API – TERMS

FOR

PATIENT SELECTION / DATA CATEGORY REQUEST / ALL DATA REQUEST

170.315(G)(7), 170.315(G)(8), 170.315(G)(9)

Table of Contents

OVERVIEW

This API document provides the information on how the clients can retrieve details for specific patients once they discontinue the service from InSync.

When the clients use EHR application other than InSync, they may want to view patient details that are recorded in the InSync application for further treatment. In this case, InSync provides the API documentation along with URL and credentials. The users can access the URL and enter the InSync ID, Username, and Password to retrieve patient details.

When extracting patient data, the system will search for parameters such as First Name, Last Name, DOB, and Sex to identify unique patient details.

The users can,

- ✓ Extract Patient data for specific date or for a specific date range.
- ✓ Extract patient's basic and clinical details such as Race, Ethnicity, Preferred Language, Vitals, Medications, Allergies, Lab Tests, Immunizations, and so forth.
- ✓ Download C-CDA file while extracting the data which can be used to display the patient details in another application as per their needs.

API DETAILS

This guide provides information about the following three Web APIs (Application Program Interfaces) required for QPP (Quality Payment Program) Measure “Provide Patient Access”.

- ✓ Patient Selection
- ✓ Data Category Request
- ✓ All Data Request

A third-party application can use these three APIs to retrieve a unique patient identifier based on patient identification. Patient identification can be First Name, Last Name, Gender, and Date of Birth. This identifier can be further used to get CCDS (Common Clinical Data Set) for a specific patient. You can filter the CCDS content by Date and/or Data Category.

Quality Payment Program (QPP) APIs are web APIs which are used to retrieve the patient information.

Regulation Text Citation	Certification Criterion
§ 170.315(g)(7)	Application access – patient selection
§ 170.315(g)(8)	Application access – data category request
§ 170.315(g)(9)	Application access – all data request

KEY INFORMATION TO SHARE

Before starting the any development on using InSync API, InSync will share credentials which are used to communicate with InSync.

InSync will share two different credentials are as

1. Staging (Used to start development on implementing the API)
2. Production (Once implementation completed InSync will Share another credential for production use)

Credentials are as below.

1. ClientId
2. Username
3. Password

These credentials are used when user start connecting to InSync for to fetch patient information.

API SYNTAX AND FUNCTION NAMES

This section includes the API syntax, function names, required and optional parameters and their data types, return variables and their types / structures, exceptions and exception handling methods and their returns.

USER LOGIN VALIDATION

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/IsUserValidate

For example, if your site name is xyz.insynchcs.com, then in YourSiteName.com will be replaced by 1.insynchcs.com.

Function Name:

IsUserValidate

Required parameters and their data types:

Parameter Name	Parameter Value	Data Type
ClientId	Parameter	String
UserName	Parameter	String
Password	Parameter	String

Return variables and their types/structures:

Return Variable Name	Return Variable Value	Data Type
practiceId	Parameter	Integer
UserId	Parameter	String
ClientId	Parameter	Integer

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

PATIENT SEARCH

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/SearchPatient

Function Name:

SearchPatient

Required parameters and their data types:

Parameter Name	Parameter Value	Data Type
PrefixText	Parameter	String
PracticeId	Parameter	Integer

Return variables and their types/structures:

Return Variable Name	Return Variable Value	Data Type
practiceId	Parameter	Integer
patientId	Parameter	Integer
patientName	Parameter	String
dob	Parameter	String
dobStr	Parameter	String
sex	Parameter	String

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

GENERATE TOKEN

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/GenerateToken

Function Name:

GenerateToken

Required parameters and their data types:

Parameter Name	Parameter Value	Data Type
PracticeId	Parameter	Integer
UserId	Parameter	Integer
PatientId	Parameter	Integer
DOB	Parameter	String
PatientName	Parameter	String

Return variables and their types/structures:

Return Variable Name	Return Variable Value	Data Type
GenerateToken	Parameter	String

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

ENCOUNTER DETAILS

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/GetEncounterDetailsByPatient

