Contents

Project 1: Building of an Automated System for Processing of Requests

Requirements Analysis

1. Inputs and Outputs?
2. Apps to Integrate?
3. Conversion, Transformation, Parsing Process?
4. Decision paths?

Planning the flow and integration

1. Trigger: When a new response is submitted in MS Forms.
2. Setup: Initialize a Variable for 'RequestStatus' and set it to 'Pending'.
3. Data Retrieval: Get the response details. Then, use the 'List rows present in a table' action from Excel Online to retrieve a list of Department Approvers.
4. Data Processing: Use a 'Filter Array' action to find the specific approver from the Excel data that matches the Department selected in the Form. Use a 'Compose' action to extract the approver's email from the array.
5. Parallel Process: Add a Parallel Branch.
    - Branch A: Create a new item in a SharePoint List (or MS Lists) to log the request details.
    - Branch B: Post a message to a specific MS Teams channel notifying the team of a new request.
6. Logic Control: Add a 'Switch' control based on the 'Request Priority' field from the Form.
    - Case 'High': Add a 'Delay' of 1 minute (simulation) before proceeding.
    - Case 'Normal': Proceed immediately.
7. Approval: Start and wait for an Approval using the email found in the Compose step.
8. Decision: Add a 'Condition' to check if the outcome is 'Approve'.
    - If Yes: Send an email via Outlook. The body must use HTML to create a formatted table summarizing the request. Update the SharePoint item to 'Approved'.
    - If No: Update the SharePoint item to 'Rejected' and terminate.

Identifying limitations and possible workarounds

Class Discussion

Project prototype building

Project Discussion	15 mins
Prototype Development	2 hrs
Checking and extension for development duration	45 mins
Troubleshooting and Adjustments	1 hr
Total	4 hrs

Testing, Checking and Evaluation

#	Skill Measured	Applied In	Specific Application in this Flow	Points	Evaluation
1	Automated Trigger Configuration	Step 1	Configuring the MS Forms "When a new response is submitted" trigger to initiate the workflow upon user input.	10	
2	Variable Initialization & Management	Step 2	Defining a String Variable (RequestStatus) and setting a default value ('Pending') for use later in the lifecycle.	10	
3	External Data Retrieval (Connectors)	Step 3	Using the Excel Online (Business) connector to fetch a reference table (List rows...) containing Department Approver data.	10	
4	Data Operations (Filtering Arrays)	Step 4	Implementing the Filter Array action to query the Excel dataset and isolate the single row matching the user's "Department" selection.	10	
5	Expression Language (Data Extraction)	Step 4	Using a Compose action with expressions (e.g., body('Filter_Array')?[0]?['Email']) to extract a single email string from the data array.	10	
6	Concurrency Control (Parallelism)	Step 5	setting up a Parallel Branch to execute two independent actions (SharePoint logging and Teams notification) simultaneously for efficiency.	10	
7	Multi-Platform Integration	Step 5	interacting with two distinct ecosystems (SharePoint and MS Teams) within a single logic block to log data and notify users.	10	
8	Advanced Control Logic (Switch Cases)	Step 6	configuring a Switch control to handle multiple distinct paths based on the 'Priority' value ('High' vs. 'Normal').	10	
9	Approval Workflow Management	Step 7	implementing the Start and wait for an approval action, dynamically assigning the approver based on the email derived in Step 4.	10	
10	HTML Formatting & Conditional Output	Step 8	Constructing a dynamic HTML table within an Outlook email body and using Conditions to determine final status updates (Approve/Reject).	10	

Finalization and Monitoring

Class Discussion

Additional Note:

Here is the HTML code snippet for a clean, table-based layout with inline styles to ensure it renders correctly in Outlook.

Instructions

1. Add a Compose action in your flow (inside the "If Yes" branch, before the "Send an email" action).
2. Paste the code below into the Inputs field.

3. Crucial Step: Delete the text inside the square brackets (e.g., [INSERT DYNAMIC CONTENT HERE]) and select the actual Dynamic Content from your MS Forms trigger (e.g., Responder Email, Submission Time, Description).

