§ 170.315(g)(9) Application Access – All Data

USCDI v3 C-CDA Export via Http Post or GraphQl API

1. Purpose and Scope (Application Access – All Data)

Purpose

- **Objective:** Provide ONC G9 "Application Access – All Data" export for a single patient.
- **Output:** A USCDI v3-compliant C-CDA XML document representing all available clinical data for the patient, constrained by optional date range.
- **Intended Use:** Third-party applications acting on behalf of a patient or authorized user to request an "all data" export via API.

Scope of Data (high-level)

The generated C-CDA document includes, at minimum, the following USCDI v3 data classes (subject to availability in source FHIR system):

- **Demographics:** Patient identity and core demographics.
- **Clinical data:**
  - Encounters
  - Problems/Conditions
  - Allergies and intolerances
  - Medications (statements and orders)
  - Immunizations (including vaccine-like medications)
  - Laboratory and clinical Observations
  - Procedures
  - Care teams
  - Devices (including implantable devices)
  - Coverage / Insurance

- **G9 / Application-Access–specific content:**
  - MedicationRequests (orders)
  - DocumentReferences (clinical and administrative documents)
  - Compositions (clinical document metadata)
  - Goals
  - CarePlans
  - DiagnosticReports
  - Provenance
  - Health concerns
  - QuestionnaireResponses
  - EpisodeOfCares
  - Appointments
  - Targeted Observations for:
    - Smoking status (LOINC `72166-2` or display containing "smoking")
    - Pregnancy status (LOINC `82810-3` or display containing "pregnancy")
    - Social Determinants of Health (selected LOINC codes: `76513-1`, `71802-3`, `75274-0`, `76514-9`, `82589-3`, or display containing "social determinant")
    - Functional status (display containing "functional", "adl", or "iadl")
    - Mental/cognitive status (display containing "mental", "cognitive", "mmse", "moca")

The resulting document is built via the USCDI v3 C-CDA builder with `isApplicationAccess = true`, ensuring inclusion of all G9-specific sections.

2. Endpoint Definition

Transport

- **Protocol:** HTTPS
- **Style:** GraphQL over HTTP

HTTP Endpoint

- **Method:** `POST`
- **URL:** `https://<APPROVED_DOMAIN>/graphql`
  - Publicly reachable over the internet (no login page), but requires API authentication (see below).

GraphQL Mutation

```
mutation generateG9Document($patientId: String!,
  $startDate: String,
    $endDate: String) {
  generateG9Document(patientId: $patientId,
    startDate: $startDate,
      endDate: $endDate) {
        ccdaUrl
        isCCDAFileExists
  }
}
```

3. Required Software Components and Configuration

The following software components and configurations are necessary for an application to successfully interact with the API and process its responses, as required by 170.315(g)(9)(ii)(A)(2).

3.1 HTTP Client Requirements

- **HTTPS Support:** Client must support TLS 1.2 or higher for secure communication
- **HTTP/1.1 or HTTP/2:** Compatible with standard HTTP protocols
- **Required Headers:**
  - `Content-Type: application/json` – Required for GraphQL requests
  - `Authorization: Bearer <access_token>` – OAuth 2.0 bearer token authentication
  - `Woundtech-id-token: <id_token>` – OAuth 2.0 id token for identification
  - `Accept: application/json` – Expected response format
- **Request Method:** Must support HTTP POST
- **Timeout Configuration:** Recommended minimum 30 seconds for document generation

3.2 GraphQL Client Configuration

- **GraphQL over HTTP:** Client must support GraphQL operations over HTTP POST
- **Variables Support:** Must support GraphQL variables in request body format:

```
{
  "query": "mutation...",
  "variables": {
    "patientId": "string",
    "startDate": "string",
    "endDate": "string"
```

- **Error Handling:** Must parse GraphQL error responses in standard format
- **Response Processing:** Must handle nested JSON response structure

