

CRITICAL BUG FIX — PCBancard Proposal Generator

SUMMARY OF FAILURE

The proposal generator is **completely broken**. A test proposal for "Cowtown Corner" was generated with:

- All form fields filled out (merchant name, owner, address, phone, email, website, agent info)
- Two valid spreadsheets uploaded (Dual Pricing + Interchange Plus) with real data
- Website URL provided: <https://www.bobs mufflerservice.com/>

Result: The generated PDF shows \$0 everywhere, NaN%, \$N/A, no merchant details, no agent details, no logo, no side-by-side comparison. The pipeline is not passing data through.

ACTUAL DATA FROM UPLOADED SPREADSHEETS (Proof the data exists)

From Cowtown Corner Spreadsheets (Melody Fitzgerald):

Current State:

- Visa Volume: \$21,838.52 (203 transactions)
- MasterCard Volume: \$12,049.60 (97 transactions)
- Discover Volume: \$850.14 (9 transactions)
- **TOTAL VOLUME: \$34,738.26**
- **TOTAL TRANSACTIONS: 309**
- Current Rate: 0.80%
- Current Per-Txn Fee: \$0.08
- Credit Pass-through: \$465.20

- Other/Misc Fees: \$43.00
- **CURRENT TOTAL FEES: \$810.83**

Dual Pricing Proposal:

- Proposed Rate: 0.00%
- Proposed Per-Txn: \$0.00
- Dual Pricing Monthly Fee: \$64.95
- **PROPOSED TOTAL: \$64.95**
- **Monthly Savings: \$745.88**
- **Annual Savings: \$8,950.51**
- Savings %: 246.47%

Interchange Plus Proposal:

- Proposed Rate: 0.60%
- Proposed Per-Txn: \$0.07
- On-File Fee: \$9.95
- Credit Pass-through: \$413.64
- **PROPOSED TOTAL: \$653.65**
- **Monthly Savings: \$157.18**
- **Annual Savings: \$1,886.12**
- Savings %: 51.94%

What the PDF Actually Shows:

Monthly Processing Volume: \$0
Monthly Transactions: 0
Average Ticket Size: \$0.00
Current Monthly Fees: \$0
Effective Rate: 0.00%
Annual Processing Cost: \$0

This proves the spreadsheet parsing pipeline is completely broken.

BUG #1: FORM DATA NOT PASSING TO GENERATOR

Symptoms:

- "Prepared For" shows only "Cowntown Corner" — no owner name, address, phone, email
- "Prepared By" shows "PCBancard Representative" — not the actual agent name
- Agent phone/email missing or truncated in Next Steps section

Root Cause Investigation:

The form collects data (screenshots confirm fields exist) but the data is NOT being passed to the proposal generation function.

FIX REQUIRED:

```
// CHECK 1: Is the form submission handler capturing all fields?
// Log the form data object before sending to generator
console.log('Form submission data:', JSON.stringify(formData, null, 2));

// CHECK 2: Is the API endpoint receiving the data?
// Add server-side logging
app.post('/api/generate-proposal', (req, res) => {
  console.log('Received request body:', JSON.stringify(req.body, null, 2));
  // ... rest of handler
});

// CHECK 3: Is the data being passed to Claude/template?
// Verify the prompt includes the actual values, not empty strings
```

Required Field Mapping (ensure these flow through):

```
// FROM FORM → TO PROPOSAL
const fieldMapping = {
  // Agent (REQUIRED)
  'agent_first_name': 'Prepared By: {value}',
  'agent_last_name': 'appended to first name',
  'agent_title': 'below agent name',
  'agent_phone': 'Direct line: {value}',
  'agent_email': 'Email: {value}',

  // Merchant (OPTIONAL but must render if provided)
  'business_name': 'Prepared For: {value}',
  'owner_name': 'Primary Contact: {value}',
```

```
'business_address': 'Address: {value}',
'business_phone': 'Phone: {value}',
'business_email': 'Email: {value}',
'business_website': 'Website: {value}',
};

// If a field is empty, HIDE the line (don't show blank placeholder)
// If agent name is empty, DO NOT default to "PCBancard Representative"
```

BUG #2: SPREADSHEET PARSING COMPLETELY BROKEN

Symptoms:

- All financial values show \$0
- Transaction counts show 0
- Effective rate shows 0.00%
- "NaN%" appears in savings calculation
- "\$N/A" appears in equipment section

Root Cause:

The parser is not extracting values from the uploaded PDFs/spreadsheets.