The HTML Code

```
<div style="font-family: 'Segoe UI', Arial, sans-serif; max-width: 600px; color: #333;">

 <div style="background-color: #003366; color: #ffffff; padding: 15px; text-align: center;">
  <h2 style="margin: 0;">Operational Request Approved</h2>
  <p style="margin: 5px 0 0 0; font-size: 14px;">Bangko Sentral ng Pilipinas - Internal Automated System</p>
 </div>

 <div style="padding: 20px; border: 1px solid #ddd; border-top: none;">
  <p>Dear Team,</p>
  <p>The following request has been successfully processed and <strong>approved</strong>. Please find the summary of the request details below:</p>

  <table style="width: 100%; border-collapse: collapse; margin-top: 15px;">

   <tr style="background-color: #f9f9f9;">
    <td style="padding: 10px; border: 1px solid #ddd; width: 35%; font-weight: bold;">Requestor Email:</td>
    <td style="padding: 10px; border: 1px solid #ddd;">[INSERT RESPONDER EMAIL HERE]</td>
   </tr>

   <tr style="background-color: #ffffff;">
    <td style="padding: 10px; border: 1px solid #ddd; font-weight: bold;">Submission Date:</td>
    <td style="padding: 10px; border: 1px solid #ddd;">[INSERT SUBMISSION TIME HERE]</td>
   </tr>

   <tr style="background-color: #f9f9f9;">
    <td style="padding: 10px; border: 1px solid #ddd; font-weight: bold;">Department:</td>
    <td style="padding: 10px; border: 1px solid #ddd;">[INSERT DEPARTMENT ANSWER HERE]</td>
   </tr>

   <tr style="background-color: #ffffff;">
    <td style="padding: 10px; border: 1px solid #ddd; font-weight: bold;">Priority Level:</td>
    <td style="padding: 10px; border: 1px solid #ddd; color: #d9534f; font-weight: bold;">[INSERT PRIORITY ANSWER HERE]</td>
   </tr>

   <tr style="background-color: #f9f9f9;">
    <td style="padding: 10px; border: 1px solid #ddd; font-weight: bold; vertical-align: top;">Description:</td>
    <td style="padding: 10px; border: 1px solid #ddd;">[INSERT DESCRIPTION ANSWER HERE]</td>
   </tr>

  </table>

  <p style="margin-top: 20px; font-size: 12px; color: #666;">
```

```
    <em>This is an automated message generated by the BSP Power Automate System. Please do not reply directly to this email.</em>
   </p>
  </div>
</div>
```

Project 2: Building of Consolidation of Schedule of COD Mandatory Leaves and Dates of Surprise Assumption Automation

Requirements Analysis

1. Inputs and Outputs?
2. Apps to Integrate?
3. Conversion, Transformation, Parsing Process?
4. Decision paths?

Planning the flow and integration

1. Trigger the flow weekly. Initialize a String variable named "ComplianceStatus" and an Integer variable named "ReviewCycle".
2. In a **Parallel Branch**:
    - Branch A: **Get items** from the SharePoint List named "Mandatory Leave Log" where the status is "Pending".
    - Branch B: **List rows** from the Excel Online table named "SurpriseAssumptionDates".
3. Use a **Compose** action to combine the SharePoint items and Excel rows. Then, use the **Filter Array** action to select only items where the "LeaveDate" is in the next 7 days.
4. Use the **Select** data operation to map the filtered data into specific columns (Employee Name, Date, Department), and then pass this into a **Create HTML Table** action.
5. **Start and wait for an approval** (First to respond) sending the HTML Table to the "Department Head" for review.
6. Add a **Condition** to check if the Approval Outcome is "Approve":
    - **If Yes:**
        - Add a **Switch** action based on the "Department" value.
        - **Case 1 (Treasury):** Post a message to the "Treasury Compliance" **Teams** channel.
        - **Case 2 (Operations):** Post a message to the "Ops Risk" **Teams** channel.
        - **Default:** Send an **Outlook** email to the generic "Audit Team" address. The email body must contain the **HTML Table** modified with a replace expression to add inline CSS borders.
    - **If No:**
        - **Delay** the flow for 1 hour.
        - Send an email to the initiator requesting a revision.
7. At the very end, use a **Compose** action to log the final run timestamp.