3.3 OAuth 2.0 Client Implementation

- **OAuth 2.0 Library:** Standards-compliant OAuth 2.0 client implementation
- **Authorization Flows:**
  - **User_Password Auth**
- **Token Management:**
  - Secure token storage (encrypted at rest)
  - Automatic token refresh capabilities
  - Token expiration handling
- **Scope Validation:** Support for FHIR resource scopes and patient context built into user role FHIR resource

3.4 File Download and Processing Components

- **HTTP Download Client:** Support for downloading from pre-signed URLs with proper timeout handling
- **XML Processing:** C-CDA XML parsing capabilities for USCDI v3 data extraction
- **URL Handling:** Handle time-limited pre-signed URLs with expiration management

4. Authentication and Authorization Overview

Authentication

- **Mechanism:** Standard API authentication using the endpoint using username and password provided after registration.

```
curl --location "https://emr.woundtech.net/getG9Token" \
--header "Content-Type: application/json" \
--data-raw "{
        \"username\": \"testuser@test.com\",
        \"password\": \"****\"
}"
```

Sample Response:

```
{
      "AuthenticationResult": {
              "AccessToken": "ey.....AA",
              "ExpiresIn": 3600,
              "IdToken": "ey....wg",
              "TokenType": "Bearer"
              }
      }
```

4.1 Obtaining an Access Token

- **Token type:** Short-lived OAuth2 bearer access token issued by the organization's authorization server
- **Grant types:**
  - USER_PASSWORD_AUTH
- **Required scopes:** Tokens must include scopes for patient data read access and USCDI v3 Application Access export functionality
- **Client registration:** Clients to contact over email for application onboarding, to generate username / password.
- **Token management:** Applications must handle token expiration and implement secure token storage

4.2 Configuration Requirements

Applications require basic configuration for:

- **API Endpoints:** GraphQL endpoint URL and OAuth 2.0 server URLs
- **Client Credentials:** username and password (provided during registration)
- **Security Settings:** TLS/SSL validation and timeout configurations

5. Request Parameters

GraphQL Variables

`patientId`

- **Type:** `String!` (required, non-null)
- **Description:** Identifier of the patient for whom the USCDI v3 Application Access C-CDA document should be generated.
- **Validation:**
  - Empty string or whitespace-only value is rejected.
  - If blank or only whitespace, API throws `BadRequestException` with message: `"Patient ID is required for USCDI v3 Application Access document generation"`.

`startDate`

- **Type:** `String` (optional)
- **Format:** Recommended: `YYYY-MM-DD` (ISO 8601 calendar date).
- **Semantics:** When provided, constrains resources to those **on or after** `startDate` based on the resource-specific date fields in the underlying FHIR data (e.g., Encounter period, Observation effective date).

`endDate`

- **Type:** `String` (optional)
- **Format:** Recommended: `YYYY-MM-DD` (ISO 8601 calendar date).
- **Semantics:** When provided, constrains resources to those **on or before** `endDate`.

6. Date and Date-Range Filtering Behavior

General Rules

Scenario	Behavior
No dates provided	No date filtering applied; all available patient data is included.
Only `startDate` provided	Data filtered to resources with dates **on or after** `startDate`.
Only `endDate` provided	Data filtered to resources with dates **on or before** `endDate`.
Both dates provided	Data filtered to resources within the inclusive range [startDate, endDate].

Validation

- If both `startDate` and `endDate` are supplied, the API internally converts them via `new Date(startDate).getTime()` and `new Date(endDate).getTime()`.
- If `startDate > endDate` (after conversion to timestamps), the request is rejected with **HTTP 400** (Bad Request) and message: `"Start date must be before or equal to end date"`.

Implementation Detail

Date parameters are passed to the underlying FHIR repository as `dateParams = { startDate?: string; endDate?: string }` and applied consistently across all G9 resource queries:

- Encounters, Conditions, Allergies, Medications, Observations, Procedures, Immunizations, CareTeams, Devices, Coverages
- MedicationRequests, DocumentReferences, Compositions, Goals, CarePlans, DiagnosticReports, Provenance, HealthConcerns
- Social history, functional status, mental/cognitive Observations
- QuestionnaireResponses, EpisodeOfCares, Appointments

> **Note:** If the repository call for a **non-critical** resource fails, it is treated as an empty list (no data), and document generation continues without interruption.

