rewritten
    scrapeCertificateData,       // NEW
    fetchEPCDataViaAPI,          // Enhanced
    // ... other functions
};
```

Deployment Instructions

1. Review Changes

```
cd /home/ubuntu/github_repos/soldcomp-analyser2
git diff main...fix/epc-rewrite-v4-structured-html
```

2. Run Tests

```
node test-epc-rewrite-v4.js
```

Expected output:

```
✅ TEST 1 PASSED: All data extracted correctly
✅ TEST 2 PASSED: Correct certificate matched (Spen Lea)
✅ TEST 3 PASSED: Correctly returned NULL
```

3. Test with Real CSV (Optional)

```
node src/main.js "/home/ubuntu/Uploads/data (5).csv"
```

Verify:
- 317 Wharf Road gets rating E
- 307 Wharf Road gets NULL or different certificate

4. Create Pull Request

```
git push origin fix/epc-rewrite-v4-structured-html
```

Then create PR on GitHub.

Performance Considerations

Scraping Load

- Each property requires 1 postcode search + N certificate fetches

- For DN17 4JW: 1 postcode search + 12 certificate fetches
- Timeout: 10 seconds per request
- User-Agent: Set to avoid blocking

Optimization Opportunities

1. Cache postcode search results (if processing multiple properties in same postcode)
2. Parallel certificate fetching (with rate limiting)
3. Store certificate data to avoid re-scraping

Future Enhancements

Potential Improvements

1. **Certificate caching** - Store scraped certificates in database
2. **Batch processing** - Process multiple properties in same postcode together
3. **Alternative data sources** - Fallback to other EPC sources
4. **Machine learning** - Train model to pick best certificate from ambiguous cases
5. **User feedback loop** - Allow users to correct matches, improve algorithm

---

Summary

This rewrite delivers:
- ✅ **Simpler code** (structured HTML parsing)
- ✅ **More reliable** (direct element extraction)
- ✅ **More accurate** (floor area tie-breaking)
- ✅ **Better debugging** (comprehensive logging)
- ✅ **Handles edge cases** (multiple matches, property names, missing data)

Key Innovation: Floor area tie-breaking for ambiguous cases, enabling accurate matching even when multiple properties share the same house number.

Status: ✅ Ready for PR
Test Coverage: ✅ 3/3 tests passing
Documentation: ✅ Complete