// Set these variables according to Google Sheet and folder setup

const SPREADSHEET\_ID = '1y-dKMsyTXKTSPosWz7Cpl-QKdCdepLh0scDdSmxoW9o';

const SHEET\_NAME = 'Receipts';

const FOLDER\_ID = '1juv5hySlOL4XUfUUpp2vJqK5mdmwL0q0';

const OPERATOR\_EMAIL = 'khchan2@gmail.com'; // Operator email

function doPost(e) {

  try {

    // Log the received parameters for debugging

    Logger.log('Received parameters: ' + JSON.stringify(e.parameters));

    // Extract single values from arrays

    const email = e.parameters.email ? e.parameters.email[0] : null;

    const orderNumber = e.parameters.orderNumber ? e.parameters.orderNumber[0] : null;

    const comments = e.parameters.comments ? e.parameters.comments[0] : '';

    const file = e.parameters.file ? e.parameters.file[0] : null;

    if (!email || !file) {

      return ContentService.createTextOutput(JSON.stringify({

        success: false,

        message: 'Error: Missing required parameters.'

      })).setMimeType(ContentService.MimeType.JSON);

    }

    // Upload image to Google Drive

    const imageName = `Receipt\_${orderNumber}\_${new Date().getTime()}.jpg`;

    const imageBlob = Utilities.newBlob(Utilities.base64Decode(file), 'image/jpeg', imageName);

    const folder = DriveApp.getFolderById(FOLDER\_ID);

    const driveFile = folder.createFile(imageBlob);

    const imageUrl = driveFile.getUrl();

    // Log the file upload details for debugging

    Logger.log('File uploaded to Google Drive: ' + imageUrl);

    // Save data to Google Sheets

    const sheet = SpreadsheetApp.openById(SPREADSHEET\_ID).getSheetByName(SHEET\_NAME);

    const timestamp = new Date();

    sheet.appendRow([timestamp, email, orderNumber , comments, imageUrl]);

    // Log the data saved to Google Sheets for debugging

    Logger.log('Data saved to Google Sheets: ' + [timestamp, orderNumber, email, comments, imageUrl].join(', '));

    // Send emails

    sendOperatorEmail(orderNumber, email, comments, imageUrl, timestamp);

    sendCustomerEmail(email, orderNumber, timestamp);

    return ContentService.createTextOutput(JSON.stringify({

      success: true,

      message: 'Receipt uploaded successfully. Confirmation emails have been sent.'

    })).setMimeType(ContentService.MimeType.JSON);

  } catch (error) {

    Logger.log('Error: ' + error.toString());

    return ContentService.createTextOutput(JSON.stringify({

      success: false,

      message: 'Error: ' + error.toString()

    })).setMimeType(ContentService.MimeType.JSON);

  }

}

function sendOperatorEmail(orderNumber, customerEmail, comments, imageUrl, timestamp) {

  const subject = `New Receipt Upload - Order #${orderNumber}`;

  const body = `

    A new receipt has been uploaded:

    Order Number: ${orderNumber}

    Customer Email: ${customerEmail}

    Timestamp: ${timestamp.toLocaleString()}

    Comments: ${comments}

    Image URL: ${imageUrl}

    Please review the uploaded receipt at your earliest convenience.

  `;

  MailApp.sendEmail(OPERATOR\_EMAIL, subject, body);

}

function sendCustomerEmail(customerEmail, orderNumber, timestamp) {

  const subject = `Receipt Upload Confirmation - Order #${orderNumber}`;

  const body = `

    Dear Customer,

    We have received your receipt upload for Order #${orderNumber} on ${timestamp.toLocaleString()}.

    Thank you for submitting your receipt. Our team will review it shortly.

    If you have any questions, please don't hesitate to contact our customer support.

    Best regards,

    TPO Wellness Store Team

  `;

  // Correcting the method signature to match the expected parameters

  MailApp.sendEmail({

    to: customerEmail,

    subject: subject,

    body: body

  });

}

function doGet() {

  return HtmlService.createHtmlOutput('<h1>This is a web app for handling receipt uploads.</h1>');

}