7. Response Format and Structure

GraphQL Response Shape (success)

```
{
  "data": {
    "generateG9Document": {
      "ccdaUrl": "https://signed-url-to-s3-bucket/transition-of-care/
              uscdi_v3_app_access_<patientId><dateSuffix><
                  hourlySuffixPart>.xml",
      "isCCDAFileExists": false
    }
  }
}
```

Fields

`ccdaUrl`

- **Type:** `String!`
- **Description:** HTTPS URL (typically a time-limited, pre-signed S3 URL) where the generated USCDI v3 Application Access C-CDA XML document can be downloaded.
- **Path format in backing storage:**
  - `${CCDA_EXPORT_S3_KEY}/transition-of-care/uscdi_v3_app_access_${patientId}${dateSuffix}${hourlySuffixPart}.xml`
- `dateSuffix` **derivation:**
  - Both `startDate` and `endDate` : `_<startDate>_<endDate>`
  - Only `endDate` : `_<endDate>`
  - Only `startDate` : `_<startDate>_`
  - No dates: empty string (`''`)

- **`hourlySuffixPart` behavior:**
  - When LaunchDarkly flag `RELEASE_USCDI_G9_V3_CACHE` is enabled: `_<YYYYMMDDHH>` (UTC year-month-day-hour)
  - When the flag is disabled: empty string (`''`)
- Clients must treat this URL as opaque and time-limited; do not assume permanence or reusability outside security policies.

`isCCDAFileExists`

- **Type:** `Boolean!`
- **Description:** Indicates whether the underlying document already existed in storage or was newly generated.
- **Current implementation:** This path currently always returns `isCCDAFileExists: false` for successful responses (no S3 existence check is performed in this mutation path).

8. Error Responses and Status Codes

Although this is GraphQL, the server conveys error conditions via standard HTTP status codes and GraphQL error payloads.

HTTP Status	Condition	Details
200 OK	Success	GraphQL request processed successfully.
400 Bad Request	Validation / config error	Missing or invalid `patientId`; invalid date range; AWS S3/storage configuration errors; pre-signed URL generation failures; file download errors.
401 / 403	Auth failure	Authentication token missing, invalid, expired, or lacking required scopes/roles.
500 Internal Server Error	Unexpected failure	FHIR back-end unavailable; patient not retrievable; document build failures; generic S3 upload failures.

Common 400 Error Messages

```
"Patient ID is required for USCDI v3 Application Access document
generation"
"Start date must be before or equal to end date"
"AWS Access Key ID is invalid or missing. Please check
AWS_ACCESS_KEY_ID configuration."
"AWS Secret Access Key is invalid. Please check AWS_SECRET_ACCESS_KEY
configuration."
"S3 access denied. Please check AWS credentials and bucket
permissions."
```

9. Versioning Information

- **Clinical Data Standard:** USCDI v3, Application Access (G9).
- **Mutation name:** `generateG9Document`.
- **File naming / storage:**
  - `uscdi_v3_app_access_<patientId><dateSuffix><hourlySuffixPart>.xml` under `transition-of-care` prefix.
  - `hourlySuffixPart` is appended only when `RELEASE_USCDI_G9_V3_CACHE` is enabled.

Backward-incompatible changes will be introduced under a new mutation name or versioned endpoint. This documentation corresponds to the deployed behavior of the current `clinical-service` release implementing USCDI v3 Application Access (G9).