Identifying limitations and possible workarounds

Class Discussion

Project prototype building

Project Discussion	15 mins
Prototype Development	2 hrs
Checking and extension for development duration	45 mins
Troubleshooting and Adjustments	1 hr
Total	4 hrs

Testing, Checking and Evaluation

Item No.	Skill	Where it appears in the Flow	Points	Participant's Score
1	**Integration (SP, Excel, Teams, Outlook)**	Trigger (Schedule), Get Items (SP), List Rows (Excel), Post Message (Teams), Send Email (Outlook).	10	
2	**MS Lists & MS Forms**	Covered by the "Get Items" (Lists) and the implicit data source. *Note: Forms is usually a trigger, but for "Consolidation" flows, Scheduled triggers are better. If you strictly need Forms, see the "Alternative" below.*	10	
3	**Approvals**	"Start and wait for an approval" step included.	10	
4	**Parallel Branching**	Specifically requested in Step 2 to fetch SP and Excel data simultaneously (performance optimization).	10	
5	**Variables**	"Initialize Variable" step included at the start.	10	
6	**Array Controls**	"Filter Array" and "Select" data operations are explicitly requested to manipulate the data sets.	10	
7	**HTML Modification**	The prompt asks for "Create HTML Table" and specifically requests a replace expression to inject CSS (a common advanced interview skill).	10	
8	**Conditions & Switches**	Step 6 includes a Condition (Approved/Rejected) and a Switch (Department routing).	10	
9	**Delays**	A 1-hour delay is included in the "If No" branch.	10	
10	**Compose**	Used to merge data and log timestamps.	10	

Finalization and Monitoring

Class Discussion

Additional Note:

In Power Automate, the Create HTML table action produces a very plain table with no borders. To make it professional (like a Banking report), you need to replace the standard HTML tags with "styled" tags using an expression.

The Code Snippet (Expression)
You will use a **Compose** action immediately after your "Create HTML Table" step. Paste this expression into the **Inputs** field of the Compose action:

```
replace(
  replace(
    replace(
      body('Create_HTML_table'),
      '<table>',
      '<table style="border-collapse: collapse; width: 100%; font-family: Arial, sans-serif;">'
    ),
    '<th>',
    '<th style="background-color: #003366; color: white; padding: 10px; border: 1px solid #ddd; text-align: left;">'
  ),
  '<td>',
  '<td style="padding: 8px; border: 1px solid #ddd;">'
)
```

Note: You must change 'Create_HTML_table' in the code above to match the exact name of your action if it has a different name (e.g., Create_HTML_table_2).

How this works:

1. **First Replace:** Finds the <table> tag and adds border-collapse (so you don't get double lines) and sets the font to Arial.
2. **Second Replace:** Finds the header <th> tags and adds a Dark Blue background (#003366 - common for banking), white text, and padding.
3. **Third Replace:** Finds the cell <td> tags and adds light grey borders and padding so the text isn't squashed.

How to implement this in your Flow:

1. Locate the **Create HTML Table** action.
2. Add a **Compose** action immediately after it.
3. Click inside the **Inputs** field of the Compose action.
4. Select the **Expression** tab (usually the fx icon).
5. Paste the code snippet above into the expression bar and click **OK**.
6. **Crucial Last Step:** In your "Send an email" action, do **not** use the *Output* from the "Create HTML Table" action. Instead, use the *Output* from this new **Compose** action.

