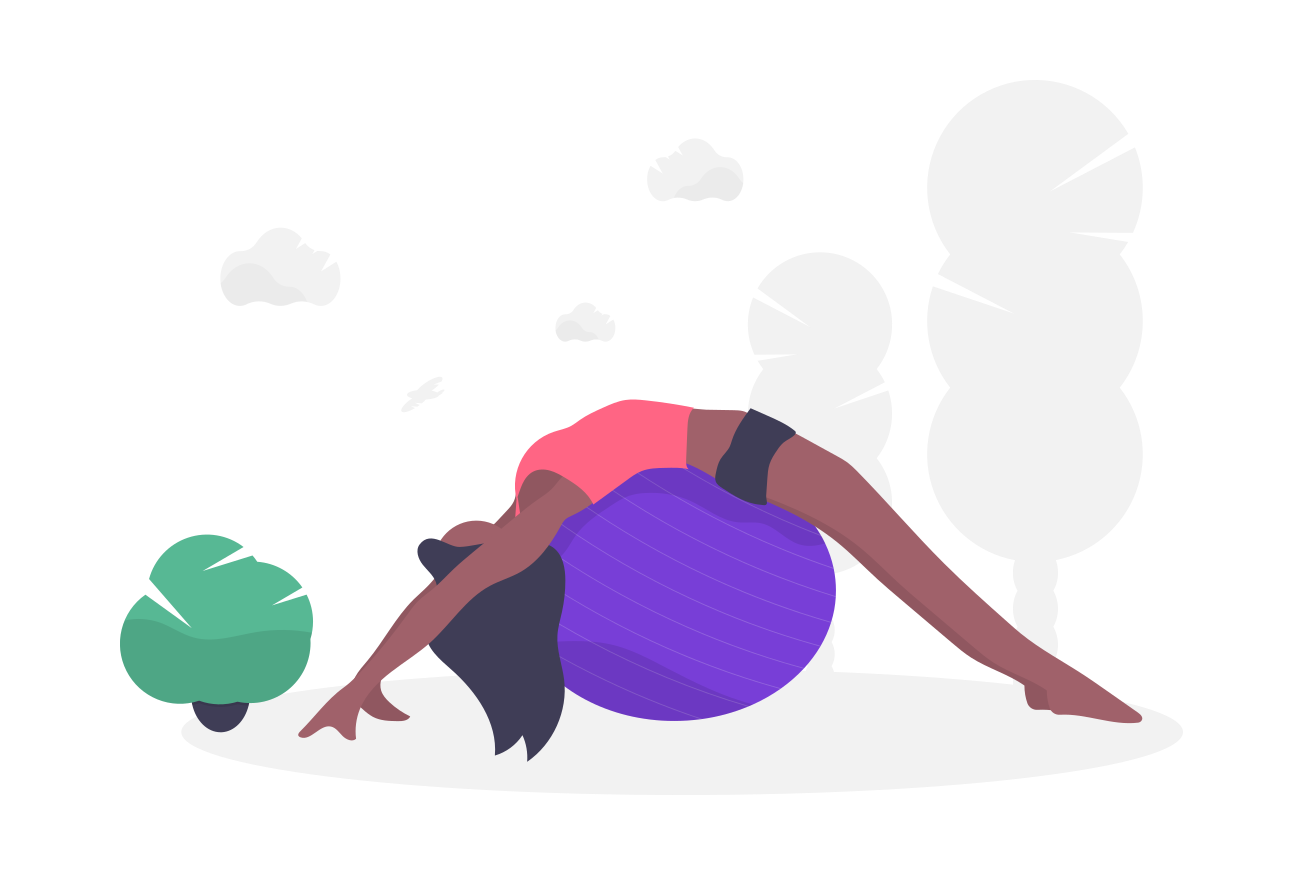
Build intelligence into your apps using Spry RTM SDKs



JS SDK Overview

## Introduction

Spry RTM developer platform is a core part of our mission to empower organizations to grow better. Our APIs are designed to enable teams of any shape or size to build robust integrations that help them customize and get the most value out of remote computer vision assessments.

All Spry SDKs and APIs are designed to have a predictable structure.

# Getting Started

The following daigram shows the major components involved in the system.