Function Name:

GetEncounterDetailsByPatient

Required parameters and their data types:

Parameter Name	Parameter Value	Data Type
PracticeId	Parameter	Integer
UserId	Parameter	Integer
token	Parameter	String
PatientId	Parameter	Integer

Return variables and their types/structures:

Return Variable Name	Return Variable Value	Data Type
EncounterId	Parameter	Integer
EncounterName	Parameter	String
EncounterVisitType	Parameter	String

API Header Authentication:

Return Variable Name	Return Variable Value	Data Type
Content-Type	application/json	String
token	Parameter	String
pid	Parameter	Integer
uid	Parameter	Integer

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

VIEW FILE XML/JSON

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/ViewDownloadFile

Function Name:

ViewDownloadFile

Required and Optional parameters and their data types:

Parameter Name	Parameter Value	Data Type	Mandatory / Optional	
PracticeId	Parameter	Integer	Mandatory	
UserId	Parameter	Integer	Mandatory	
Token	Parameter	String	Mandatory	
isXML	Parameter	Boolean	Mandatory	
PatientId	Parameter	Integer	Mandatory	
SelectedElem	Parameter	Selection List For example, [{"Key":"1","Selected":true}, {"Key":"2","Selected":false}]	Mandatory	Refer to the Section Elements below the table for key values
fromDate	Parameter	String	Optional	
toDate	Parameter	String	Optional	
EncounterId	Parameter	String	Optional	

Selection List Elements (Key Value)

If you want to view patient name, then, key value for patient name i.e. 1 should be true, for example, {"Key":"1","Selected":true}

Similarly, following is a list of remaining key values:

Clinical Elements	Key	Clinical Elements	Key
Patient Name	1	Laboratory Tests	11
Sex	2	Laboratory Values(s)/Result(s)	12
Date of Birth	3	Vital Signs	13
Race	4	Procedures	14

Ethnicity	5	Care Team Member(s)	15
Preferred Language	6	Immunizations	16
Smoking Status	7	Unique Device Identifier(s) for a Patient's Implantable Device(s)	17
Problems	8	Assessment and Plan of Treatment	18
Medications	9	Goals	19
Medication Allergies	10	Health Concerns	20

API Header Authentication:

Return Variable Name	Return Variable Value	Data Type
Content-Type	application/json	String
token	Parameter	String
pid	Parameter	Integer
uid	Parameter	Integer

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

Return variables and their types/structures:

Xml File / Json File

C-CDA

API Syntax:

https://{YourSiteName.com}/api/PatientSelectionMU/GetXML_ForCDA

Function Name:

GetXML_ForCDA

Required and Optional parameters and their data types:

Parameter Name	Parameter Value	Data Type	Mandatory / Optional	
PracticeId	Parameter	Integer	Mandatory	
UserId	Parameter	Integer	Mandatory	
Token	Parameter	String	Mandatory	
TypeId	Parameter	Integer	Mandatory	
PatientId	Parameter	Integer	Mandatory	
Sections	Parameter	String For example, "1001,1301,1401,1501,1701,1801,1901,2001,2601,2701,2801,6000,7000"	Mandatory	Refer to the Section Elements below the table for key values
fromDate	Parameter	String	Optional	
toDate	Parameter	String	Optional	
EncounterId	Parameter	String	Optional	

Section Elements (Value)

If you want to view Allergies, then, key value for Allergies is 1001.

Similarly, following is a list of remaining key values:

SectionId	SectionName	SectionId	SectionName
1001	Allergies	3101	General Status
1101	Chief Complaint + Reason for Visit	3201	History of Past Illness
1201	Family History	3301	Review of Systems
1301	Immunization	3401	DICOM Object Catalog