10. Sample Requests

10.1. All Data (no date filter)

HTTP

- `POST https://<YOUR_APPROVED_DOMAIN>/graphql`
- Headers:
  - `Content-Type: application/json`
  - `Authorization: Bearer <access_token>`

Body

```
{
  "query": "mutation generateG9Document($patientId: String!,
  $startDate: String, $endDate: String) {
    generateG9Document(patientId: $patientId,
      startDate: $startDate,
        endDate: $endDate) { ccdaUrl isCCDAFileExists } }",
  "variables": {
    "patientId": "12345",
    "startDate": null,
    "endDate": null
  }
}
```

10.2. Date-limited export (inclusive range)

```
{
  "query": "mutation generateG9Document($patientId: String!,
  $startDate: String,
    $endDate: String) {
      generateG9Document(patientId: $patientId,
        startDate: $startDate,
          endDate: $endDate) { ccdaUrl isCCDAFileExists } }",
  "variables": {
    "patientId": "12345",
    "startDate": "2024-01-01",
    "endDate": "2024-03-31"
  }
}
```

11. ONC G9 Documentation Alignment

> **Regulatory Authority:** 45 CFR § 170.315(g)(9) requires Health IT Modules to provide application access to all data for a patient. This is implemented via a GraphQL mutation that generates a USCDI v3-compliant C-CDA XML document containing all available clinical data for the specified patient.

- **Application Access – All Data:** This mutation returns a single downloadable C-CDA file containing all USCDI v3 data available for the patient within the specified date range (or all known history if no date constraints), including G9-specific elements.
- **Behavioral accuracy:** The specification here is derived directly from the deployed resolver and service implementations and reflects actual request/response behavior, validation, and error handling.

> **API Terms and Conditions Notice:** The following sections address API Transparency Conditions required by §170.404(A)(2). These terms and conditions may be updated from time to time to ensure continued compliance with applicable healthcare regulations. Material changes will be communicated to registered developers with appropriate notice. For API access requests or technical support, please contact: api.support@woundtech.net

12. API Terms and Conditions

This section addresses API Transparency Conditions required by §170.404(A)(2).

12.1 API Usage Fees

- **No Usage Fees**: There are no fees charged for accessing or using this USCDI v3 Application Access API
- **Free Access**: All authorized applications can access the API without charge once properly registered and authenticated
- **Infrastructure Costs**: Organizations are responsible for their own infrastructure costs (internet connectivity, computing resources, storage) required to integrate with and use the API

12.2 Usage Restrictions and Limitations

- **Authorized Use Only**: API access is restricted to applications that have completed the registration process and maintain valid authentication credentials
- **Patient Data Scope**: Applications may only access data for patients who have provided appropriate consent or for whom the application is authorized to act
- **Request Limits**:
  - Maximum request body size: 10MB
  - Connection timeout: 90 seconds
  - Document generation timeout: Recommended minimum 30 seconds client-side timeout
- **Data Usage Restrictions**:
  - Downloaded C-CDA documents contain Protected Health Information (PHI) and must be handled according to HIPAA requirements
  - Pre-signed download URLs are time-limited and must not be stored, logged, or shared
  - API responses and PHI data must not be cached beyond what is necessary for application functionality
- **Prohibited Activities**:
  - Attempting to access data for unauthorized patients
  - Sharing, selling, or redistributing patient data without proper authorization
  - Using the API for purposes other than authorized healthcare operations
  - Attempting to circumvent authentication or access controls

12.3 Developer Obligations

- **Compliance Requirements**:
  - Maintain compliance with HIPAA Privacy and Security Rules
  - Implement appropriate technical and administrative safeguards for PHI
  - Report any suspected data breaches according to organizational policies
  - Follow applicable state and federal healthcare regulations
- **Technical Requirements**:
  - Implement secure token storage and management
  - Use only approved authentication methods (OAuth 2.0)
  - Maintain current API client implementations compatible with documented specifications
  - Implement proper error handling and user notifications
- **Operational Obligations**:
  - Keep registration information current and accurate
  - Notify the organization of any changes to application functionality or data usage
  - Maintain audit logs of API access and patient data handling
  - Respond promptly to security or compliance inquiries