Project 3: Preparation of Disbursement Vouchers

Requirements Analysis

1. Inputs and Outputs?
2. Apps to Integrate?
3. Conversion, Transformation, Parsing Process?
4. Decision paths?

Planning the flow and integration

Phase 1: Preparation (SharePoint & MS Forms)
1. Create SharePoint List: TravelRequests
   - ID: (Default column, use this for EDTSno).
   - Title: Rename default "Title" column to TAOno.
   - TRACKERno: Single line of text.
   - Fullname: Single line of text.
   - Department: Single line of text.
   - StartDate: Date and Time (Date only).
   - EndDate: Date and Time (Date only).
   - SubmissionDate: Single line of text.
   - UsingTracker: Choice (Values: Yes, No).
   - SourceOfFunds: Choice (Values: PersonalExpense, CompanyAllowance).
   - TrainingAuthority: Single line of text (to store the filename).
   - Status: Single line of text (Default value: Pending).
2. Create SharePoint List: DisbursementVouchers
   - Title: Rename to TAOid.
   - Fullname: Single line of text.
   - TotalAmountDue: Currency or Number.
   - *Note:* The files (doc1, doc2, doc3) will be attached to the list item or sent via email.
3. Create MS Form 1: Request Form
   - Questions: Fullname (Text), Department (Text), SourceOfFunds (Choice), UsingTracker? (Choice: Yes/No), StartDate (Date), EndDate (Date), Training Authority (File Upload).
4. Create MS Form 2: DV Prep Form
   - Questions: TAOid (Text), Fullname (Text), EventStart (Date), EventEnd (Date), Doc1 (Upload), Doc2 (Upload), Doc3 (Upload).

Phase 2: Flow 1 (Submission & Approval)
Note: We cannot generate EDTSno (the SharePoint ID) until the item is created. We will create the item first, then update it with the logic.

1. Trigger: MS Forms – When a new response is submitted.
2. Action: MS Forms – Get response details.
3. Action: Data Operation – Initialize Variable.
   - Name: varFileName.
   - Type: String.
   - Value: Use this expression to get the filename from the Form upload: first(json(outputs('Get_response_details')?['body/your_file_question_id']))?['name'] *(Replace your_file_question_id with the dynamic content from the form).*
4. Action: Data Operation – Compose (Generate Tracker No).

- Inputs: formatDateTime(utcNow(), 'yyyyMMddHHmm')

5. Action: SharePoint – Create item.
   - *List:* TravelRequests.
   - *Fullname, Department, etc.:* Map from Form dynamic content.
   - *TrainingAuthority:* Select the varFileName variable.
   - *Status:* Enter "Pending".
6. Action: SharePoint – Update item.
   - *ID:* Select ID from the "Create item" step.
   - *TAOno (Title):* Use this expression logic: if(equals(outputs('Get_response_details')?['body/UsingTracker'], 'Yes'), outputs('Compose'), string(outputs('Create_item')?['body/ID']))
   - *TRACKERno:* Select output from the Compose step.
7. Action: Approvals – Start and wait for an approval.
   - Type: Approve/Reject - First to respond.
   - Title: Request for @{triggerOutputs()?['body/resourceData/responseId']}.
8. Action: Condition.
   - *Expression:* Outcome is equal to Approve.
   - If Yes:
     - Action: SharePoint – Update item.
       - *ID:* ID from Create item.
       - *Status:* Approved.
   - If No:
     - Action: SharePoint – Update item.
       - *Status:* Denied.

Phase 3: Flow 2 (Reimbursement/Liquidation)

Note: To convert HTML to PDF without premium connectors, we use the "OneDrive for Business" workaround.

