**Phase5**

**1️⃣ Validation Rule — Prevent Past Dates**

**Purpose:** Users cannot schedule appointments in the past.

**Object:** Appointment

**Rule Name:** [Appointment\_Date\_Validation](https://orgfarm-8272ae4574-dev-ed.develop.my.salesforce-setup.com/lightning/setup/ObjectManager/01Ig50000009xiP/ValidationRules/03dg50000005MkfAAE/view)**Formula:**

Appointment\_Date\_\_c < TODAY()

**Error Message:** “Appointment date must be today or in the future.”

**Error Location:** Appointment Date field

✅ Users can only schedule future appointments.

**2️⃣ Flow — Auto Confirm Appointment**

**Purpose:** Automatically confirm appointment if doctor is available, otherwise mark as rescheduled.

**Flow Name:** Auto\_Confirm\_Appointment\_Simple

**Type:** Record-Triggered Flow

**Object:** Appointment

**Trigger:** Created or Updated

**Conditions:** Status\_\_c != 'Cancelled'

**Optimize For:** Actions and Related Records

**Flow Elements:**

| **Step** | **Element** | **Details** |
| --- | --- | --- |
| 1 | Start | Trigger: Appointment Created/Updated, Status != Cancelled |
| 2 | Get Doctor Appointments (Optional) | Filter: Doctor\_\_c = $Record.Doctor\_\_c, Status\_\_c != 'Cancelled' |
| 3 | Decision — Doctor Availability | Outcomes: Available → Scheduled, Unavailable → Rescheduled |
| 4 | Update Appointment Status | Scheduled Path → Status\_\_c = 'Scheduled' Rescheduled Path → Status\_\_c = 'Rescheduled' Use $Record as record to update |
| 5 | Send Confirmation Email (Scheduled Only) | Use Email Alert or Flow Send Email Action Recipient: Patient & Doctor emails Body: appointment details |

✅ This flow ensures appointments are automatically scheduled, rescheduled, and notifications are sent.

**3️⃣ Flow — Appointment Completion Automation**

**Purpose:** When an appointment is marked Completed, create a Visit record and update Doctor’s appointment count.

**Objects:**

**Visit (Custom Object)**

* Label: Visit
* Plural Label: Visits
* Record Name: Visit Number (Auto Number, format VISIT-{0000})
* Fields:
  + Appointment\_\_c (Lookup → Appointment)
  + Patient\_\_c (Lookup → Contact)
  + Doctor\_\_c (Lookup → User or Contact)
  + Visit Date\_\_c (Date/Time)
  + Notes\_\_c (Text Area)

**Appointment**

* Status\_\_c = Completed triggers the flow

**Flow Elements:**

| **Step** | **Element** | **Details** |
| --- | --- | --- |
| 1 | Start | Trigger: Appointment Updated, Status = Completed |
| 2 | Create Visit Record | Map fields: Appointment\_\_c = $Record.Id, Patient\_\_c = $Record.Patient\_\_c, Doctor\_\_c = $Record.Doctor\_\_c, Visit Date\_\_c = $Record.Appointment\_Date\_\_c |
| 3 | Update Doctor Appointment Count | Object: Doctor, Field: Current\_Appointment\_Count\_\_c += 1 |
| 4 | End | Flow complete |

✅ Tracks completed appointments as Visits and updates Doctor metrics.

**4️⃣ Apex Scheduled Class — Daily Doctor Schedule**

**Purpose:** Send doctors a daily summary of their next day’s appointments.

**Implementation:** Apex code instead of Scheduled Flow

**Apex Class:**

global class DailyDoctorScheduleEmail implements Schedulable {

global void execute(SchedulableContext sc) {

Date tomorrow = Date.today().addDays(1);

DateTime startOfDay = DateTime.newInstance(tomorrow, Time.newInstance(0, 0, 0, 0));

DateTime endOfDay = DateTime.newInstance(tomorrow, Time.newInstance(23, 59, 59, 999));

List<Appointment\_\_c> appointments = [

SELECT Id, Appointment\_Date\_\_c, Appointment\_Time\_\_c, Reason\_\_c,

Patient\_\_r.Name, Doctor\_\_c, Doctor\_\_r.Email\_\_c

FROM Appointment\_\_c

WHERE Appointment\_Date\_\_c >= :startOfDay

AND Appointment\_Date\_\_c <= :endOfDay

AND Status\_\_c != 'Cancelled'

];

Map<Id, List<Appointment\_\_c>> doctorAppointments = new Map<Id, List<Appointment\_\_c>>();

for (Appointment\_\_c app : appointments) {

if (!doctorAppointments.containsKey(app.Doctor\_\_c)) {

doctorAppointments.put(app.Doctor\_\_c, new List<Appointment\_\_c>());

}

doctorAppointments.get(app.Doctor\_\_c).add(app);

}

List<Messaging.SingleEmailMessage> emails = new List<Messaging.SingleEmailMessage>();

for (Id doctorId : doctorAppointments.keySet()) {

List<Appointment\_\_c> apps = doctorAppointments.get(doctorId);

String body = 'Hello Doctor,\n\nHere are your appointments for tomorrow:\n\n';

for (Appointment\_\_c a : apps) {

body += 'Patient: ' + a.Patient\_\_r.Name +

' | Time: ' + String.valueOf(a.Appointment\_Time\_\_c) +

' | Reason: ' + a.Reason\_\_c + '\n';

}

body += '\nRegards,\nHospital Admin';

Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();

mail.setToAddresses(new String[] { apps[0].Doctor\_\_r.Email\_\_c });

mail.setSubject('Tomorrow\'s Appointments');

mail.setPlainTextBody(body);

emails.add(mail);

}

if (!emails.isEmpty()) {

Messaging.sendEmail(emails);

}

}

}

**Scheduled:** Everyday at 8 PM

✅ Sends one consolidated email per doctor listing all appointments.

**5️⃣ Email Alert / Flow — Appointment Cancelled**

**Purpose:** Notify patient and doctor when an appointment is cancelled.

**Implementation:** Flow → Send Email Action

**Trigger:** Appointment updated, Status = Cancelled

**To Addresses:** {!$Record.Doctor\_\_r.Email\_\_c};{!$Record.Patient\_\_r.Email\_\_c}

**Subject:** Appointment Cancelled – {!$Record.Appointment\_Date\_\_c}

**Body (HTML):**

<p>Hello {!$Record.Patient\_\_r.First\_Name\_\_c},</p>

<p>We regret to inform you that your appointment scheduled on <strong>{!$Record.Appointment\_Date\_\_c}</strong>

with Dr. {!$Record.Doctor\_\_r.First\_Name\_\_c} has been cancelled.</p>

<p>If you wish to reschedule, please contact us at your earliest convenience.</p>

<p>Thank you,<br/>

Hospital Appointment System</p>

✅ Ensures that cancellation notices always reach recipients.