MedHub - Hospital Integration Guide (EHR-linked)

Purpose: Enable secure, reliable integration between a hospital's EHR and MedHub to support appointment scheduling, symptom tracking, patient sync, clinician notifications, and reporting.

1. Executive summary

MedHub integrates with hospital systems to:

- Synchronize patient demographics and consent.
- Create, update, and cancel appointments.
- Send/receive symptom tracking updates (questionnaires, observations).
- Surface clinician alerts and dashboards inside the EHR or clinicians' workflows.
- Maintain auditable logs for compliance.

Primary standards used: **FHIR R4** (preferred), **HL7v2** (legacy), or vendor APIs. Authentication: **OAuth2 (with scopes)** and TLS. Compliance considerations: **HIPAA / GDPR / country-specific regs**.

2. Integration architecture (high level)

1. **MedHub Backend** — hosts APIs, webhook endpoints, scheduler, analytics, and user-facing UI.
2. **EHR System** — FHIR server or legacy HL7 interface.
3. **Integration Layer / Middleware (optional)** — n8n, Mirth Connect, or hospital ESB to transform and route messages.
4. **Auth & Consent Manager** — manages OAuth2 client creds, scopes, and user consent.
5. **Monitoring & Logging** — central logs, metrics, alerting.

Diagram (conceptual):
MedHub ⇄ (FHIR REST / OAuth2) ⇄ EHR
MedHub ⇄ (Webhooks / HL7) ⇄ Middleware ⇄ EHR
Patient mobile/web app ⇄ MedHub (HTTPS/TLS)

3. Prerequisites & project roles

Hospital prerequisites

- EHR admin contact + API/IT access.
- FHIR server endpoints (or HL7 interface specs).
- Sandbox/test environment credentials.
- Data protection officer / compliance signoff.

MedHub prerequisites

- OAuth2 client credentials per hospital.
- Tenant configuration for hospital-specific mappings.
- Webhook URL(s) reachable via TLS.

Key roles

- Clinical lead (validates workflows).
- IT/Integration engineer (implements integration).
- Security officer (audits, certs).
- Product manager (coordinates rollout).

4. Recommended data standards & resources

Primary (modern):

- FHIR R4 resources: `Patient`, `Practitioner`, `Appointment`, `Slot`, `Encounter`, `Observation`, `Questionnaire`, `QuestionnaireResponse`, `Condition`, `Procedure`, `Consent`.

Fallback / legacy:

- HL7 v2 ADT (for patient demographics), ORU (observations), SIU (scheduling).

Other protocols

- SMART on FHIR (for launching MedHub in EHR context).
- OAuth2 + OpenID Connect (authentication & delegated consent).
- Webhooks (for asynchronous notifications).

5. Core workflows & mapping

5.1 Patient creation / sync

Trigger: Patient registers on MedHub or is created in EHR.

FHIR

- MedHub `POST /Patient` to EHR FHIR server (or `PUT` if updating).
- Key fields mapping:
  - `Patient.identifier` → hospital MRN, national ID (NIN/BVN), mobile phone.
  - `Patient.name`, `Patient.birthDate`, `Patient.gender`.
  - `Patient.telecom` → phone/email.
  - `Patient.address`.

HL7v2

- ADT^A28 / A31 messages for create/update.

Notes

- Reconcile duplicates using identifiers (MRN, national ID, phone + DOB).
- Store EHR patient ID in MedHub as `externalId` for future reference.

5.2 Appointment scheduling (two-way)

Use-case: Patient books via MedHub; appointment appears in EHR calendar and triggers reminders.

FHIR

- Create `Slot` and `Appointment` resources:
  - `Slot` identifies available time.
  - `Appointment` resource includes `participant` referencing `Patient` and `Practitioner`, `status`, and `start`/`end`.
- MedHub `POST /Appointment` → EHR FHIR endpoint.
- EHR may respond with `Appointment` id — store as `externalAppointmentId`.

HL7v2

- SIU^S12 or ORM scheduling messages.

Edge cases

- Double-booking: EHR to respond with conflict status (409) — MedHub shows alternative slots.
- Cancellations & reschedules: send `Appointment` update with `status` field (`cancelled`, `fulfilled`, `noshow`).