FIX REQUIRED:

Step 1: Verify files are being received

```
// Log file info
console.log('DP file:', dpFile?.name, dpFile?.size, 'bytes');
console.log('IC file:', icFile?.name, icFile?.size, 'bytes');
```

Step 2: Fix the extraction logic

The spreadsheets have this structure (based on actual PDFs):

```
Row: "VS Interchange" → Column: "Volume" = $21,838.52
Row: "MC Interchange" → Column: "Volume" = $12,049.60
Row: "Discover" → Column: "Volume" = $850.14
Row: "VS Item Fee" → Column: "Number of Transactions" = 203
Row: "TOTAL PROCESSING FEES:" → value = $810.83 (current) or $64.95 (proposed)
```

Row: "Estimated Monthly Processing Savings" → value = \$745.88
Row: "Estimated Yearly Processing Savings" → value = \$8,950.51

Step 3: Add extraction validation

```
function parseSpreadsheet(fileBuffer) {
  const extracted = {
    visa_volume: null,
    mc_volume: null,
    discover_volume: null,
    visa_transactions: null,
    mc_transactions: null,
    discover_transactions: null,
    current_total_fees: null,
    proposed_total_fees: null,
    monthly_savings: null,
    annual_savings: null,
  };

  // ... parsing logic ...

  // VALIDATION: Check if we got real data
  const totalVolume = (extracted.visa_volume || 0) +
    (extracted.mc_volume || 0) +
    (extracted.discover_volume || 0);

  if (totalVolume === 0) {
    console.error('EXTRACTION FAILED: Total volume is $0');
    console.error('Raw file content sample:', fileBuffer.slice(0, 500));
    throw new Error('Could not extract volume data from spreadsheet');
  }

  return extracted;
}
```

Step 4: Handle PDF parsing specifically

The uploaded files are PDFs, not Excel files. Ensure you're using a PDF parser:

```
const pdf = require('pdf-parse');

async function parsePricingPDF(fileBuffer) {
  const data = await pdf(fileBuffer);
  const text = data.text;

  // Extract values using regex patterns matching the PDF structure
```

```

const visaVolume = text.match(/VS Interchange.*?\$([0-9,]+\.\d{2})/)?.[1];
const mcVolume = text.match(/MC Interchange.*?\$([0-9,]+\.\d{2})/)?.[1];
// ... etc

console.log('Extracted from PDF:', { visaVolume, mcVolume });

return { /* normalized data */ };
}

```

BUG #3: WEBSITE SCRAPING NOT WORKING

Symptoms:

- No merchant logo appears on cover page
- Website URL was provided (<https://www.bobsmufflerservice.com/>) but not used

FIX REQUIRED:

```

const puppeteer = require('puppeteer');
// OR use cheerio for lighter scraping
const cheerio = require('cheerio');
const fetch = require('node-fetch');

async function scrapeBusinessLogo(websiteUrl) {
  if (!websiteUrl) return null;

  try {
    const response = await fetch(websiteUrl);
    const html = await response.text();
    const $ = cheerio.load(html);

    // Try multiple selectors in order of preference
    const logoSelectors = [
      'meta[property="og:image"]',
      'link[rel="icon"][sizes="192x192"]',
      'img[class*="logo"]',
      'img[id*="logo"]',
      'img[alt*="logo"]',
      'header img',
      '.logo img',
      '#logo img',
      'a[class*="brand"] img',
      'link[rel="apple-touch-icon"]',

