DEVELOPERS PORTAL

TECHNICAL DOCUMENTATION

CONFIDENTIAL

V1.1

2024-07-31

CONFIDENTIAL

Index

INDEX 3

INTRODUCTION 4

REGISTER INTO REHABILIFY 5

APLICATION MANAGEMENT 6

Add new app 6

Edit App 8

Client Id and Secret 8
Creating New Credentials 8
Credentials listing 9

WEBHOOKS 12

Signature 12

Health Status 12

Creating a webhook subscription 13

Webhook events 14
TPS Prescribed 14
TPS Unprescribed 15
TPS Program Assigned 16
TPS Program Unassigned 17

APIS 19

Obtaining a Client Token 19

Add Tasks 19

Remove Tasks 21

Add variables and results 22

USE CASE EXAMPLE 25

Enrolling a patient 25

Configuring exercises or characteristics 25
Directly via TPS 25
Configuration via Webhooks 26

Scheduling tasks into Rehabilify 26

CONFIDENTIAL

Introduction

Rehabilify offers developers a range of tools such as webhooks and APIs, facilitating smooth integration for third-party solutions (TPS). By providing these technologies, Rehabilify ensures that developers can seamlessly connect their systems, allowing for real-time data exchange and enhanced functionality. The webhooks enable automatic updates in response to events within Rehabilify, ensuring that third-party applications are always synchronized with the latest data. Meanwhile, the APIs provide a robust interface for performing a variety of actions, such as retrieving user data or managing tasks, making it easier for developers to create integrated solutions that enhance user experiences. This comprehensive support empowers developers to efficiently integrate and innovate within their applications, leveraging Rehabilify's capabilities to the fullest.

CONFIDENTIAL

Register into Rehabilify

A URL will be shared for TPS to register as a developer in Rehabilify. For security and privacy reasons, this registration must be done in each of the existing instances of Rehabilify, and in which the TPS is interested and authorized to participate.

Figure 1. Register as developer

Once the data has been filled in and validated, you will receive an email with a link to generate the new password to access the platform for managing your Rehabilify.

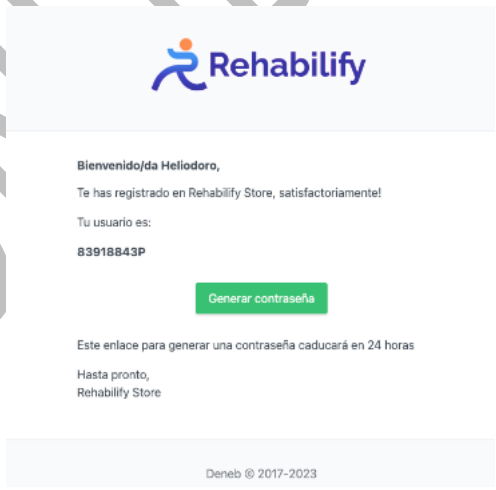

Figure 2. Mail to finish register

CONFIDENTIAL

Aplication management

Once inside the Rehabilify developers portal, the TPS can manage their applications from the corresponding section. Each TPS can have several applications registered.

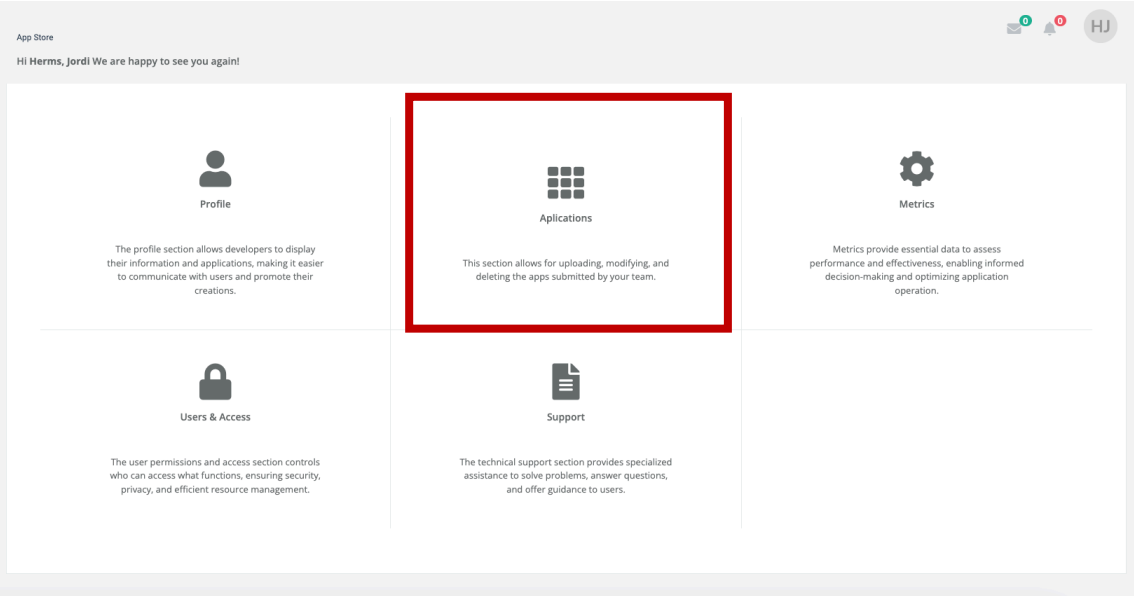

Figure 3. Developers Portal

Add new app

To start registering an application, a user should navigate to the "Applications" section and select "New." This action opens the form displayed in the image for filling out the necessary details.