1401	Medication	3501	Findings (Radiology Study Observation)
1501	Problem List	3601	Hospital Course
1601	Reason for Referral	3701	Hospital Discharge Diagnosis
1701	Vital Sign	3801	Hospital Discharge Medications
1801	Social History	3901	Anesthesia
1901	Results	4001	Complications
2001	Procedures	4101	Postoperative Diagnosis
2101	Plan of Care	4201	Preoperative Diagnosis
2201	Instruction	4301	Procedure Description
2301	Functional and Cognitive Status	4401	Procedure Disposition
2401	Advanced Directives	4501	Procedure Estimated Blood Loss
2501	Payers	4601	Procedure Findings
2601	Medical Equipment	4701	Procedure Indications
2701	Encounters	4801	Procedure Specimens Taken
2801	Assessment	4901	Postprocedure Diagnosis
2901	History of Present Illness	5001	Medications Administered
3001	Physical Exam	5101	SOCIALHISTORY_New

API Header Authentication:

Return Variable Name	Return Variable Value	Data Type
Content-Type	application/json	String
token	Parameter	String
pid	Parameter	Integer
uid	Parameter	Integer

Exceptions and exception handling methods and their returns:

Exception Type	Exception Value	Data Type	Output
Null Exception Handling	-	String	Json format
Error Exception Handling	-	String	Json format

Return variables and their types/structures:

C-CDA Xml File

SOFTWARE COMPONENTS AND CONFIGURATIONS

This section includes the software components and configurations that would be necessary for an application to implement to be able to successfully interact with the API and process its response(s).

USER LOGIN VALIDATION (EXAMPLE OF C# CODE)

```
[HttpPost]
 public ActionResult Index(ApiUser objApiUser)
    {
            var hclient = new HttpClient();
            var res = hclient.PostAsync(ApiUrl + "api/PatientSelectionMU/IsUserValidate",
objApiUser, new JsonMediaTypeFormatter()).Result;
            string resRespo = null;
            if (res.IsSuccessStatusCode)
            {
                resRespo = res.Content.ReadAsStringAsync().Result;
            }

            if (resRespo != "null")
            {
                ClientUserDetail ClientDetail;
                ClientDetail = JsonConvert.DeserializeObject<ClientUserDetail>(resRespo);
                Session["UserId"] = ClientDetail.UserId;
                Session["PracticeId"] = ClientDetail.PracticeId;

                Session["DBServerName"] = ClientDetail.SourceServerIP;
                Session["DBName"] = ClientDetail.DatabaseName;
                Session["DBLogin"] = ClientDetail.DBUserName;
                Session["DBPassword"] = ClientDetail.DBPassword;

                FormsAuthentication.SetAuthCookie(objApiUser.UserName, true);
                return RedirectToAction("ExtractPatientData");

            }
            Session["PracticeId"] = null;
            ModelState.AddModelError("invalidlogin", "Invalid credentials");
            return View(objApiUser);
        }       Session["PracticeId"] = null;
            ModelState.AddModelError("invalidlogin", "Invalid credentials");
            return View(objApiUser);
     }
```

PATIENT SEARCH (EXAMPLE OF C# CODE)

```
  [Authorize]
        public ActionResult PatientSearch(string prefixText)
        {
            if (checkSession())
                return redirectToLogin();

            WebClient client = new WebClient();
            client.BaseAddress = ApiUrl;
            client.Headers[HttpRequestHeader.ContentType] = "application/json";

            var inputParam = new { PracticeId = Convert.ToInt32(Session["PracticeId"]),
prefixText = prefixText };
            string jsonData = JsonConvert.SerializeObject(inputParam);

            var response = client.UploadString("api/PatientSelectionMU/SearchPatient",
jsonData);
            var hclient = new HttpClient();
            var res = hclient.PostAsync(ApiUrl + "api/PatientSelectionMU/SearchPatient",
inputParam, new JsonMediaTypeFormatter()).Result;
            string resRespo = null;
            if (res.IsSuccessStatusCode)
            {
                resRespo = res.Content.ReadAsAsync<object>().Result.ToString();
            }
            return Content(resRespo);
        }
```