## Spry Client SDK

Spry Client SDK takes care of the

* Fetching patient details from Spry Interoperable Bridge(supports AthenaHealth) using CareQuality Framework
* Real-time patient monitoring for patients

## Spry Clinical Interface

Using the Spry Clinical Interface, a user i.e. therapist can create customized exercise programs for the patient.

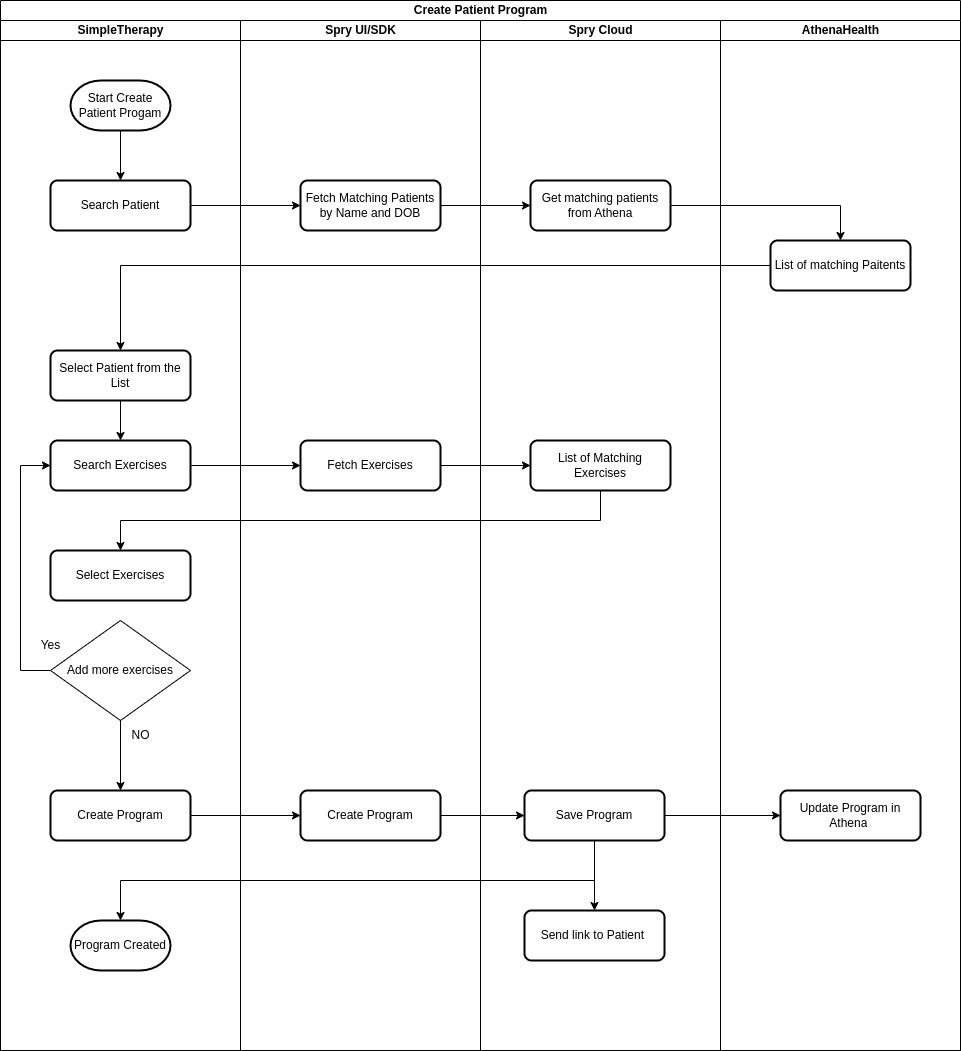
It also facilitates access to assessment reports from patients.

## Spry Cloud

Spry Cloud provides APIs for creating customized assessment programs and real-time patient monitoring using cutting-edge machine learning algorithms.

