Google Docs Wedding Contract Template - Implementation Guide

Overview

This comprehensive guide will walk you through setting up and using the wedding contract template with Google Sheets mail merge functionality. The template includes 14 major contract sections with professional merge fields that automatically populate from your Google Sheets data.

Table of Contents

Initial Setup

Prerequisites

- Google Account with access to Google Drive, Docs, and Sheets
- Wedding contract template files (provided)
- Basic understanding of Google Workspace tools

File Structure

Your wedding contract system should include:
- `contract_template.docx` - Word version of the contract
- `contract_template.md` - Markdown source file
- Google Sheets workbook with client data and calculations
- Google Docs version (created from Word file)

Google Sheets Data Structure

Required Sheets in Your Workbook

1. Client Information Sheet

Create a sheet named “Client_Data” with these columns:

Basic Client Information:
- `Contract_Number` - Unique identifier (e.g., WED-2025-001)
- `Contract_Date` - Date contract was created

- `Client_Primary_Name` - Primary client full name
- `Client_Secondary_Name` - Secondary client full name
- `Client_Address` - Street address
- `Client_City_State_ZIP` - City, state, and ZIP code
- `Client_Primary_Phone` - Primary contact phone
- `Client_Secondary_Phone` - Secondary contact phone
- `Client_Primary_Email` - Primary email address
- `Client_Secondary_Email` - Secondary email address
- `Emergency_Contact_Name` - Emergency contact person
- `Emergency_Contact_Phone` - Emergency contact number

Event Details:
- `Wedding_Date` - Wedding date (MM/DD/YYYY format)
- `Ceremony_Start_Time` - Ceremony start time
- `Reception_Start_Time` - Reception start time
- `Event_End_Time` - Event end time
- `Total_Event_Duration` - Total hours of service
- `Ceremony_Venue_Name` - Ceremony venue name
- `Ceremony_Venue_Address` - Ceremony venue address
- `Reception_Venue_Name` - Reception venue name
- `Reception_Venue_Address` - Reception venue address
- `Backup_Venue` - Backup venue (if applicable)
- `Guest_Count` - Expected number of guests
- `Wedding_Party_Size` - Size of wedding party
- `Special_Accommodations` - Any special needs

2. Services and Pricing Sheet

Create a sheet named “Services_Pricing” with these columns:

Service Details:
- `Primary_Services_List` - Main services included
- `Additional_Services_List` - Add-on services
- `Setup_Time` - Setup start time
- `Service_Start_Time` - Service start time
- `Service_End_Time` - Service end time
- `Breakdown_Time` - Breakdown/cleanup time
- `Deliverables_Timeline` - Timeline for deliverables
- `Equipment_Materials_List` - Equipment provided
- `Client_Responsibilities` - Client obligations

Pricing Information:
- `Base_Service_Fee` - Base service cost
- `Additional_Services_Fee` - Additional services cost
- `Travel_Charges` - Travel expenses
- `Equipment_Rental_Fee` - Equipment rental costs
- `Overtime_Rate` - Hourly overtime rate
- `Subtotal` - Subtotal before tax
- `Tax_Rate` - Tax percentage
- `Tax_Amount` - Calculated tax amount
- `Total_Contract_Amount` - Final total

3. Payment Schedule Sheet

Create a sheet named “Payment_Schedule” with these columns:

- `Retainer_Amount` - Initial deposit amount
- `Retainer_Due_Date` - Deposit due date
- `Second_Payment_Amount` - Second payment amount
- `Second_Payment_Due_Date` - Second payment due date
- `Final_Payment_Amount` - Final payment amount
- `Final_Payment_Due_Date` - Final payment due date
- `Accepted_Payment_Methods` - Payment methods accepted
- `Late_Fee_Percentage` - Late fee percentage
- `Payment_Grace_Period` - Grace period for late payments

4. Company Information Sheet

Create a sheet named “Company_Info” with these columns:

- `Company_Name` - Your business name
- `Company_Contact_Person` - Primary contact
- `Company_Address` - Business address
- `Company_City_State_ZIP` - Business city, state, ZIP
- `Company_Phone` - Business phone
- `Company_Email` - Business email
- `Business_License` - License number
- `Tax_ID` - Tax identification number

5. Legal and Policy Terms Sheet

Create a sheet named “Legal_Terms” with these columns:

Insurance and Liability:
- `General_Liability_Amount` - General liability coverage
- `Professional_Liability_Amount` - Professional liability coverage
- `Equipment_Insurance_Amount` - Equipment insurance coverage
- `Indemnification_Terms` - Indemnification clause

Cancellation Policies:
- `Cancellation_Period_1` - First cancellation period (days)
- `Refund_Percentage_1` - Refund percentage for period 1
- `Cancellation_Period_2` - Second cancellation period (days)
- `Refund_Percentage_2` - Refund percentage for period 2
- `Refund_Percentage_3` - Refund percentage for final period
- `Provider_Cancellation_Compensation` - Compensation if provider cancels
- `Rescheduling_Fee` - Fee for rescheduling
- `Rescheduling_Notice_Period` - Required notice for rescheduling

Intellectual Property:
- `IP_Ownership_Terms` - Intellectual property ownership
- `Usage_Rights_Description` - Client usage rights
- `Copyright_Terms` - Copyright information
- `Marketing_Usage_Restrictions` - Marketing usage limitations

6. Administrative Details Sheet

Create a sheet named “Admin_Details” with these columns:

- `Major_Change_Notice` - Days notice for major changes
- `Minor_Change_Notice` - Days notice for minor changes
- `Change_Administrative_Fee` - Fee for contract changes
- `Provider_Primary_Contact` - Primary provider contact
- `Client_Primary_Contact` - Primary client contact
- `Day_Of_Coordinator` - Day-of coordinator name
- `Primary_Communication_Method` - Preferred communication
- `Emergency_Contact_Method` - Emergency communication method
- `Response_Time_Business` - Response time commitment
- `Vendor_Coordination_Terms` - Third-party vendor terms
- `Governing_State` - Governing law state
- `Legal_Jurisdiction` - Legal jurisdiction
- `Special_Terms_Conditions` - Special terms
- `Weather_Policy` - Weather-related policies
- `Provider_Signatory_Name` - Person signing for provider
- `Provider_Signatory_Title` - Signatory title
- `Document_Version` - Template version
- `Last_Updated_Date` - Last update date
- `Template_ID` - Template identifier

Converting Template to Google Docs

Method 1: Upload Word Document

1. Open Google Drive
2. Click “New” → “File upload”
3. Select the `contract_template.docx` file
4. Once uploaded, right-click the file
5. Select “Open with” → “Google Docs”
6. Google will automatically convert it to Google Docs format
7. Rename the document to “Wedding Contract Template”

Method 2: Create from Markdown

1. Copy the content from `contract_template.md`
2. Open Google Docs
3. Create a new document
4. Paste the markdown content
5. Format the document:
   - Apply heading styles to section titles
   - Format merge fields consistently
   - Add page breaks where appropriate
   - Adjust spacing and typography

Mail Merge Setup Options

Option 1: Google Apps Script (Recommended)

