Testing Guide for Soldcomp-Analyser2

Sample Test Data

A sample CSV file is provided at `test/sample-data.csv` with:
- 1 target property (123 Main Street, SW1A 1AA)
- 4 comparable properties
- Complete data for testing ranking algorithm

Local Testing (Without Apify Platform)

Since this actor is designed for Apify platform, testing locally requires some modifications:

Option 1: Mock KVS for Local Testing

Create a test script that bypasses Apify KVS:

```
// test/localTest.js
const fs = require('fs');
const path = require('path');

// Mock Apify Actor for local testing
const mockActor = {
    main: (fn) => fn(),
    openKeyValueStore: async () => ({
        getValue: async (key) => {
            const filePath = path.join(__dirname, 'sample-data.csv');
            return fs.readFileSync(filePath, 'utf-8');
        },
        setValue: async (key, value) => {
            const outputPath = path.join(__dirname, 'output.csv');
            fs.writeFileSync(outputPath, value);
            console.log(`Output saved to: ${outputPath}`);
        }
    })
};

// Replace apify module
require.cache[require.resolve('apify')] = {
    exports: { Actor: mockActor, log: console }
};

// Set test environment variables
process.env.GOOGLE_API_KEY = 'your_test_google_api_key';
process.env.KV_STORE_NAME = 'test';
process.env.DATA_KEY = 'data.csv';
process.env.OUTPUT_KEY = 'output.csv';

// Run main script
require('../src/main');
```

Option 2: Test on Apify Platform

1. **Upload to Apify:**
   `bash`
   `apify push`

2. **Create test KVS:**
   - Go to Apify Console → Storage → Key-Value Stores
   - Create store: `clive.caseley/soldcomp-analyser-kvs`
   - Upload `test/sample-data.csv` as `data.csv`

3. **Set environment variables:**
   - `GOOGLE_API_KEY`: Your Google API key
   - Leave other variables as defaults

4. **Run actor:**
   - Click “Start” in Apify Console
   - Monitor logs for progress
   - Check output in KVS under `output.csv`

Testing Checklist

Core Functionality

- [ ] CSV parsing with header detection
- [ ] Target property detection (exactly 1 target)
- [ ] Target validation (has postcode + address)
- [ ] URL classification (Rightmove, PropertyData)
- [ ] Geocoding and distance calculation
- [ ] Ranking algorithm
- [ ] Duplicate detection
- [ ] Excel HYPERLINK generation
- [ ] Output ordering (postcode searches → EPC row → target → ranked comparables)

Error Handling

- [ ] No target found → fatal error
- [ ] Multiple targets found → fatal error
- [ ] Target missing postcode → fatal error
- [ ] Target missing address → fatal error
- [ ] Scraping failure → flag with needs_review=1
- [ ] Geocoding failure → flag with needs_review=1

Edge Cases

- [ ] Missing headers in CSV
- [ ] Inconsistent column order
- [ ] Missing data fields (should still rank with 0 score)
- [ ] Duplicate properties (should merge)
- [ ] Invalid URLs (should handle gracefully)

Manual Testing Steps

1. Test Target Detection

Test Case: Valid target

```
Address,Postcode,isTarget
123 Main St,SW1A 1AA,1
456 Oak Ave,SW1A 1AB,
```

Expected: Actor completes successfully, target identified

Test Case: No target

```
Address,Postcode
123 Main St,SW1A 1AA
456 Oak Ave,SW1A 1AB
```

Expected: Fatal error - “No target property found”

Test Case: Multiple targets

```
Address,Postcode,isTarget
123 Main St,SW1A 1AA,1
456 Oak Ave,SW1A 1AB,target
```

Expected: Fatal error - “Multiple target properties found”

Test Case: Target missing postcode

```
Address,Postcode,isTarget
123 Main St,,1
456 Oak Ave,SW1A 1AB,
```

Expected: Fatal error - “Target missing required fields”

2. Test Ranking Algorithm

Use sample-data.csv and verify:
- Properties with similar floor area score higher
- Closer properties score higher
- Exact bedroom match gets 100 points, ±1 gets 50 points
- Recent sales score higher
- Missing data results in 0 score for that criterion

3. Test Scraping (Manual Verification)

Create a CSV with real URLs:

```
URL,isTarget
https://www.rightmove.co.uk/properties/123456789,
https://propertydata.co.uk/property/address,
123 Main Street,SW1A 1AA,target
```

Verify:
- URLs are scraped
- Data is extracted (address, price, bedrooms, etc.)
- Rate limiting is applied (check logs for delays)
- Failures are flagged with needs_review=1

4. Test Iterative Processing

Run 1:
- Input: `data.csv` with partial data
- Output: `output.csv` with enriched data

Run 2:
- Rename `output.csv` → `data.csv`
- Run actor again
- Verify: No duplicates, target maintained, data enriched further

Expected Output Structure

For sample-data.csv, expect output with:

1. **No postcode search listings** (none in sample data)
2. **EPC lookup row:** Single row with EPC search URL
3. **Target property:** 123 Main Street, SW1A 1AA, isTarget=1
4. **Ranked comparables:** 4 properties sorted by ranking score

Sample output columns:

```
Date of sale | Address | Postcode | ... | Distance | Ranking | isTarget
             | EPC Lookup | SW1A 1AA | ... |          |         |
2023-05-15   | 123 Main Street | SW1A 1AA | ... | | | 1
2023-04-10   | 789 Park Lane | SW1A 1AC | ... | 0.3mi | 85 | 0
2023-06-20   | 456 Oak Avenue | SW1A 1AB | ... | 0.2mi | 78 | 0
...
```

Performance Expectations

- **CSV parsing:** < 5 seconds for typical files (< 500 rows)
- **Target detection:** < 1 second
- **Geocoding:** ~1-2 seconds per property (Google API)
- **Scraping:** 2-3 seconds per URL (rate limited)
- **Ranking:** < 1 second for typical datasets

Total runtime: Depends on number of URLs to scrape. Estimate:
- Base processing: ~10-20 seconds
- + (number of URLs × 2.5 seconds)
- + (number of properties × 1.5 seconds for geocoding)

Debugging Tips

1. **Check logs:** Apify provides detailed logs for each step
2. **Monitor KVS:** Check intermediate data in Key-Value Store

3. **Verify API keys:** Ensure Google API key is valid and has quota
4. **Test URLs manually:** Open URLs in browser to verify they're accessible
5. **Check CSV format:** Ensure proper CSV encoding (UTF-8)

Common Issues and Solutions

Issue	Cause	Solution
"No target property found"	Target not marked or mis-spelled	Check target marker in CSV
"Target missing required fields"	Missing postcode or address	Add missing data to target row
Geocoding failures	Invalid API key or quota ex-ceeded	Check Google Cloud Console
Scraping failures	Anti-bot measures or invalid URLs	Review needs_review column
No distance calculated	GOOGLE_API_KEY not set	Set environment variable
Duplicates appearing	Inconsistent address/post-code format	Normalize data in CSV

Ready for production testing on Apify platform!