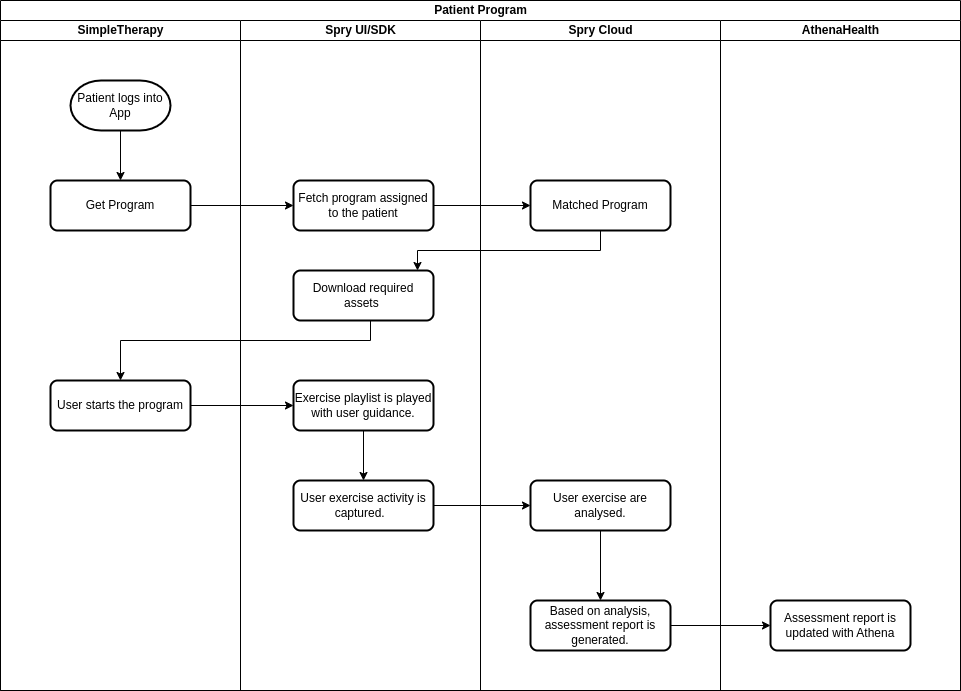
Spry Cloud uses its proprietary interoperable EHR bridge to fetch and send patient data with other EHRs. Currently supported EHRs include Athena, EPIC, and Cerner.

The below flow presents how the therapists will add a custom exercise program to the patient.



The following steps are involved in creating a customized exercise program

1. User (i.e. Therapist) searches for the patient using Name and Date of Birth
2. Spry SDK will return a list of matching patients.
3. User selects the patient.
4. User proceeds to create a customized program for the patient.
5. User searches for exercises and adds them to the program.
6. User selects the repetition, duration, and frequency for each exercise.
7. The created program is sent to AthenaHealth using Spry Interoperable Bridge.
8. The patient is notified about the program via email/SMS/push notification.



The above flow presents how the patient/user will adhere to the prescribed programs. Once prescribed exercises are done, their assessment is available via API and is also synced with AthenaHealth using Spry Interoperability Bridge.

The details for assessment reports is available in API Document

# Introduction

This page is an overview of the Javascript SDK documentation and related resources.

type:video

The Assessment Portal SDK for JavaScript enables developers to build libraries and applications that use Spry’s Assessment Portal content and services. You can use the JavaScript API in the browser and inside Node.js applications on the server.

The AP SDK for JavaScript API Reference Guide provides a JavaScript API. You can use the JavaScript API to build libraries or applications for Node.js or the browser.

# Introduction

This documentation describes how to integrate your iOS app with Spry RTM SDK to build engaging computer vision assessment apps by using the Spry SDK for iOS.

Code and samples for the Spry SDK for iOS will be available on GitHub.

When you use the Spry SDK for iOS, follow the Spry RTM Terms of Use and Privacy Policy.

# App Store Connect Requirements

To provide functionality within the Spry iOS SDK, we may receive and process certain contact, location, identifier, and device information associated with assessment users and their use of your application. The information we receive depends on what SDK features third party applications use.

# Getting Started

This documentation describes how to integrate your iOS app with Spry RTM SDK to build engaging computer vision assessment apps by using the Spry SDK for iOS.

Code and samples for the Spry SDK for iOS will be available on GitHub.

When you use the Spry SDK for iOS, follow the Spry RTM Terms of Use and Privacy Policy.