Step 1: Create Apps Script Project

1. Open your Google Sheets workbook
2. Go to “Extensions” → “Apps Script”
3. Delete the default code and paste this mail merge script:

```
function createContractFromTemplate() {
  // Configuration
  const TEMPLATE_DOC_ID = 'YOUR_TEMPLATE_DOC_ID'; // Replace with your template ID
  const SHEET_NAME = 'Client_Data'; // Main data sheet

  // Get the active spreadsheet and data
  const ss = SpreadsheetApp.getActiveSpreadsheet();
  const sheet = ss.getSheetByName(SHEET_NAME);
  const data = sheet.getDataRange().getValues();
  const headers = data[0];

  // Process each row (skip header row)
  for (let i = 1; i < data.length; i++) {
    const rowData = data[i];

    // Create object with header-value pairs
    const mergeData = {};
    headers.forEach((header, index) => {
      mergeData[header] = rowData[index] || '';
    });

    // Create new document from template
    const templateDoc = DriveApp.getFileById(TEMPLATE_DOC_ID);
    const newDoc = templateDoc.makeCopy(`Contract - ${mergeData.Client_Primary_Name} -
${mergeData.Wedding_Date}`);
    const doc = DocumentApp.openById(newDoc.getId());
    const body = doc.getBody();

    // Replace merge fields
    Object.keys(mergeData).forEach(key => {
      const placeholder = `{{${key}}}`;
      body.replaceText(placeholder, mergeData[key]);
    });

    // Save and close
    doc.saveAndClose();

    // Log success
    console.log(`Contract created for ${mergeData.Client_Primary_Name}`);
  }
}

function onOpen() {
  const ui = SpreadsheetApp.getUi();
  ui.createMenu('Wedding Contracts')
    .addItem('Generate Contracts', 'createContractFromTemplate')
    .addToUi();
}
```

Step 2: Configure the Script

1. Replace `YOUR_TEMPLATE_DOC_ID` with your Google Docs template ID
2. Save the script (Ctrl+S or Cmd+S)
3. Run the `onOpen` function once to create the menu

Step 3: Set Permissions

1. Click “Run” to execute the script for the first time
2. Grant necessary permissions when prompted
3. The script will create a “Wedding Contracts” menu in your spreadsheet

Option 2: Third-Party Add-ons

Autocrat Add-on

1. Install Autocrat from Google Workspace Marketplace
2. Open your Google Sheets
3. Go to “Add-ons” → “Autocrat” → “Launch”
4. Follow the setup wizard to connect your sheet to the template
5. Map merge fields to spreadsheet columns
6. Configure trigger conditions (manual or automatic)

Yet Another Mail Merge

1. Install from Google Workspace Marketplace
2. Configure data source (your Google Sheets)
3. Set up document template (your Google Docs)
4. Map merge fields
5. Run merge process

Field Mapping Guide

Critical Field Mappings

Ensure these essential fields are properly mapped:

Client Information:
- `{{Client_Primary_Name}}` → Client_Data!Client_Primary_Name
- `{{Client_Secondary_Name}}` → Client_Data!Client_Secondary_Name
- `{{Wedding_Date}}` → Client_Data!Wedding_Date
- `{{Total_Contract_Amount}}` → Services_Pricing!Total_Contract_Amount

Financial Fields:
- `{{Base_Service_Fee}}` → Services_Pricing!Base_Service_Fee
- `{{Tax_Amount}}` → Services_Pricing!Tax_Amount
- `{{Retainer_Amount}}` → Payment_Schedule!Retainer_Amount

Company Information:
- `{{Company_Name}}` → Company_Info!Company_Name
- `{{Company_Phone}}` → Company_Info!Company_Phone
- `{{Company_Email}}` → Company_Info!Company_Email

Data Validation Tips

1. **Date Formatting:** Ensure dates are in consistent format (MM/DD/YYYY)
2. **Currency Fields:** Format as currency in Google Sheets
3. **Phone Numbers:** Use consistent format (XXX) XXX-XXXX
4. **Text Fields:** Check for proper capitalization and spelling
5. **Required Fields:** Mark essential fields and validate they're not empty

---

Testing and Troubleshooting

Pre-Merge Testing Checklist

- [ ] All merge fields have corresponding data columns
- [ ] Sample data is entered in all required fields
- [ ] Date and currency formatting is correct
- [ ] Template document is accessible and properly formatted
- [ ] Apps Script permissions are granted

Common Issues and Solutions

Issue: Merge fields not replacing

Solution:
- Check exact spelling and capitalization of field names
- Ensure merge field format matches ({{Field_Name}})
- Verify data exists in corresponding spreadsheet cells

Issue: Formatting lost during merge

Solution:
- Apply formatting to merge fields in template, not just surrounding text
- Use Google Docs native formatting rather than imported formatting
- Test with simple formatting first

