function sendDutyReminders() {

const ss = SpreadsheetApp.getActiveSpreadsheet();

// ====== LOG SHEET SETUP ======

let logSheet = ss.getSheetByName("EmailLog");

if (!logSheet) {

logSheet = ss.insertSheet("EmailLog");

logSheet.appendRow(["Timestamp", "Email Type", "Recipient Name", "Recipient Email", "Status", "Message"]);

}

function logEmail(type, name, email, status, message) {

logSheet.appendRow([new Date(), type, name || "", email || "", status, message || ""]);

}

// ====== MAIN SHEET ======

const sheet = ss.getSheetByName("Sheet1"); // Change if needed

const data = sheet.getDataRange().getValues();

const headers = data[0];

// ====== DATE DETECTION ======

const today = new Date();

const tomorrow = new Date(today.getFullYear(), today.getMonth(), today.getDate() + 1);

const dateColIndex = headers.findIndex(h => {

if (!h) return false;

let headerDate;

if (Object.prototype.toString.call(h) === "[object Date]" && !isNaN(h)) {

headerDate = new Date(h.getFullYear(), h.getMonth(), h.getDate());

} else {

let parsed = new Date(h);

if (!isNaN(parsed)) {

headerDate = new Date(parsed.getFullYear(), parsed.getMonth(), parsed.getDate());

}

}

return headerDate && headerDate.getTime() === tomorrow.getTime();

});

if (dateColIndex === -1) {

logEmail("System", "", "", "Failed", "Tomorrow's date column not found");

return;

}

// ====== FIND VOLUNTEERS ======

const validMarkers = ["yes", "y", "available", "ok", "present", "1"];

let volunteers = [];

for (let i = 1; i < data.length; i++) {

if (data[i][dateColIndex]) {

let val = data[i][dateColIndex].toString().trim().toLowerCase();

if (validMarkers.includes(val)) {

volunteers.push({

name: data[i][0],

email: data[i][2]

});

}

}

}

// ====== SEND REMINDERS ======

volunteers.forEach(v => {

if (v.email) {

try {

GmailApp.sendEmail(

v.email,

"Gentle Reminder for Tomorrow Duty",

"Dear " + v.name + ",\n\nA gentle reminder for tomorrow's duty.\n\nThank you."

);

logEmail("Reminder", v.name, v.email, "Sent", "");

} catch (err) {

logEmail("Reminder", v.name, v.email, "Failed", err.message);

}

}

});

// ====== READ COORDINATORS ======

const coordSheet = ss.getSheetByName("Coordinators");

const coordData = coordSheet.getRange(2, 1, coordSheet.getLastRow() - 1, 1).getValues();

let coordinators = coordData.flat().filter(e => e);

// ====== SEND SINGLE SUMMARY EMAIL WITH CC ======

if (coordinators.length > 0) {

let volunteerList = volunteers.map(v => v.name + " (" + v.email + ")").join("\n");

if (!volunteerList) volunteerList = "No volunteers available.";

let toEmail = coordinators[0];

let ccEmails = coordinators.slice(1).join(",");

try {

GmailApp.sendEmail(

toEmail,

"List of Volunteer(s) on duty tomorrow",

"Hi, \n\nBelow is the list of Volunteer(s) available for duty tomorrow:\n\n" + volunteerList,

{ cc: ccEmails }

);

logEmail("Summary", "", toEmail + (ccEmails ? ", " + ccEmails : ""), "Sent", "");

} catch (err) {

logEmail("Summary", "", toEmail + (ccEmails ? ", " + ccEmails : ""), "Failed", err.message);

}

}

}

function sendCustomCoordinatorEmail() {

const ss = SpreadsheetApp.getActiveSpreadsheet();

// ==== Get coordinators ====

const coordSheet = ss.getSheetByName("Coordinators");

if (!coordSheet) {

Logger.log("Coordinators sheet not found.");

return;

}

const coordinatorEmails = coordSheet

.getRange(2, 1, coordSheet.getLastRow() - 1, 1)

.getValues()

.flat()

.filter(email => email && email.toString().trim() !== "");

if (coordinatorEmails.length === 0) {

Logger.log("No coordinator emails found.");

return;

}

// ==== Get volunteers for tomorrow ====

const rosterSheet = ss.getSheetByName("Sheet1"); // change if needed

const headerRow = rosterSheet.getRange(1, 1, 1, rosterSheet.getLastColumn()).getValues()[0];

const nameColIndex = headerRow.findIndex(h => h.toString().toLowerCase().trim() === "name");

const phoneColIndex = headerRow.findIndex(h => h.toString().toLowerCase().trim() === "mobile number");

const emailColIndex = headerRow.findIndex(h => h.toString().toLowerCase().trim() === "email");

if (nameColIndex === -1 || phoneColIndex === -1 || emailColIndex === -1) {

Logger.log("One or more required columns ('Name', 'Mobile Number', 'Email') not found.");

return;

}

const tomorrow = new Date();

tomorrow.setDate(tomorrow.getDate() + 1);

const tomorrowColIndex = headerRow.findIndex(h => {

const parsedDate = new Date(h);

return !isNaN(parsedDate) && parsedDate.toDateString() === tomorrow.toDateString();

});

if (tomorrowColIndex === -1) {

Logger.log("No column found for tomorrow's date.");

return;

}

const data = rosterSheet.getRange(2, 1, rosterSheet.getLastRow() - 1, rosterSheet.getLastColumn()).getValues();

const volunteers = data

.filter(row => row[tomorrowColIndex] && row[tomorrowColIndex].toString().trim() !== "")

.map(row => ({

name: row[nameColIndex],

phone: row[phoneColIndex]

}));

// ==== Build email body dynamically ====

let body = "Hi,\n\n";

body += "Please add the below mentioned Volunteer(s) in WhatsApp Communication group:\n\n";

if (volunteers.length > 0) {

volunteers.forEach((v, i) => {

body += `${i + 1}. ${v.name} — ${v.phone}\n`;

});

} else {

body += "No volunteers are currently marked for tomorrow.\n";

}

body += "\nBest regards,\nABC”;

// ==== Send email ====

const to = coordinatorEmails[0];

const cc = coordinatorEmails.slice(1).join(",");

GmailApp.sendEmail(to, "List of Volunteer(s) on duty tomorrow", body, {

cc: cc

});

Logger.log(`Dynamic email sent to coordinators: ${to} + CC: ${cc}`);

}