

## Appsheet (Paramater) – Post data to App Script

+

SAVE

S

## Settings

Send an email

Send a notification

Send an SMS

HTTP  
Call a webhook

Create a new file

Call a script

Table: **Invoice**

Table that this task works with. This is set in the process or determined by the event.

### Apps Script Project (APPSHEET CORE)

pdf\_create

Sample scripts Open

Script will be executed as somtola.smt@gmail.com.

### Function Name

Create\_Bulk\_Pdf (date, dd, customer\_name, |

### Function Parameters

date	= [Date]	
dd	= [Due Date]	
customer_name	= Select(Customers[	
phonenumber_1	= [Phone Number 1]	

+

↶

↷

⚠

SAVE

▼

S

⋮

📱

🖨

⚙

🔗

date	= [Date]	⚠
dd	= [Due Date]	⚠
customer_name	= Select(Customers[	⚠
phonenumber_1	= [Phone Number 1]	⚠
phonenumber_2	= [Phone Number 2]	⚠
address	= [Address]	⚠
delivery_man	= [DeliveryMan]	⚠
product_name	= [Product Name]	⚠
unit_price	= [Unit Price]	⚠
qty	= [Quantity]	⚠
total_amount	= [Total Amount]	⚠
delivery_fee	= [Delivery Fee]	⚠
grand_total	= [Grand Total]	⚠
amount_received	= [Amount Receive]	⚠
balance_due	= [Balance Due]	⚠
balance_due_khr	= [Balance Due (KHR	⚠
qr	= [QR]	⚠

## App Script use for execute on generate pdf and printnode

- Script
- Main function

```
function Create_Bulk_Pdf(date, dd, customer_name, phonenumber_1,
phonenumber_2, address, delivery_man, product_name, unit_price, qty,
total_amount, delivery_fee, grand_total, amount_received, balance_due,
balance_due_khr, qr, order, cashier, note, email, invID) {
  const fileId = "1yBUWsB0bnhqVhwhS_lsVyKEqD1RRYH4fX8piGaf8YUk";
  const folderId = "1vIVcfw0J5IG3YswgW9da14a04Yh11Sil";
  const folder = DriveApp.getFolderById(folderId);
  const tempFile = DriveApp.getFileById(fileId).makeCopy(folder);
  const docFile = DocumentApp.openById(tempFile.getId());

  const replacements = {
    "{date}": date,
    "{dd}": dd,
    "{customer_name}": customer_name,
    "{phonenumber_1}": phonenumber_1,
    "{phonenumber_2}": phonenumber_2,
    "{address}": address,
    "{delivery_man}": delivery_man,
    "{delivery_fee}": parseFloat(delivery_fee).toFixed(2),
    "{grand_total}": parseFloat(grand_total).toFixed(2),
    "{amount_received}": parseFloat(amount_received).toFixed(2),
    "{balance_due}": parseFloat(balance_due).toFixed(2),
    "{balance_due_khr}": parseFloat(balance_due_khr).toLocaleString('en-US'),
    "{order}": order,
    "{cashier}": cashier,
    "{note}": note,
  };

  // Perform all replacements in one go
  const body = docFile.getBody();
  Object.keys(replacements).forEach(key => {
    body.replaceText(key, replacements[key]);
  });

  // Batch replace placeholders for product details
  replacePlaceholders(docFile, product_name, "{product_name_", 10, false);
  replacePlaceholders(docFile, unit_price, "{unit_price_", 10, true);
  replacePlaceholders(docFile, qty, "{qty_", 10, false);
  replacePlaceholders(docFile, total_amount, "{total_amount_", 10, true);
}
```

```

// Insert QR code image if present
if (qr) {
  try {
    const imageBlob = UrlFetchApp.fetch(qr).getBlob();
    const searchResult = body.findText('{image}');
    if (searchResult) {
      const rangeElement = searchResult.getElement();
      const paragraph = rangeElement.getParent();
      paragraph.clear(); // Clear the placeholder
      paragraph.insertInlineImage(0, imageBlob);
    }
  } catch (error) {
    console.log("Failed to fetch QR image: " + error.message);
  }
}

docFile.saveAndClose();

// Create PDF and remove temporary file in one operation
const pdfFile =
folder.createFile(tempFile.getAs(MimeType.PDF)).setName(invID);
folder.removeFile(tempFile);

// Perform email and PrintNode operations concurrently
Promise.all([
  sendEmailAsync(email, customer_name, "Invoice was generated.", pdfFile),
  sendPrintNodeAsync(pdfFile, "73757406",
"WbT2Sm6TrSZubcDXYTtcLvAU8Y0S67Y1cd0taSRzo2A")
]).then(() => {
  console.log("Email sent and print job submitted.");
});
}

```

- Helper function to replace placeholders in the document

```
function replacePlaceholders(doc, values, placeholderPrefix,
maxPlaceholders, isCurrency) {
  const body = doc.getBody();
  for (let i = 0; i < values.length; i++) {
    let placeholder = placeholderPrefix + (i + 1) + "}";
    let value = values[i];
    let formattedValue = isCurrency ? "$" +
parseFloat(value).toFixed(2) : value;
    body.replaceText(placeholder, formattedValue);
  }
  for (let i = values.length + 1; i <= maxPlaceholders; i++) {
    let unusedPlaceholder = placeholderPrefix + i + "}";
    body.replaceText(unusedPlaceholder, "");
  }
}
```