5.3 Symptom tracking & remote monitoring

Use-case: Patients complete symptom questionnaires; data should flow to EHR as Observations or QuestionnaireResponses.

FHIR mapping

- `Questionnaire` (template) stored in MedHub; patient answers sent as `QuestionnaireResponse`.
- For numeric/vital sign data, map to `Observation` resources with proper `code` (LOINC codes where possible).
- Example:
  - Symptom severity: Observation.code = LOINC for pain severity (if applicable).
  - Temperature: Observation with `component` fields.

Workflow

- MedHub `POST /QuestionnaireResponse` → EHR or middleware transforms to `Observation` for clinician consumption.
- Optionally create an `Encounter` or link to existing appointment/encounter.

5.4 Clinician notifications & alerts

- Create EHR `Task` or use native notification APIs to alert clinicians when:
  - Symptom exceeds threshold.
  - Patient flags “red” condition.
- MedHub triggers webhook to hospital middleware with `event` payload; middleware routes to EHR inbox.

5.5 Audit & consent

- On patient consent, create `Consent` resource in FHIR: store scope (share with MedHub, share with third parties), timestamp, and signer.
- All patient data exchanges must reference Consent.

6. Security, privacy, and compliance

6.1 Transport & encryption

- All transport via **HTTPS/TLS 1.2+**.
- Use **mutual TLS** if required by hospital.
- Store encryption at rest (AES-256 or equivalent).

6.2 Authentication & authorization

- **OAuth2 (Client Credentials or Authorization Code)** — hospital issues client_id/client_secret.
- Use **SMART on FHIR** when linking clinician sessions inside EHR (patient-level delegation).
- Scopes should be minimal (e.g., `patient/Observation.write`, `patient/Appointment.write`, `patient/Patient.read`).

6.3 Access control & least privilege

- Role-based access control inside MedHub (clinician, admin, support).
- Audit logs for all CRUD operations with user id, timestamp, and IP.

6.4 Data residency & retention

- Follow hospital/country policies for data residency (store PII within country if required).

- Implement configurable retention policy per hospital.

6.5 Compliance

- Align with **HIPAA**, **GDPR**, **NDPR** (Nigeria) as applicable.
- Perform periodic security assessments and penetration tests.

7. API & webhook design (examples)

7.1 OAuth2 token endpoint (example)

`POST /oauth/token`
Payload: `grant_type=client_credentials&client_id=...&client_secret=...`
Response:

```
{
  "access_token": "eyJ..",
  "token_type": "Bearer",
  "expires_in": 3600,
  "scope": "patient/Patient.write patient/Appointment.write"
}
```

7.2 Create Appointment (FHIR-style)

`POST /fhir/Appointment` (Authorization: Bearer token)

```
{
  "resourceType": "Appointment",
  "status": "proposed",
  "type": { "text": "Outpatient Consultation" },
  "start": "2025-10-02T09:00:00+01:00",
  "end": "2025-10-02T09:20:00+01:00",
  "participant": [
    { "actor": { "reference": "Patient/123", "display": "Jane Doe" } },
    { "actor": { "reference": "Practitioner/456", "display": "Dr. A" } }
  ],
  "description": "MedHub booking via mobile app"
}
```

7.3 Symptom webhook payload (MedHub → hospital)

POST to hospital webhook `/medhub/events`

```
{
  "event": "questionnaire_response.created",
  "timestamp": "2025-09-02T10:35:00Z",
  "data": {
    "patientExternalId": "MB-000123",
    "questionnaireId": "symptoms-v1",
    "responses": [
      { "code": "cough", "answer": "yes" },
      { "code": "fever_temp", "answer": "38.5", "unit": "C" }
    ]
  }
}
```

8. Error handling & retry policy

- **Idempotency**: use `Idempotency-Key` headers for operations (appointments, patient create) to avoid duplicates.
- **HTTP codes**:
  - `200/201` success
  - `202` accepted (async)
  - `400` bad request — return helpful validation errors
  - `401` unauthorized — refresh token / check scopes
  - `409` conflict — resource already exists
  - `429` rate limit — implement exponential backoff
  - `500` server errors — retry with backoff

- Retries: exponential backoff with jitter, limit retries to 5 attempts.