GENERATE TOKEN (EXAMPLE OF C# CODE)

```
        [Authorize]
         [HttpPost]
         public ActionResult GenerateToken(int PatientId, string DOBStr, string PatientName =
"")
         {
             if (checkSession())
                 return redirectToLogin();

             WebClient client = new WebClient();
             client.BaseAddress = ApiUrl;
             client.Headers[HttpRequestHeader.ContentType] = "application/json";
             //client.Headers[HttpRequestHeader.Authorization] = "insync:pwd";
             var inputParam = new
             {
                 PracticeId = Convert.ToInt32(Session["PracticeId"]),
                 UserId = Convert.ToInt32(Session["UserId"]),
                 PatientId = PatientId,
                 DOB = DOBStr,
                 PatientName = PatientName
             };
             string jsonData = JsonConvert.SerializeObject(inputParam);

             var response = client.UploadString("api/PatientSelectionMU/GenerateToken",
jsonData);

             return Content(JsonConvert.DeserializeObject(response).ToString());
         }
```

ENCOUNTER DETAILS (EXAMPLE OF C# CODE)

```
[Authorize]
        [HttpPost]
        public ActionResult EncounterDetailsByPatient(string token, int PatinetId)
        {
            WebClient client = new WebClient();
            client.BaseAddress = ApiUrl;
            client.Headers[HttpRequestHeader.ContentType] = "application/json";
            if (token != "")
            {
                client.Headers.Add("token", token);
                client.Headers.Add("pid", Convert.ToString(Session["PracticeId"]));
                client.Headers.Add("uid", Convert.ToString(Session["UserId"]));
            }
            var inputParam = new
            {
                PracticeId = Convert.ToInt32(Session["PracticeId"]),
                UserId = Convert.ToInt32(Session["UserId"]),
                token = token,
                PatientId = PatinetId,
            };

            string jsonData = JsonConvert.SerializeObject(inputParam);
            string response = "";
            try
            {


                response =
client.UploadString("api/PatientSelectionMU/GetEncounterDetailsByPatient", jsonData);
                if (response != null && response != "null")
                {
                    return Json(response, JsonRequestBehavior.AllowGet);
                }
            }
            catch (WebException exception)
            {
                string responseText;
                ResponseModel responseModal;
                using (var reader = new StreamReader(exception.Response.GetResponseStream()))
                {
                    responseText = reader.ReadToEnd();
                    responseModal = JsonConvert.DeserializeObject<ResponseModel>(responseText);
                }
                return Json(responseModal, JsonRequestBehavior.AllowGet);
            }
            return Json("");
        }
```

VIEW FILE XML/JSON (EXAMPLE OF C# CODE)

```
[HttpPost]
        public JsonResult ViewDownloadFile(string token, int PatientId, bool isDownLoad, string
selectedEl, bool isXML, DateTime? fromDate, DateTime? toDate,string EncounterId)
        {
            if (checkSession())
                return redirectToLogin();

            toDate = toDate ?? fromDate;
            var sElements = JsonConvert.DeserializeObject<List<SelectedElement>>(selectedEl);
            WebClient client = new WebClient();
            client.BaseAddress = ApiUrl;
            client.Headers[HttpRequestHeader.ContentType] = "application/json";
            if (token != "")
            {
                client.Headers.Add("token", token);
                client.Headers.Add("pid", Convert.ToString(Session["PracticeId"]));
                client.Headers.Add("uid", Convert.ToString(Session["UserId"]));
            }

            var inputParam = new
            {
                PracticeId = Convert.ToInt32(Session["PracticeId"]),
                UserId = Convert.ToInt32(Session["UserId"]),
                token = token,
                isXML = isXML,
                PatientId = PatientId,
                SelectedElem = sElements,
                fromDate = fromDate,
                toDate = toDate,
                EncounterId = EncounterId
            };
            string jsonData = JsonConvert.SerializeObject(inputParam);

            string response = "";
            try
            {
                response = client.UploadString("api/PatientSelectionMU/ViewDownloadFile",
jsonData);

                if (response != null && response != "null")
                {
                    var Output = JsonConvert.DeserializeObject<FileObject>(response);
                    return Json(Output, JsonRequestBehavior.AllowGet);
                }
            }
            catch (WebException exception)
            {
                string responseText;
```

```
            ResponseModel responseModal;
            using (var reader = new StreamReader(exception.Response.GetResponseStream()))
            {
                responseText = reader.ReadToEnd();
                responseModal = JsonConvert.DeserializeObject<ResponseModel>(responseText);
            }
            return Json(responseModal, JsonRequestBehavior.AllowGet);
        }
        return Json("");
    }
```

