User Guide: Leave Request & Attendance System with Email & Calendar Automation

Part 1: Creating the Google Form

Purpose:
Collect leave request data from employees.

Steps:
1. Go to Google Forms.
2. Click Blank to create a new form.
3. Add the following fields:
- Employee Name (Short answer)
- Email Address (Short answer - enable email validation)
- Leave Type (Dropdown - e.g., Annual, Sick, Maternity)
- Start Date (Date)
- End Date (Date)
- Reason for Leave (Paragraph)
4. Under Settings > Responses, enable Collect email addresses.
5. Link responses to Google Sheets via the Responses tab > Link to Sheets.

Part 2: Setting Up the Google Sheet

Your form responses will be saved in a linked Google Sheet.

Additional Columns to Add:
- Approval Status (Dropdown: Approved, Rejected, Pending)
- Feedback Status (Initially blank to track email sent status)

Part 3: Email Automation Using Apps Script

Steps to Add Script:
1. Open the linked Google Sheet.
2. Click Extensions > Apps Script.
3. Paste the following script:

```
function sendLeaveFeedback() {
  const sheet = SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  const data = sheet.getDataRange().getValues();
  const header = data[0];

  const emailCol = header.indexOf("Email Address");
  const statusCol = header.indexOf("Approval Status");
  const sentCol = header.indexOf("Feedback Status");
  const nameCol = header.indexOf("Employee Name");
  const leaveCol = header.indexOf("Leave Type");

  for (let i = 1; i < data.length; i++) {
    const email = data[i][emailCol];
    const status = data[i][statusCol];
    const sent = data[i][sentCol];
    const name = data[i][nameCol];
    const leave = data[i][leaveCol];
```

```
    if (email && status && !sent && name && leave) {
      let subject = `Leave Request ${status}`;
      let htmlMessage = "";

      if (status === "Approved") {
                              htmlMessage = `Dear ${name},<br><br>Your ${leave} leave request has been
<b>APPROVED</b>.<br><br>Regards,<br>HR Department`;
      } else if (status === "Rejected") {
                              htmlMessage = `Dear ${name},<br><br>Your ${leave} leave request has been
<b>REJECTED</b>.<br><br>Regards,<br>HR Department`;
      } else if (status === "Pending") {
          htmlMessage = `Dear ${name},<br><br>Your ${leave} leave request is currently <b>PENDING</b>. Please visit
the HR office for clarification.<br><br>Regards,<br>HR Department`;
      } else {
        continue;
      }

      GmailApp.sendEmail(email, subject, "", {htmlBody: htmlMessage});
      sheet.getRange(i + 1, sentCol + 1).setValue("Sent");
    }
  }
}
```

Part 4: Automating the Script

Steps to Set Time Trigger:
1. In Apps Script, go to Triggers > + Add Trigger.
2. Set the following:
- Function to run: sendLeaveFeedback
- Event source: Time-driven
- Type: Hourly or daily depending on your needs

Part 5: Attendance Form Setup (Optional)

Steps:
1. Create another Google Form for attendance:
- Employee Name
- Date (or use form timestamp)
- Check-In Time
- Check-Out Time
2. Link it to a new Sheet.
3. Use formulas/pivot tables to analyze daily or monthly attendance.

Part 6: Google Calendar Integration for Approved Leave

Purpose:
Automatically add approved leave requests to a shared Google Calendar.

Steps to Add Google Calendar Integration:

User Guide: Leave Request & Attendance System with Email & Calendar Automation

1. Create or Identify Your Shared Calendar:
- Go to Google Calendar.
- Create a new calendar called "Leave Tracker".
- Share it with your team if needed.

2. Add a New Script for Calendar Automation:

```
function syncLeaveCalendar() {
  const calendarName = "Leave Tracker";
  const calendar = CalendarApp.getCalendarsByName(calendarName)[0];

  if (!calendar) return;

  const sheet = SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  const data = sheet.getDataRange().getValues();
  const header = data[0];

  const nameCol = header.indexOf("Employee Name");
  const statusCol = header.indexOf("Approval Status");
  const startCol = header.indexOf("Start Date");
  const endCol = header.indexOf("End Date");
  const calendarStatusCol = header.indexOf("Calendar Status");
  const leaveCol = header.indexOf("Leave Type");
  const timeZone = Session.getScriptTimeZone();

  const futureEvents = calendar.getEvents(new Date(), new Date("2100-01-01"));

  for (let i = 1; i < data.length; i++) {
    const name = data[i][nameCol];
    const status = data[i][statusCol];
    const startDate = new Date(data[i][startCol]);
    const endDate = new Date(data[i][endCol]);
    const calendarStatus = data[i][calendarStatusCol];
    const leave = data[i][leaveCol];

    const title = `${leave} Leave - ${name}`;

    if (status === "Approved" && name && startDate && endDate && calendarStatus !== "Event Created") {
      const exists = calendar.getEvents(startDate, new Date(endDate.getTime() + 24*60*60*1000), { search: title })
        .some(event => event.getTitle() === title);
      if (!exists) {
        calendar.createAllDayEvent(title, startDate, new Date(endDate.getTime() + 24*60*60*1000));
        sheet.getRange(i + 1, calendarStatusCol + 1).setValue("Event Created");
      }
    }

    if (calendarStatus === "Event Created" && status !== "Approved") {
      const matchingEvents = futureEvents.filter(event =>
        event.getTitle() === title &&
        Utilities.formatDate(event.getStartTime(), timeZone, "yyyy-MM-dd") ===
```

```
        Utilities.formatDate(startDate, timeZone, "yyyy-MM-dd")
      );
      matchingEvents.forEach(event => event.deleteEvent());
      sheet.getRange(i + 1, calendarStatusCol + 1).clearContent();
    }
  }
}
```

3. Add Calendar Status Column in your Sheet.
4. Set a Time Trigger to Run syncLeaveCalendar regularly.

Best Practices

- Use separate Sheets or tabs for Leave Requests and Attendance.
- Protect columns like Approval Status and Feedback Status from accidental edits.
- Use Data Validation in Google Sheets for clean input.
- Back up your data regularly.
- Build a dashboard with Google Sheets charts or Google Looker Studio for insights.
- Test with sample data before rolling out organization-wide.