§ 170.315(g)(7) Application Access – Patient Selection

Secure API-Based Patient Selection and Context Sharing

Overview

This documentation demonstrates compliance with **45 CFR § 170.315(g)(7) – Application Access: Patient Selection**. Our Health IT Module enables authorized third-party applications to receive a patient selection context from the EHR — without the application having to independently manage patient look-up — using secure APIs.

Requirement Details

> **Regulatory Reference:** 45 CFR § 170.315(g)(7) requires the Health IT Module to respond to requests from applications for patient selection. Specifically, the EHR must be capable of passing a patient identifier to an application so the application can retrieve relevant data without requiring a separate patient search.

Authentication and Authorization Overview

Authentication

- **Mechanism:** Standard API authentication using the endpoint with username and password provided during 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"
  }
}
```

Obtaining an Access Token

- **Token type:** Short-lived OAuth2 bearer access token issued along with the idToken
- **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.

Configuration Requirements

Applications require basic configuration for:

- **API Endpoints:** HTTP endpoint URL for authentication and authorization token.
- **Client Credentials:** Username and password (provided during registration).
- **Security Settings:** TLS/SSL validation and timeout configurations.

Woundtech

API Reference Documentation

This section fulfills the documentation requirement under **§ 170.315(g)(7)(ii)(A)(1)**: API syntax, function names, required and optional parameters and their data types, return variables and their types/structures, and exceptions and exception handling methods.

1. Authentication — `POST /getG9Token`

Authenticates a registered application and returns bearer tokens for subsequent API calls.

Request

Parameter	Location	Type	Required	Description
`username`	Body (JSON)	string	Required	Registered application username (email format).
`password`	Body (JSON)	string	Required	Application password provided during registration.

Response — `200 OK`

Field	Type	Description
`AuthenticationResult.AccessToken`	string	Short-lived OAuth 2.0 bearer token. Include as `Authorization: Bearer <AccessToken>` in subsequent requests.
`AuthenticationResult.ExpiresIn`	number	Token validity in seconds (typically `3600`). Applications must re-authenticate upon expiry.
`AuthenticationResult.IdToken`	string	JWT identity token. Contains claims about the authenticated application identity.
`AuthenticationResult.TokenType`	string	Always `"Bearer"`.

Exception Handling

> **Common Error Responses:**

HTTP Status	Error Code	Description	Recommended Handling
`400`	`InvalidParameterException`	Missing or malformed `username` / `password` fields.	Validate request body before sending. Surface configuration error to operator.
`401`	`NotAuthorizedException`	Incorrect username or password.	Do not retry automatically. Prompt re-entry of credentials or contact support.
`404`	`UserNotFoundException`	Username does not exist in the system.	Verify registration status. Contact api.support@woundtech.net.
`429`	`TooManyRequestsException`	Authentication rate limit exceeded.	Implement exponential back-off. Minimum retry delay: 5 seconds.
`500`	`InternalErrorException`	Unexpected server-side error.	Retry up to 3 times with back-off. If persistent, contact support with request ID.

2. Patient Selection Context — `POST /g7/patientSelection`

Passes a selected patient's identifier and context from the EHR to the third-party application. The application uses this context to retrieve patient-specific data without performing an independent patient search.

Request Headers

Header	Type	Required	Description
`Authorization`	string	Required	Bearer token obtained from `/getG9Token`. Format: `Bearer <AccessToken>`.
`Content-Type`	string	Required	Must be `application/json`.

Request Body Parameters

Parameter	Type	Required	Description
`patientId`	string	Required	Unique EHR patient identifier (internal system ID).
`encounterId`	string	Optional	Active encounter ID to scope context to a specific visit.
`applicationId`	string	Required	Registered application identifier issued at onboarding.
`contextScope`	array<string>	Optional	Array of data domains to include in context. Accepted values: `"demographics"`, `"conditions"`, `"medications"`, `"allergies"`, `"vitals"`. Defaults to `["demographics"]` if omitted.
`callbackUrl`	string	Optional	HTTPS URL to receive an asynchronous context payload. Must be pre-registered.

Example Request