```

```

        'link[rel="icon"]',
    ];

    for (const selector of logoSelectors) {
        let logoUrl = null;

        if (selector.startsWith('meta')) {
            logoUrl = $(selector).attr('content');
        } else if (selector.startsWith('link')) {
            logoUrl = $(selector).attr('href');
        } else {
            logoUrl = $(selector).first().attr('src');
        }

        if (logoUrl) {
            // Make absolute URL if relative
            if (logoUrl.startsWith('/')) {
                const base = new URL(websiteUrl);
                logoUrl = `${base.origin}${logoUrl}`;
            }

            console.log(`Found logo via ${selector}: ${logoUrl}`);
            return logoUrl;
        }
    }

    console.log('No logo found on website');
    return null;

} catch (error) {
    console.error('Logo scraping failed:', error.message);
    return null;
}
}

```

For Bob's Muffler Service specifically: The logo is visible on the homepage. The scraper should find it via `header img` or similar selector.

BUG #4: SIDE-BY-SIDE COMPARISON MISSING

Symptoms:

- Options Comparison table shows only "Current Processor: \$0"

- No Dual Pricing row
- No Interchange Plus row

FIX REQUIRED:

The comparison table MUST include all three options:

Option	Monthly Cost	Monthly Savings	Annual Savings
Current Processor	\$810.83	—	—
Dual Pricing	\$64.95	\$745.88	\$8,950.51
Interchange Plus	\$653.65	\$157.18	\$1,886.12

Ensure both spreadsheets are parsed and their results are included:

```
const comparisonData = {
  current: {
    monthly_cost: currentFees, // from either spreadsheet, "Current" column
  },
  dual_pricing: {
    monthly_cost: dpProposedTotal,
    monthly_savings: dpMonthlySavings,
    annual_savings: dpAnnualSavings,
  },
  interchange_plus: {
    monthly_cost: icpProposedTotal,
    monthly_savings: icpMonthlySavings,
    annual_savings: icpAnnualSavings,
  }
};
```

BUG #5: NaN AND \$N/A APPEARING

Symptoms:

- "NaN% reduction in fees" in the recommendation
- "\$N/A" in equipment section
- Divide-by-zero errors

FIX REQUIRED:


```

// Safe division helper
function safeDivide(numerator, denominator, fallback = null) {
  if (!denominator || denominator === 0 || isNaN(denominator)) {
    return fallback;
  }
  const result = numerator / denominator;
  if (!isFinite(result) || isNaN(result)) {
    return fallback;
  }
  return result;
}

// Safe percentage formatting
function formatPercent(value) {
  if (value === null || value === undefined || isNaN(value)) {
    return '-'; // Em dash, not "NaN%"
  }
  return `${value.toFixed(2)}%`;
}

// Safe currency formatting
function formatCurrency(value) {
  if (value === null || value === undefined || isNaN(value)) {
    return 'Not provided'; // Not "$N/A"
  }
  return `$$${value.toLocaleString('en-US', { minimumFractionDigits: 2 })}`;
}

// Before rendering, validate all values
function validateProposalData(data) {
  const issues = [];

  if (!data.current_monthly_volume || data.current_monthly_volume === 0) {
    issues.push('Monthly volume is missing or zero');
  }
  if (!data.current_monthly_fees || data.current_monthly_fees === 0) {
    issues.push('Current fees are missing or zero');
  }
  if (isNaN(data.dp_monthly_savings)) {
    issues.push('Dual pricing savings calculation failed');
  }

  if (issues.length > 0) {
    console.error('DATA VALIDATION FAILED:', issues);
    // Either throw error or handle gracefully
  }
}

```

```
    return issues;
}
```

BUG #6: IMPLEMENTATION PLAN TABLE TRUNCATED

Symptoms:

- Table rows cut off mid-sentence: "Conduct a detailed consultation with Robert to rev"

FIX REQUIRED:

Adjust table column widths or enable text wrapping:

```
// If using docx-js
new Table({
  width: { size: 100, type: WidthType.PERCENTAGE },
  columnWidths: [500, 2000, 4000, 1000], // Adjust proportions
  rows: [
    // ... rows with proper cell configuration
    new TableCell({
      // Enable text wrapping
      children: [
        new Paragraph({
          children: [new TextRun({ text: fullDescriptionText })]
        })
      ],
      // Don't set fixed width that causes overflow
    })
  ]
});
```

If using PDF renderer (reportlab, etc.), ensure multi-line cell support is enabled.