# App Store Connect Requirements

To provide functionality within the Spry iOS SDK, we may receive and process certain contact, location, identifier, and device information associated with assessment users and their use of your application. The information we receive depends on what SDK features third party applications use.

# Introduction

This documentation describes how to integrate your iOS app with Spry RTM SDK to build engaging computer vision assessment apps by using the Spry SDK for iOS.

Code and samples for the Spry SDK for iOS will be available on GitHub.

When you use the Spry SDK for iOS, follow the Spry RTM Terms of Use and Privacy Policy.

# App Store Connect Requirements

To provide functionality within the Spry iOS SDK, we may receive and process certain contact, location, identifier, and device information associated with assessment users and their use of your application. The information we receive depends on what SDK features third party applications use.

# Getting Started

This documentation describes how to integrate your iOS app with Spry RTM SDK to build engaging computer vision assessment apps by using the Spry SDK for iOS.

Code and samples for the Spry SDK for iOS will be available on GitHub.

When you use the Spry SDK for iOS, follow the Spry RTM Terms of Use and Privacy Policy.

# App Store Connect Requirements

To provide functionality within the Spry iOS SDK, we may receive and process certain contact, location, identifier, and device information associated with assessment users and their use of your application. The information we receive depends on what SDK features third party applications use.

# Overview

The Spry RTM API is organized around [REST](https://en.wikipedia.org/wiki/Representational_state_transfer). Automated messages arrive at your designated address, whenever an activity is generated. Our API has predictable resource-oriented URLs, accepts form-encoded request bodies, returns JSON-encoded responses, and uses standard HTTP response codes, authentication, and verbs.

For activities in the past, or should you wish to fetch data manually, REST endpoints are also available.

Our API accepts JSON-encoded request bodies (where a body is required) or a standard set of query parameters, returns JSON-encoded responses, and uses standard HTTP response codes and verbs.

See Integrations for a full list of API-based integrations. The Spry RTM API differs for every account as we release new versions and tailor functionality. Log in to see docs customized to your version of the API, with your test key and data.

# Authentication

The RTM API uses API keys to authenticate requests. You can view and manage your API keys in the RTM Dashboard.

Test mode secret keys have the prefix sk\_test\_ and live mode secret keys have the prefix sk\_live\_. Alternatively, you can use restricted API keys for granular permissions.

Your API keys carry many privileges, so be sure to keep them secure! Do not share your secret API keys in publicly accessible areas such as GitHub, client-side code, and so forth.

Authentication to the API is performed via HTTP Basic Auth. Provide your API key as the basic auth username value. You do not need to provide a password.

If you need to authenticate via bearer auth (e.g., for a cross-origin request), use -H "Authorization: Bearer rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc" instead of -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc.

All API requests must be made over HTTPS. Calls made over plain HTTP will fail. API requests without authentication will also fail.

# Errors

Spry RTM APIs uses conventional HTTP response codes to indicate the success or failure of an API request. In general: Codes in the 2xx range indicate success. Codes in the 4xx range indicate an error that failed given the information provided (e.g., a required parameter was omitted, a charge failed, etc.). Codes in the 5xx range indicate an error with Spry RTM’s servers (these are rare).

Some 4xx errors that could be handled programmatically (e.g., a card is declined) include an error code that briefly explains the error reported.

### Error Codes

* 200 - OK Everything worked as expected.
* 400 - Bad Request The request was unacceptable, often due to missing a required parameter.
* 401 - Unauthorized No valid API key provided.
* 402 - Request Failed The parameters were valid but the request failed.
* 403 - Forbidden The API key doesn’t have permissions to perform the request.
* 404 - Not Found The requested resource doesn’t exist.
* 409 - Conflict The request conflicts with another request (perhaps due to using the same idempotent key).
* 429 - Too Many Requests Too many requests hit the API too quickly. We recommend an exponential backoff of your requests.
* 500, 502, 503, 504 - Server Errors Something went wrong on Spry RTM’s end.

### Error Types

* api\_error - API errors cover any other type of problem (e.g., a temporary problem with Spry RTM’s servers), and are extremely uncommon.
* card\_error - Card errors are the most common type of error you should expect to handle. They result when the user enters a card that can’t be charged for some reason.
* idempotency\_error - Idempotency errors occur when an Idempotency-Key is re-used on a request that does not match the first request’s API endpoint and parameters.
* invalid\_request\_error - Invalid request errors arise when your request has invalid parameters.