13. Developer Registration Process

13.1 Registration Requirements

- **Application Information**:
  - Application name, version, and description
  - Developer/organization contact information
  - Intended use case and patient population
  - Technical architecture and integration approach
- **Security Documentation**:
  - Data security and privacy policies
  - HIPAA compliance documentation
  - Technical security controls implementation
  - Incident response procedures
- **Legal Requirements**:
  - Business Associate Agreement (BAA) execution
  - Terms of Use acceptance
  - Liability and insurance documentation
  - Regulatory compliance attestations

13.2 Registration Process Steps

1. **Initial Application Submission**:
    - Complete developer registration form
    - Provide required documentation and attestations
    - Submit for initial review
2. **Security and Compliance Review**:
    - Technical architecture assessment
    - Security controls validation
    - HIPAA compliance verification
    - Legal documentation review
3. **Testing and Validation**:
    - Sandbox environment access provided
    - Integration testing with test data
    - API functionality validation
    - Security testing and penetration testing (if required)
4. **Production Approval**:
    - Final compliance review
    - Production credentials issuance
    - Go-live authorization

13.3 Approval Criteria and Timeline

- **Approval Criteria**:
  - Complete and accurate registration information
  - Demonstrated HIPAA compliance capabilities
  - Appropriate technical security controls
  - Valid business justification for API access
  - Successfully completed testing and validation
- **Review Timeline**:
  - Initial review: 5-10 business days
  - Security assessment: 10-15 business days
  - Testing phase: 15-30 business days (developer-dependent)
  - Final approval: 3-5 business days
  - Total process: Typically 30-60 business days

14. Business Policies and Support

14.1 Support and Contact Information

- **Technical Support**:
  - Business hours: Monday-Friday 9:00 a.m. – 5:00 p.m. (EST Time)
  - Response time: Within 4 business hours for critical issues
  - Support channels: Email

14.2 Liability and Warranties

By accessing and using the application, the user agrees to the following service disclaimers, limitation of liability, and developer responsibilities.

- **Service Disclaimers**:
  - API provided "as is" without warranties, express, implied, or statutory, including implied warranties of merchantability, non-infringement, and fitness for a particular purpose.
  - No guarantee of uninterrupted service or error-free operation.
  - Clinical decisions should not be based solely on API data without proper validation
- **Limitation of Liability**:
  - No liability for consequential or indirect damages, exemplary, indirect, incidental, or special damages arising out of the access and use of the application or any API, or of the delivery, installation, use, maintenance, or operation of the application, API, any other software, or professional services provided hereunder even if Woundtech has been advised of the possibility of such damages.
  - In no event shall Woundtech have any liability for any medical treatment decisions made based upon use of the API or any other software, including, without limitation, any claims for negligence, medical, dental or podiatric malpractice, wrongful death, bodily harm or any other personal injury or property damage.
  - Maximum liability capped at amount specified in Business Associate Agreement.

- **Developer Responsibilities**:
  - Developers assume responsibility for proper data handling.
  - Indemnification of Woundtech by user for unauthorized use or data breaches as specified within the Business Associate Agreement, and User agrees to defend, indemnify and hold Woundtech harmless including its officers, directors, agents, employees and their respective representatives, successors, and permitted assigns from and against any and all third-party investigations, claims, demands, suits, actions, and proceedings, and any debts, liabilities, obligations, expenses, losses, damages, assessments, orders and judgments (including without limitation reasonable attorneys' fees and all legal costs and disbursements) of any kind or nature whatsoever resulting from any such claims and/or liabilities, known or unknown, fixed, contingent, accrued, absolute, threatened or otherwise, to the extent sustained, suffered or incurred by or made against Woundtech after the access and use of the application and/or API.
  - Compliance with all applicable healthcare regulations, including but not limited to Health Insurance Portability and Accountability Act of 1996 (“HIPAA”), U.S. Export Control Laws, and all applicable laws and regulations.
  - User acknowledges that any use of artificial intelligence, machine learning, and automated decision-making technologies (collectively, "AI Technologies") will be in compliance with HIPAA, applicable laws and regulations, and industry standards.