```
curl --location "https://emr.woundtech.net/g7/patientSelection" \
    --header "Authorization: Bearer ey.....AA" \
    --header "Content-Type: application/json" \
    --data-raw "{
        \"patientId\": \"PT-00123456\",
        \"encounterId\": \"ENC-20240901-001\",
        \"applicationId\": \"app-wt-external-001\",
        \"contextScope\": [\"demographics\", \"conditions\",
\"medications\"]
    }"
```

Response — `200 OK`

Field	Type	Description
`status`	string	`"success"` on successful context delivery.
`patientContext`	object	Top-level container for patient context data (see sub-fields below).
`patientContext.patientId`	string	EHR patient identifier echoed from the request.
`patientContext.encounterId`	string	Active encounter identifier (present only when supplied in request).
`patientContext.demographics`	object	Patient demographic data: `firstName`, `lastName`, `dateOfBirth` (ISO 8601), `gender`, `mrn`.

`patientContext.conditions`	array<object>	Active problem list entries. Each entry: `code` (ICD-10), `display`, `onsetDate`.
`patientContext.medications`	array<object>	Active medication list entries. Each entry: `rxNorm`, `name`, `dose`, `frequency`, `status`.
`patientContext.allergies`	array<object>	Allergy and intolerance records. Each entry: `substance`, `reaction`, `severity`.
`patientContext.vitals`	array<object>	Most recent vital signs. Each entry: `type`, `value`, `unit`, `recordedAt` (ISO 8601).
`contextToken`	string	Short-lived opaque token (TTL: 15 minutes) that an application may present to retrieve or refresh the same patient context without repeating the full request.
`expiresAt`	string	ISO 8601 timestamp indicating when `contextToken` expires.

Example Response


```
{
  "status": "success",
  "patientContext": {
    "patientId": "PT-00123456",
    "encounterId": "ENC-20240901-001",
    "demographics": {
      "firstName": "Jane",
      "lastName": "Doe",
      "dateOfBirth": "1975-04-12",
      "gender": "female",
      "mrn": "MRN-987654"
    },
    "conditions": [
      { "code": "L97.319", "display": "Non-pressure chronic ulcer of right
ankle", "onsetDate": "2024-01-15" }
    ],
    "medications": [
      { "rxNorm": "197381", "name": "Metformin 500 mg", "dose": "500 mg",
"frequency": "BID", "status": "active" }
    ]
  },
  "contextToken": "ctx_abc123xyz",
  "expiresAt": "2024-09-01T14:30:00Z"
}
```


Exception Handling

HTTP Status	Error Code	Description	Recommended Handling
`400`	`InvalidRequest`	Missing required fields (`patientId` , `applicationId`) or invalid `contextScope` values.	Inspect `errorDetail` field in response body. Correct the request before retrying.

`401`	`Unauthorized`	Missing, expired, or invalid bearer token.	Re-authenticate via `/getG9Token` and retry with fresh token.
`403`	`Forbidden`	Application is not authorized to access the specified patient.	Verify patient consent/authorization. Do not retry without resolving authorization.
`404`	`PatientNotFound`	`patientId` does not match any record in the system.	Verify the patient identifier. Notify the user that patient context could not be loaded.
`408`	`RequestTimeout`	Request exceeded the 90-second connection timeout.	Implement client-side timeout of at least 30 seconds. Retry once; if failure persists, surface a timeout error to the user.
`413`	`PayloadTooLarge`	Request body exceeds 10 MB limit.	Reduce `contextScope` to necessary domains only.

`429`	`TooManyRequests`	Rate limit exceeded.	Implement exponential back-off starting at 5 seconds. Honor `Retry-After` response header when present.
`500`	`InternalServerError`	Unexpected server-side error.	Log the `requestId` from the response. Retry up to 3 times with back-off. Report persistent failures to api.support@woundtech.net.
`503`	`ServiceUnavailable`	API temporarily unavailable (maintenance or overload).	Retry after the interval indicated in `Retry-After` header. Display a service-unavailable message to the end user.

Error Response Structure

All error responses share a consistent JSON structure:

```
{
  "status": "error",
  "errorCode": "PatientNotFound",
  "errorDetail": "No patient record found for patientId: PT-99999999",
  "requestId": "req_7f3ab12c-4e89-4d0a-a1bc-123456789abc",
  "timestamp": "2024-09-01T13:45:00Z"
}
```

Field	Type	Description
`status`	string	Always `"error"` for non-2xx responses.
`errorCode`	string	Machine-readable error code (see tables above).
`errorDetail`	string	Human-readable description of the error condition.
`requestId`	string	Unique identifier for the failed request. Include when contacting support.
`timestamp`	string	ISO 8601 UTC timestamp of when the error occurred.

Software Components and Configuration

This section fulfills **§ 170.315(g)(7)(ii)(A)(2)**: the software components and configurations necessary for an application to successfully interact with the API and process its responses.

Required Software Components

Component	Purpose	Notes
HTTPS Client	Send authenticated API requests over TLS 1.2+.	TLS 1.0/1.1 are not supported. Certificate chain validation must be enabled.
JSON Parser	Deserialize all request and response payloads.	All payloads use `Content-Type: application/json` with UTF-8 encoding.
OAuth 2.0 Token Manager	Obtain, store, and refresh bearer tokens.	Must detect `ExpiresIn` and re-authenticate proactively (recommended: refresh 60 s before expiry).
Secure Credential Store	Store `username`, `password`, and tokens at rest.	Credentials must never be stored in source code, logs, or URL parameters. Use OS keychain, secrets manager, or encrypted vault.
Error Handler / Retry Logic	Handle non-2xx responses gracefully.	Implement exponential back-off for `429` and `5xx` responses. Maximum 3 automatic retries.
Audit Logger	Record API access events for HIPAA compliance.	Log `requestId`, timestamp, endpoint, and response status. Do not log PHI or token values.

Network and Transport Configuration

Setting	Required Value
Protocol	HTTPS (TLS 1.2 minimum; TLS 1.3 recommended)
Base URL	`https://emr.woundtech.net`
Connection Timeout	90 seconds (server-side limit; set client timeout ≥ 30 s)
Maximum Request Body	10 MB
Certificate Validation	Required — do not disable SSL verification in production

Processing API Responses

1. **Check HTTP status code** before parsing the body. Any non-`2xx` status indicates an error; parse `errorCode` and `errorDetail` accordingly.
2. **Extract** `contextToken` and `expiresAt` from successful patient-selection responses. Cache the token securely for the duration of the session (max 15 minutes).
3. **Handle missing optional fields** defensively — fields such as `encounterId`, `conditions`, or `vitals` may be absent if not requested or not available for the patient.
4. **Validate ISO 8601 date fields** (`dateOfBirth`, `onsetDate`, `recordedAt`) using a standards-compliant date parser before display or calculation.
5. **PHI handling:** Patient context payloads contain Protected Health Information. Store and transmit in compliance with HIPAA. Do not cache beyond session need; do not log PHI fields.

Integration Checklist

- ✓ TLS 1.2+ enforced on all outbound connections to `emr.woundtech.net`
- ✓ Bearer token obtained via `/getG9Token` and attached to every API request header
- ✓ Token expiry monitored; re-authentication triggered before expiry
- ✓ Credentials stored in a secure vault — not in source code or environment variables in plain text
- ✓ Request body validated against required parameters before sending
- ✓ Non-2xx responses caught and handled per error-handling table
- ✓ Exponential back-off implemented for `429` / `5xx` responses
- ✓ Audit log records API calls without capturing PHI or token values
- ✓ PHI from patient context handled in accordance with HIPAA Security Rule
- ✓ Pre-signed or time-limited URLs (e.g., `contextToken`) not stored, logged, or shared

> **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

API Terms and Conditions

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

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.

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: 10 MB
  - 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.

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.

Developer Registration Process

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

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

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.