1. Trigger: SharePoint – When an item is created or modified.
   - *List:* TravelRequests.
2. Action: Control – Condition.
   - *Logic:* Status is equal to Approved.
   - *(Crucial: Go to Settings of the Trigger and add a "Trigger Condition" to prevent infinite loops: @equals(triggerBody()?['Status'], 'Approved'))*.
3. If Yes (inside Condition):
   - Action: Data Operation – Compose (Calculate Days).
     - Expression: div(sub(ticks(formatDateTime(triggerOutputs()?['body/EndDate'],'yyyy-MM-dd')),ticks(formatDateTime(triggerOutputs()?['body/StartDate'],'yyyy-MM-dd'))),86400000)
   - Action: Data Operation – Compose (Calculate Amount).
     - Expression: mul(outputs('Compose_-_Calculate_Days'), 1000)
   - Action: Data Operation – Compose (HTML Table).
     - Input: Write standard HTML code here.

```
<html>
<body>
 <h2>Reimbursement Details</h2>
 <table>
  <tr><td>Name:</td><td>@{triggerOutputs()?['body/fullname']}</td></tr>
  <tr><td>TAO No:</td><td>@{triggerOutputs()?['body/Title']}</td></tr>
  <tr><td>Total Amount:</td><td>@{outputs('Compose_-_Calculate_Amount')}</td></tr>
 </table>
</body>
```

```
</html>
```

- Action: OneDrive for Business – Create file.
  - *Folder Path:* /Tmp
  - *File Name:* Reimbursement_@{triggerOutputs()?['body/ID']}.html
  - *File Content:* Output of "Compose (HTML Table)".
- Action: OneDrive for Business – Convert file (preview).
  - *File:* Id from "Create file".
  - *Target format:* pdf.
- Action: Outlook – Send an email (V2).
  - *To:* (User's email).
  - *Subject:* Approved: Reimbursement for @{triggerOutputs()?['body/Title']}.
  - *Body:* Output of "Compose (HTML Table)".
  - *Attachments Name:* Reimbursement.pdf.
  - *Attachments Content:* File content from "Convert file".

4. If No (inside Condition):
   - (Add a nested condition to check if Status is Denied, then Send Email Notification).

Phase 4: Flow 3 (PrepDV)

1. Trigger: MS Forms – When a new response is submitted (DV Form).
2. Action: MS Forms – Get response details.
3. Action: Data Operation – Compose (Calculate DV Amount).
   - Expression:
     mul(div(sub(ticks(formatDateTime(outputs('Get_response_details')?['body/EventEnd'],'yyyy-MM-dd')),ticks(formatDateTime(outputs('Get_response_details')?['body/EventStart'],'yyyy-MM-dd'))),86400000), 1000)
4. Action: SharePoint – Create item (List: DisbursementVouchers).
   - Map fields from Form.
   - *TotalAmountDue:* Output of Compose (Calculate DV Amount).
5. Action: OneDrive for Business – Create file (HTML).
   - Content: Create HTML string similar to Flow 2, including the calculated amount.
6. Action: OneDrive for Business – Convert file (HTML to PDF).
7. Action: Approvals – Start and wait for an approval.
   - *Title:* Approve DV for @{outputs('Get_response_details')?['body/Fullname']}.
   - *Attachment Name:* Voucher.pdf.
   - *Attachment Content:* Body of "Convert file".
8. Action: Condition (Outcome = Approve).
   - If Yes: Send Email to User (Subject: DV Approved) + Attach PDF + HTML in Body.
   - If No: Send Email to User (Subject: DV Denied).

Identifying limitations and possible workarounds

Class Discussion

Project prototype building

Project Discussion	15 mins
Prototype Development	2 hrs
Checking and extension for development duration	45 mins
Troubleshooting and Adjustments	1 hr
Total	4 hrs