C-CDA (EXAMPLE OF C# CODE)

```
[HttpPost]
        public JsonResult GetCDA(string token, int PatientId, bool isDownLoad, string
selectedEl, bool isXML, DateTime? fromDate, DateTime? toDate,string EncounterId)
        {
            if (checkSession())
                return redirectToLogin();

            toDate = toDate ?? fromDate;
            var sElements = JsonConvert.DeserializeObject<List<SelectedElement>>(selectedEl);
            WebClient client = new WebClient();
            client.BaseAddress = ApiUrl;
            client.Headers[HttpRequestHeader.ContentType] = "application/json";
            if (token != "")
            {
                client.Headers.Add("token", token);
                client.Headers.Add("pid", Convert.ToString(Session["PracticeId"]));
                client.Headers.Add("uid", Convert.ToString(Session["UserId"]));

            }

            var inputParam = new
            {
                PracticeId = Convert.ToInt32(Session["PracticeId"]),
                UserId = Convert.ToInt32(Session["UserId"]),
                token = token,
                TypeId = 1,
                PatientId = PatientId,
                Sections = "1001,1301,1401,1501,1701,1801,1901,2001,2601,2701,2801,6000,7000",
                fromDate = fromDate,
                toDate = toDate,
                EncounterId = EncounterId
            };


            string jsonData = JsonConvert.SerializeObject(inputParam);

            string response = "";
            try
            {
                response = client.UploadString("api/PatientSelectionMU/GetXML_ForCDA",
jsonData);

                if (response != null && response != "null")
                {
                    var Output = JsonConvert.DeserializeObject<FileObject>(response);
                    return Json(Output, JsonRequestBehavior.AllowGet);
                }
            }
```

```
        catch (WebException exception)
        {
            string responseText;
            ResponseModel responseModal;
            using (var reader = new StreamReader(exception.Response.GetResponseStream()))
            {
                responseText = reader.ReadToEnd();
                responseModal = JsonConvert.DeserializeObject<ResponseModel>(responseText);
            }
            return Json(responseModal, JsonRequestBehavior.AllowGet);
        }
        return Json("");
    }
```

TERMS OF USE

These are our terms and conditions for use of the network, which you may access in several ways. In these terms and conditions, when we say the "InSync Site" we mean the healthcare information network operated by InSync Healthcare Solutions LLC regardless of how you access the network.

REGISTRATION

You may access areas of the InSync Site that require registration by becoming a registered member and creating an account with us. You agree to be responsible for maintaining the confidentiality of your passwords or other account identifiers which you choose and all activities that occur under your account.

By registering on the InSync Site, you agree that:
(i) your account and password are personal to you and may not be used by anyone else to access the InSync Site;

(ii) you will not do anything which would assist anyone who is not a registered user to gain access to any registration area of the InSync Site; and

(iii) you will not create registration accounts for the purpose of abusing the functionality of the site, or other users; nor will you seek to pass yourself off as another user.

You agree to notify us immediately if you become aware any unauthorized use of your password or account identifiers by others.

TERMINATION OF REGISTRATION

If you no longer wish to have a registered account, you may terminate your account by sending an email to support@insynchcs.com. If you no longer accept these terms and conditions, or any future modification to these terms and conditions, you must cease using the InSync Site. Continued use of the InSync Site indicates your continued acceptance of these terms and conditions.

If, for any reason, we believe that you have not complied with these terms and conditions, we may, at our sole discretion, cancel your access to the registration areas of InSync Site immediately and without prior notice. We may terminate your registered account, at our sole discretion, by emailing you at the address you have registered stating that the agreement has terminated.