9. Testing strategy

Environment setup

- Use hospital sandbox EHR environment + MedHub staging tenant.
- Create test patient sets covering edge cases (no national ID, duplicate phone, minors).

Test types

- Unit tests (API contract).
- Integration tests (end-to-end create appointment → EHR stores appointment).
- Security tests (token rotation, scope testing).
- Load tests (peak booking volumes).
- Clinical acceptance testing (clinicians validate that symptom flows are actionable and fit workflows).

UAT checklist

- Patient data appears correctly in EHR.
- Appointment updates propagate both ways.
- Alerts from symptom thresholds appear in clinician inbox.
- Consent appears and limits data flow as expected.

10. Deployment & rollout plan (recommended)

Phase 0 — Discovery (1–2 weeks)

- Map systems, confirm FHIR endpoints, authentication, SLAs.
- Agree on required data retention & residency.

Phase 1 — Sandbox Integration (2–4 weeks)

- Implement OAuth2 handshake.
- Test Patient create/update and Appointment create flow.
- Test QuestionnaireResponse → Observation pipeline.

Phase 2 — Pilot (4–8 weeks)

- Integrate with 1–2 clinical departments (e.g., outpatient & triage).
- Collect clinician feedback; iterate.

Phase 3 — Gradual Rollout (2–3 months)

- Expand to more departments.
- Provide training sessions for clinical staff and support.

Phase 4 — Full Production

- Go live across hospital with monitoring, SLA, and support rota.

11. Monitoring, SLA & KPIs

Monitoring

- Uptime dashboards for MedHub APIs and webhook delivery.
- Error rate, average response time.
- Audit logs for data access.

SLA

- API availability: 99.9% (or per hospital agreement).
- Response time for appointment create: <1s in successful cases.

KPIs

- Appointment sync success rate (%).
- Median time from patient booking to EHR appointment visibility.
- Symptom alert-to-clinician action time.
- Number of duplicate patient records created (goal: 0).

12. Operational considerations

- **Support**: 24/7 incident escalation for critical outages.
- **Backup & DR**: daily backups and tested restore plan; disaster recovery RTO/RPO agreed.
- **Training**: clinician onboarding sessions, admin docs, quick-reference guides.
- **Change control**: versioned APIs and scheduled maintenance windows.

13. Integration checklist (for IT)

- Obtain hospital sandbox FHIR endpoint & OAuth2 credentials.
- Configure MedHub tenant with hospital identifiers and callback URLs.
- Implement PATIENT create/update mapping & run test sync.
- Implement Appointment create/update/cancel flows; test reschedule and cancellation.
- Implement Questionnaire/Observation mapping; test threshold alerts.
- Implement Consent recording & enforcement.

- Validate logging, audit trails, and retention settings.
- Complete security & penetration testing.
- Run clinician UAT and address feedback.
- Agree rollout plan & go-live schedule.

14. Appendix — FHIR resource quick references

Patient

```
{
  "resourceType":"Patient",
  "id":"123",
  "identifier":[{ "system":"http://hospital.example/mrn", "value":"MRN-00123"}],
  "name":[{"family":"Doe","given":["Jane"]}],
  "gender":"female",
  "birthDate":"1994-05-14"
}
```

Appointment — see section 7.2 example.

Observation

```
{
 "resourceType":"Observation",
 "status":"final",

"category":[{"coding":[{"system":"http://terminology.hl7.org/CodeSystem/observation-category","code":"vital-signs"}]}],
 "code":{"coding":[{"system":"http://loinc.org","code":"8310-5","display":"Body temperature"}]},
 "subject":{"reference":"Patient/123"},
 "effectiveDateTime":"2025-09-02T09:00:00Z",
 "valueQuantity":{"value":38.5,"unit":"C","system":"http://unitsofmeasure.org","code":"Cel"}
}
```

QuestionnaireResponse

- Store raw `QuestionnaireResponse` in EHR or transform to `Observation` as needed.

15. Final notes & next steps

1. **Kickoff**: schedule a technical kickoff with hospital EHR team, MedHub engineering, and clinical lead.

2. **Sandbox tests**: run end-to-end scenarios including edge cases (offline patients, partially complete data).

3. **Pilot**: start small (1 department) and iterate based on clinician & admin feedback.

4. **Governance**: sign off on data-sharing agreements, consent forms, and incident response procedures.