- Async email sending function

```
function sendEmailAsync(email, subject, body, pdfFile) {
  return new Promise((resolve) => {
    MailApp.sendEmail(email, subject, body, {
      attachments: [pdfFile.getAs(MimeType.PDF)],
    });
    resolve();
  });
}
```

- Async function for sending print job

```
function sendPrintNodeAsync(pdfFile, printerId, apiKey) {
  return new Promise((resolve) => {
    sendToPrintNode(pdfFile, printerId, apiKey);
    resolve();
  });
}
```

- Function to send the pdf to PrintNode for printing

```
function sendToPrintNode(pdfFile, printerId, apiKey) {  
  const url = "https://api.printnode.com/printjobs";  
  const pdfContent = pdfFile.getBlob().getBytes();  
  const encodedPdf = Utilities.base64Encode(pdfContent);  
  
  const payload = {  
    printerId: printerId,  
    title: "Invoice",  
    contentType: "pdf_base64",  
    content: encodedPdf,  
    source: "Google Apps Script"  
  };  
  
  const options = {  
    method: "post",  
    contentType: "application/json",  
    payload: JSON.stringify(payload),  
    headers: {  
      Authorization: "Basic " + Utilities.base64Encode(apiKey + ":")  
    }  
  };  
  
  UrlFetchApp.fetch(url, options);  
}
```



Apps Script pdf\_create

```

83
84 // Async email sending function
85 function sendEmailAsync(email, subject, body, pdfFile) {
86   return new Promise((resolve) => {
87     MailApp.sendEmail(email, subject, body, {
88       attachments: [pdfFile.getAs(MimeType.PDF)],
89     });
90     resolve();
91   });
92 }
93
94 // Async function for sending print job
95 function sendPrintNodeAsync(pdfFile, printerId, apiKey) {
96   return new Promise((resolve) => {
97     sendToPrintNode(pdfFile, printerId, apiKey);
98     resolve();
99   });
100 }
101
102 // Function to send the PDF to PrintNode for printing
103 function sendToPrintNode(pdfFile, printerId, apiKey) {
104   const url = "https://api.printnode.com/printjobs";
105   const pdfContent = pdfFile.getBlob().getBytes();
106   const encodedPdf = Utilities.base64Encode(pdfContent);
107
108   const payload = {
109     printerId: printerId,
110     title: "Invoice",
111     contentType: "pdf_base64",
112     content: encodedPdf,
113     source: "Google Apps Script"
114   };
115
116   const options = {
117     method: "post",
118     contentType: "application/json",
119     payload: JSON.stringify(payload),
120     headers: {
121       Authorization: "Basic " + Utilities.base64Encode(apiKey + ":")
122     }
123   };
124
125   UrlFetchApp.fetch(url, options);
126 }
127

```

- Template Google Docs

Link:

[https://docs.google.com/document/d/1yBUWsB0bnhqVhwhS\\_IsVyEqD1R\\_RYH4fX8piGaf8YUk/edit?usp=sharing](https://docs.google.com/document/d/1yBUWsB0bnhqVhwhS_IsVyEqD1R_RYH4fX8piGaf8YUk/edit?usp=sharing)

Template Invoices

File Edit View Insert Format Tools Extensions Help

100% Normal text Arial

INVOICE

អ្នក (aaa) អ្នកប្រគល់ (ddd)

ឈ្មោះអតិថិជន (customer\_name)

លេខទូរស័ព្ទ 0{phonenumber\_1} 0{phonenumber\_2}

អាសយដ្ឋាន {address}

ប្រគល់ដោយ (delivery\_man)

ផលិតផល	តម្លៃ	ចំនួន	សរុប
{product_name_1}	{unit_price_1}	{qty_1}	{total_amount_1}
{product_name_2}	{unit_price_2}	{qty_2}	{total_amount_2}
{product_name_3}	{unit_price_3}	{qty_3}	{total_amount_3}
{product_name_4}	{unit_price_4}	{qty_4}	{total_amount_4}
{product_name_5}	{unit_price_5}	{qty_5}	{total_amount_5}
{product_name_6}	{unit_price_6}	{qty_6}	{total_amount_6}
{product_name_7}	{unit_price_7}	{qty_7}	{total_amount_7}
{product_name_8}	{unit_price_8}	{qty_8}	{total_amount_8}
{product_name_9}	{unit_price_9}	{qty_9}	{total_amount_9}
{product_name_10}	{unit_price_10}	{qty_10}	{total_amount_10}

ប្រាក់បញ្ញើផ្គត់ផ្គង់ Delivery(\$)

សរុប (\$)

ប្រាក់ទទួលបាន Received(\$)

ប្រាក់នៅត្រូវបង់ \$ {balance\_due}

ប្រាក់នៅត្រូវបង់ Balance(\$)

ប្រាក់នៅត្រូវបង់ {balance\_due\_khr}

{(image)}

{order} - {cashier}

{note}