Issue: Script execution errors

Solution:
- Check Google Apps Script execution transcript for specific errors
- Verify document IDs are correct
- Ensure proper permissions are granted
- Test with smaller data sets first

Issue: Generated documents not found

Solution:
- Check Google Drive for generated files
- Verify script has permission to create files
- Check if files are being created in a specific folder

Testing Process

1. **Create Test Data:** Use sample client information
2. **Run Small Batch:** Test with 1-2 records first
3. **Review Output:** Check generated contracts thoroughly

4. **Verify Calculations:** Ensure pricing and dates are correct
5. **Test Edge Cases:** Try with missing data, special characters
6. **Full Production Test:** Run with complete dataset

Advanced Customization

Adding Conditional Logic

Modify the Apps Script to include conditional content:

```
// Example: Add different terms based on service type
if (mergeData.Service_Type === 'Photography') {
  body.replaceText('{{Service_Specific_Terms}}', 'Photography-specific terms here...');
} else if (mergeData.Service_Type === 'Catering') {
  body.replaceText('{{Service_Specific_Terms}}', 'Catering-specific terms here...');
}
```

Automated Email Delivery

Add email functionality to your script:

```
function emailContract(docId, clientEmail, clientName) {
  const doc = DriveApp.getFileById(docId);
  const pdf = doc.getAs('application/pdf');

  GmailApp.sendEmail(
    clientEmail,
    `Wedding Contract - ${clientName}`,
    'Please find your wedding contract attached. Please review and return signed
copy.',
    {
      attachments: [pdf],
      name: 'Your Wedding Service Company'
    }
  );
}
```

Multi-Sheet Data Integration

Pull data from multiple sheets:

```
function getAllSheetData() {
  const ss = SpreadsheetApp.getActiveSpreadsheet();
  const sheets = ['Client_Data', 'Services_Pricing', 'Payment_Schedule',
'Company_Info', 'Legal_Terms', 'Admin_Details'];

  let allData = {};

  sheets.forEach(sheetName => {
    const sheet = ss.getSheetByName(sheetName);
    const data = sheet.getDataRange().getValues();
    const headers = data[0];
    const values = data[1]; // Assuming single row of data per sheet

    headers.forEach((header, index) => {
      allData[header] = values[index] || '';
    });
  });

  return allData;
}
```

Document Formatting Enhancements

Add professional formatting through script:

```
function formatDocument(doc) {
  const body = doc.getBody();

  // Set document margins
  body.setMarginTop(72);    // 1 inch
  body.setMarginBottom(72);
  body.setMarginLeft(72);
  body.setMarginRight(72);

  // Format headings
  const headings = body.findText('## .*');
  while (headings) {
    const element = headings.getElement();
    element.setFontSize(14);
    element.setBold(true);
    headings = body.findText('## .*', headings);
  }
}
```

Security and Compliance Considerations

Data Protection

- Store sensitive client data securely
- Use Google Sheets' built-in sharing controls
- Regularly audit access permissions
- Consider data retention policies

Legal Compliance

- Ensure contract terms comply with local laws

- Include required legal disclaimers
- Maintain version control of contract templates
- Keep audit trail of contract generations

Backup and Recovery

- Regularly backup your Google Sheets data
- Maintain template version history
- Document your merge field mappings
- Test recovery procedures

Support and Maintenance

Regular Maintenance Tasks

- Update contract templates as laws change
- Review and update merge field mappings
- Test mail merge functionality monthly
- Update company information as needed
- Archive completed contracts appropriately

Getting Help

- Google Apps Script documentation: https://developers.google.com/apps-script
- Google Workspace support: https://support.google.com/a
- Community forums for troubleshooting
- Consider hiring a developer for complex customizations

Conclusion

This implementation guide provides a comprehensive framework for automating your wedding contract generation process. Start with the basic setup and gradually implement advanced features as your needs grow. Regular testing and maintenance will ensure your system continues to work effectively as your business evolves.

For additional support or custom development needs, consider consulting with a Google Workspace specialist or Apps Script developer.