# Session APIs

### Session ID Report

Retrieves the details of a session as taken by the user. Supply the unique identifier of the event, which you might have received in a webhook.

curl -X 'GET' \  
 'https://api.rtm.sprypt.com/v1/report/35faf477' \  
 -H 'accept: application/json'  
 -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc

Here is the sample response

"https://assessment.spryhealth.care/report/35faf477"

### Session ID / Assessment

Retrieves the details of a assessment as taken by the user from each session. Supply the unique identifier of the event, which you might have received in a webhook.

curl -X 'GET' \  
 'https://api.rtm.sprypt.com/v1/100/Squats' \  
 -H 'accept: application/json'  
 -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc

Here is the sample response

{  
 "count": 1,  
 "individual\_reps\_list": [  
 {  
 "rep\_number": 1,  
 "start\_frame\_number": 0,  
 "peak\_frame\_number": 50,  
 "end\_frame\_number": 99,  
 "overall\_accuracy\_measurement": 98,  
 "measurement\_results": [  
 {  
 "type": "peak",  
 "frame": 50,  
 "accuracy": 100,  
 "value": 66,  
 "measurement\_id": 1,  
 "message": "Hip flexion",  
 "priority": 2  
 },  
 {  
 "type": "peak",  
 "frame": 50,  
 "accuracy": 100,  
 "value": 59,  
 "measurement\_id": 2,  
 "message": "Knee flexion",  
 "priority": 1  
 },  
 {  
 "type": "peak",  
 "frame": 50,  
 "accuracy": 92,  
 "value": 5,  
 "measurement\_id": 3,  
 "message": "Keep your knees aligned with your toes",  
 "priority": 2  
 },  
 {  
 "type": "static",  
 "accuracy": 98,  
 "error": {  
 "frame": 81,  
 "accuracy": 69,  
 "message": "Do not lift your toes or heel"  
 },  
 "measurement\_id": 4,  
 "message": "Do not lift your toes or heel",  
 "priority": 2  
 }  
 ]  
 }  
 ]  
}

### Session Information

Retrieves the meta data of a session as taken by the user. Supply the unique identifier of the event, which you might have received in a webhook.

curl -X 'GET' \  
 'https://api.rtm.sprypt.com/v1/100' \  
 -H 'accept: application/json'  
 -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc

Here is the sample response

{  
 "100": 100,  
 "user\_id": 36,  
 "prescription\_id": 9,  
 "assessments": [  
 "Squats",  
 "L Cervical Extension",  
 "R Shoulder flexion"  
 ]  
}

# Prescriber APIs

### Post Prescription

Post the details of a prescription as prescribed for the user. Supply the details of the assessment which the user might have to be prescribed.

curl -X 'POST' \  
 'http://api.rtm.sprypt.com/v1/prescribe' \  
 -H 'accept: application/json' \  
 -H 'Content-Type: application/json' \  
 -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc \  
 -d '{  
 "prescriber\_id": 99,  
 "user\_id": 10,  
 "assessments": [  
 {  
 "assessment\_id": 12,  
 "name": "Squats",  
 "prescribed\_count": 10,  
 "start\_day": "20220101",  
 "end\_day": "20220110"  
 },  
 {  
 "assessment\_id": 10,  
 "name": "R Shoulder flexion",  
 "prescribed\_time": 30,  
 "start\_day": "20220101",  
 "end\_day": "20220110"  
 }  
 ]  
}'

Here is the sample response

{  
 "prescriber\_id": "wqwe09d"  
}

### Get Prescription Information

Get the details of a prescription as prescribed for the user. Supply the unique identifier of the prescription, which you might have received in a webhook.

curl -X 'GET' \  
 'https://api.rtm.sprypt.com/v1/100' \  
 -H 'accept: application/json'  
 -u rtm\_test\_4eC39HqLyjWDarjtT1zdp7dc

Here is the sample response

{  
 "user\_id": 36,  
 "prescription\_id": 9,  
 "assessments": [  
 "Squats",  
 "L Cervical Extension",  
 "R Shoulder flexion"  
 ]  
}