Testing, Checking and Evaluation

#	Skill	Where it appears	Application in your Solution	Points	Participant's Score
1	**JSON Parsing**	Flow 1 (File Upload)	Using json(...) to extract the filename from the MS Forms attachment array.	10	
2	**WDL Expressions**	Flow 2 (Calc. Allowance)	Using div(sub(ticks(End), ticks(Start)), ...) to calculate date differences.	10	
3	**Trigger Conditions**	Flow 2 (Trigger Settings)	Setting @equals(triggerBody()?['Status'], 'Approved') in settings.	10	
4	**Sequential Logic**	Flow 1 (Create vs Update)	Creating the item *first* to get the ID, then updating it to set the TAOno.	10	
5	**Doc Generation**	Flow 2 & 3 (PDFs)	Creating an HTML file in OneDrive, then using "Convert File" to make it a PDF.	10	
6	**Inline Conditionals**	Flow 1 (Tracker Logic)	Using if(...) inside the "Update Item" field to switch between TrackerNo and ID.	10	
7	**Approvals V2**	Flow 1 & 3	Using "Start and wait for an approval" to pause automation for human input.	10	
8	**Date Formatting**	Flow 1 (Tracker No)	Using formatDateTime(utcNow(), 'yyyyMMddHHmm') to generate IDs.	10	
9	**Data Operations**	Throughout all Flows	Using "Compose" for HTML templates and calculations vs. "Variables".	10	
10	**Dynamic Attachments**	Flow 2 & 3 (Email)	Passing the *file content* from OneDrive directly into the Outlook attachment field.	10	

Finalization and Monitoring

Class Discussion

Additional Notes

- **Dates:** SharePoint and Forms sometimes handle time zones differently. Use formatDateTime(..., 'yyyy-MM-dd') explicitly to ensure calculations don't break due to timestamps.
- **File Parsing:** The json(...) expression in Flow 1 Step 3 is required because Forms sends file details as a JSON array string, not a direct filename.
- **PDF Conversion:** The OneDrive "Convert File" action is the standard non-premium way to create PDFs. You must create the HTML file in OneDrive *first*, then convert that specific file.

Here are the HTML templates for your solution. You can copy these directly into the **"Compose"** actions in your flows.

1. Reimbursement Table (Flow 2)

*Use this in **Flow 2**, Step 4 ("Create HTML table").*

```
<!DOCTYPE html>
<html>
<head>
<style>
 body { font-family: 'Segoe UI', Arial, sans-serif; color: #333; }
 .container { width: 100%; max-width: 600px; margin: 0 auto; border: 1px solid #ddd; }
 .header { background-color: #0078d4; color: white; padding: 20px; text-align: center; }
 .content { padding: 20px; }
 table { width: 100%; border-collapse: collapse; margin-top: 20px; }
 th, td { padding: 12px; border-bottom: 1px solid #ddd; text-align: left; }
 th { background-color: #f9f9f9; width: 40%; }
 .total-row { font-weight: bold; background-color: #e6f2fa; }
 .footer { font-size: 12px; color: #777; padding: 20px; text-align: center; background-color: #f4f4f4; }
</style>
</head>
<body>
 <div class="container">
  <div class="header">
   <h2>Travel Reimbursement Summary</h2>
  </div>
  <div class="content">
   <p>Dear <strong>@{triggerOutputs()?['body/fullname']}</strong>,</p>
   <p>Your travel request has been <strong>APPROVED</strong>. Below is the computation for your
reimbursement.</p>

   <table>
    <tr>
     <th>Tracking / TAO No.</th>
     <td>@{triggerOutputs()?['body/Title']}</td>
    </tr>
    <tr>
     <th>Department</th>
     <td>@{triggerOutputs()?['body/department']}</td>
    </tr>
    <tr>
     <th>Start Date</th>
```

```
      <td>@{formatDateTime(triggerOutputs()?['body/StartDate'], 'MM/dd/yyyy')}</td>
     </tr>
     <tr>
      <th>End Date</th>
      <td>@{formatDateTime(triggerOutputs()?['body/EndDate'], 'MM/dd/yyyy')}</td>
     </tr>
     <tr>
      <th>Status</th>
      <td style="color: green;">@{triggerOutputs()?['body/Status']}</td>
     </tr>
     <tr class="total-row">
      <th>Total Amount Due</th>
      <td>₱ @{outputs('Compose_-_Calculate_Amount')}</td>
     </tr>
    </table>
   </div>
   <div class="footer">
    Generated by Power Automate on @{formatDateTime(utcNow(), 'MM/dd/yyyy HH:mm')}
   </div>
  </div>
</body>
</html>
```