DEBUGGING CHECKLIST

Run through this checklist to identify exactly where the pipeline breaks:

- 1. FORM SUBMISSION
 - Log form data on submit: `console.log(formData)`
 - Verify all fields have values (not empty strings)

- 2. API RECEIPT
 - Log req.body on server
 - Verify files are attached and have content (size > 0)
 - 3. FILE PARSING
 - Log raw file content (first 500 chars)
 - Verify PDF parser is being used (not Excel parser)
 - Log each extracted value individually
 - 4. DATA VALIDATION
 - Check for zeros/nulls/NaN before proceeding
 - Log validation results
 - 5. TEMPLATE INJECTION
 - Log the complete data object being sent to Claude/template
 - Verify placeholders are being replaced
 - 6. PDF GENERATION
 - Check for rendering errors in console
 - Verify images/logos are embedded correctly
-

EXPECTED OUTPUT AFTER FIX

When the same inputs are submitted, the proposal should show:

Cover Page:

[PCBancard Logo]

Custom Payment Processing Proposal
Prepared exclusively for Cowtown Corner

[Bob's Muffler Logo - scraped from website]

Prepared For:
Cowtown Corner
Robert Smith (Owner)
123 Main St, Durham, NC
(555) 123-4567
robert@cowtowncorner.com
www.bobsmufflerservice.com

Prepared on: 1/30/2026

Prepared by: [Agent First] [Agent Last], Account Executive
Direct line: [Agent Phone] | Email: [Agent Email]

Current Processing Snapshot:

Monthly Processing Volume: \$34,738.26
Monthly Transactions: 309
Average Ticket Size: \$112.42
Current Monthly Fees: \$810.83
Effective Rate: 2.33%
Annual Processing Cost: \$9,729.96

Options Comparison:

Option	Monthly Cost	Monthly Savings	Annual Savings
Current Processor	\$810.83	—	—
Dual Pricing	\$64.95	\$745.88	\$8,950.51
Interchange Plus	\$653.65	\$157.18	\$1,886.12

Equipment Recommendation:

Based on your monthly volume of \$34,738.26, [Equipment Name] is recommended...

IMPLEMENTATION PRIORITY

1. **HIGHEST:** Fix spreadsheet/PDF parsing (this is completely broken)
2. **HIGH:** Fix form data flow (agent/merchant info not passing through)
3. **HIGH:** Fix side-by-side comparison rendering
4. **MEDIUM:** Add website logo scraping
5. **MEDIUM:** Fix NaN/\$N/A display issues
6. **LOW:** Fix table truncation

Start with #1 — if parsing works, most other issues may resolve themselves.

TEST CASE

Use this exact data to verify the fix:

Input:

- Business: Cowtown Corner
- Owner: Robert Smith
- Website: <https://www.bobsmufflerservice.com/>
- DP Spreadsheet: Cowtown_Corner__DP__-_Melody_Fitzgerald.pdf
- IC Spreadsheet: Cowtown_Corner_-_Melody_Fitzgerald.pdf

Expected Extraction:

- Volume: \$34,738.26
- Transactions: 309
- Current Fees: \$810.83
- DP Cost: \$64.95, Savings: \$745.88/mo, \$8,950.51/yr
- IC Cost: \$653.65, Savings: \$157.18/mo, \$1,886.12/yr

If any of these values show as \$0, 0, NaN, or N/A — the fix is incomplete.