2. Disbursement Voucher (Flow 3)

*Use this in **Flow 3**, Step 4 ("Creates Disbursement Voucher as HTML Table").*

```
<!DOCTYPE html>
<html>
<head>
<style>
 body { font-family: 'Segoe UI', Arial, sans-serif; color: #333; }
 .dv-container { width: 100%; border: 2px solid #333; max-width: 700px; margin: 0 auto; }
 .dv-header { background-color: #333; color: white; padding: 15px; text-align: center; font-weight: bold; font-size:
18px; }
 .dv-details { padding: 20px; }
 table { width: 100%; border-collapse: collapse; margin-bottom: 20px; }
 td, th { border: 1px solid #999; padding: 10px; text-align: left; }
 .label-col { background-color: #f0f0f0; font-weight: bold; width: 35%; }
 .amount-box { font-size: 1.2em; font-weight: bold; color: #d9534f; }
 .signatures { margin-top: 40px; display: flex; justify-content: space-between; }
 .sig-block { width: 45%; border-top: 1px solid #333; padding-top: 10px; text-align: center; }
</style>
</head>
<body>
 <div class="dv-container">
  <div class="dv-header">
   DISBURSEMENT VOUCHER
  </div>
  <div class="dv-details">
   <table>
```

```
    <tr>
     <td class="label-col">TAO ID</td>
     <td>@{outputs('Get_response_details')?['body/TAOid']}</td>
    </tr>
    <tr>
     <td class="label-col">Payee / Fullname</td>
     <td>@{outputs('Get_response_details')?['body/Fullname']}</td>
    </tr>
    <tr>
     <td class="label-col">Event Start</td>
     <td>@{formatDateTime(outputs('Get_response_details')?['body/EventStart'], 'MM/dd/yyyy')}</td>
    </tr>
    <tr>
     <td class="label-col">Event End</td>
     <td>@{formatDateTime(outputs('Get_response_details')?['body/EventEnd'], 'MM/dd/yyyy')}</td>
    </tr>
    <tr>
     <td class="label-col">Submission Date</td>
     <td>@{formatDateTime(utcNow(), 'MM/dd/yyyy')}</td>
    </tr>
    <tr>
     <td class="label-col">Total Amount Due</td>
     <td class="amount-box">₱ @{outputs('Compose_-_Calculate_DV_Amount')}</td>
    </tr>
   </table>

   <br>

   <table style="border: none;">
    <tr style="border: none;">
      <td style="border: none; text-align: center; padding-top: 50px;">
        _____<br>
        <strong>Approved By</strong>
      </td>
      <td style="border: none; text-align: center; padding-top: 50px;">
        _____<br>
        <strong>Received By</strong>
      </td>
    </tr>
   </table>
  </div>
 </div>
</body>
</html>
```

How to use these in Power Automate:

1. **Copy the code:** Copy the code block above entirely.
2. **Paste into "Compose":** In your Flow, paste the code into the **Inputs** field of your "Compose" action.
3. **Replace Dynamic Content:**
    - The parts looking like @{...} are placeholders.
    - Highlight the placeholder (e.g., @{outputs('Compose_-_Calculate_Amount')}) in the input box.
    - Delete it, and immediately select the actual **Dynamic Content** from the lightning bolt menu (e.g., select the output of your calculation step) to ensure the mapping works correctly.

Other Possible projects:

- High-Value Transaction Exception Handling
    - A branch manager submits a request to clear a transaction that exceeds the standard daily limit. This requires an audit trail and multi-team notification.
- New Merchant/Corporate Client Onboarding
    - When a new business client is signed, multiple departments (Legal, IT, Operations) need to act simultaneously to set them up.
- Loan Application Document Chasing
    - A loan officer marks an application as "Missing Documents." The flow automatically notifies the customer and creates a follow-up task, preventing the application from stalling.
- Monthly Branch Compliance Certification
    - Every month, branch managers must certify that they have audited their cash drawers and security logs.