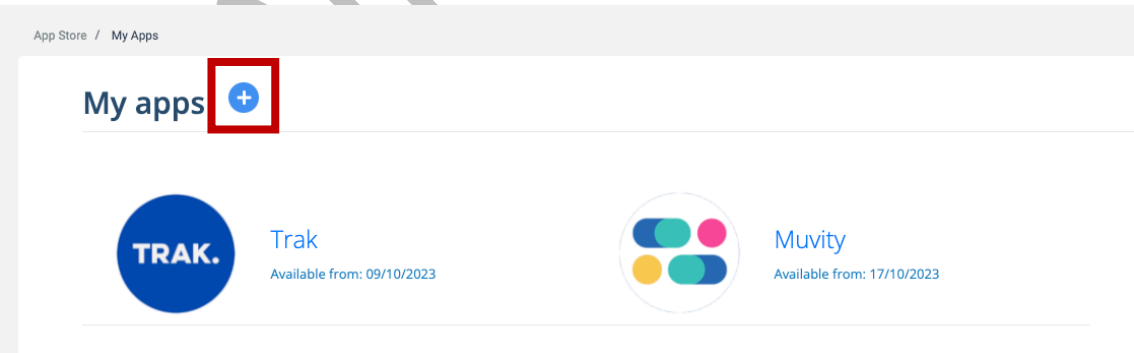

Figure 4. Apps listing

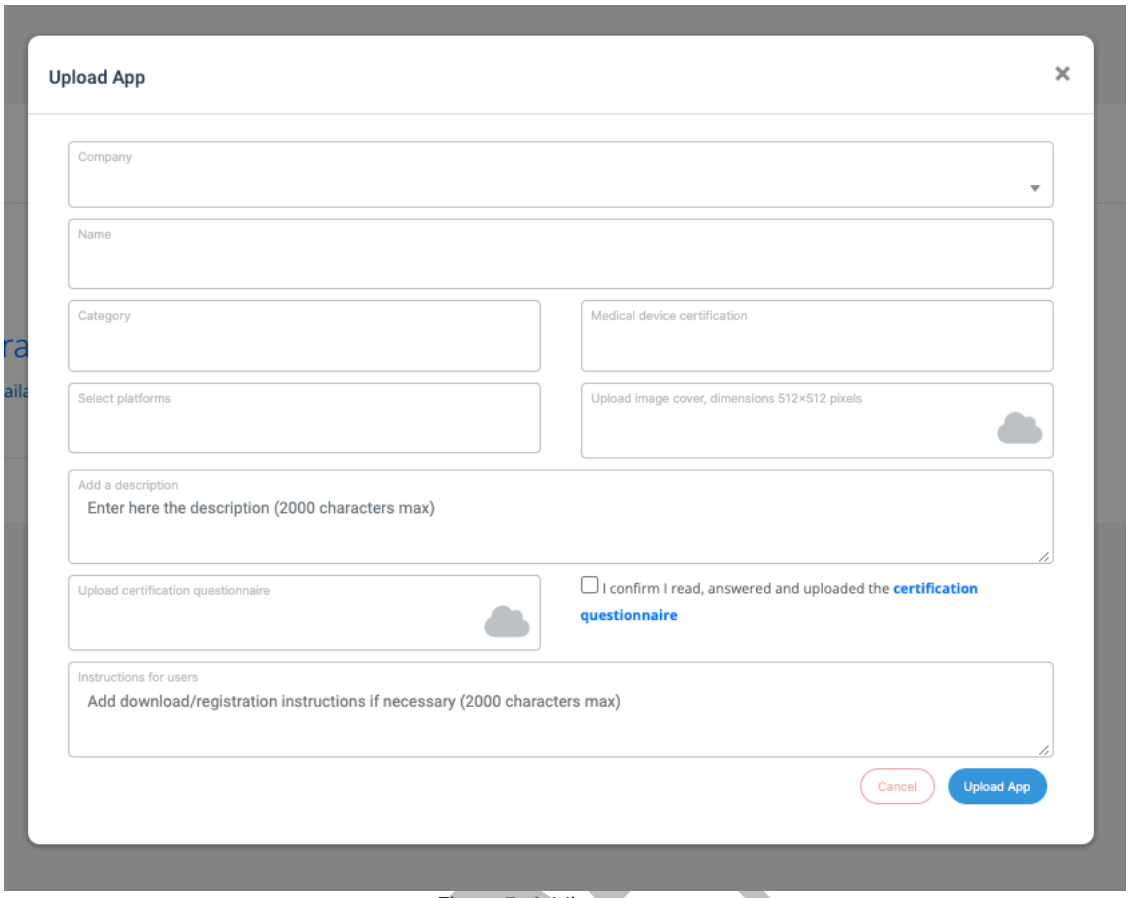

Figure 5. Adding a new app

Form Details:

1. **Company**: This field is for entering the name of the company that owns or is distributing the app.
2. **Name**: Here, the actual name of the app should be entered.
3. **Category**: Users can select the category from a dropdown menu that best describes the app. This could include categories like "Medical Device," "Health Monitoring," "Fitness," etc.
4. **Select Platforms**: This dropdown allows users to choose on which platforms (iOS, Android, Web, etc.) the app is available.
5. **Upload Image Cover**: A field to upload the cover image of the app. The required dimensions are specified (512x512 pixels).
6. **Add a Description**: A text box where a description of the app can be added, limited to 2000 characters. This should encapsulate what the app does and its key features.
7. **Certification questionnaire**: A checkbox for users to confirm they have completed the necessary steps regarding the certification questionnaire. You must download and fill the questionnaire, that will be uploaded next.
8. **Upload Certification Questionnaire**: upload the questionnaire used for compliance or verification purposes.

CONFIDE

9. **Instructions for Users**: An additional field where instructions related to downloading or registration can be provided if necessary, also limited to 2000 characters.

Edit App

In this screen the TPS has the ability to enter or update details about their app.

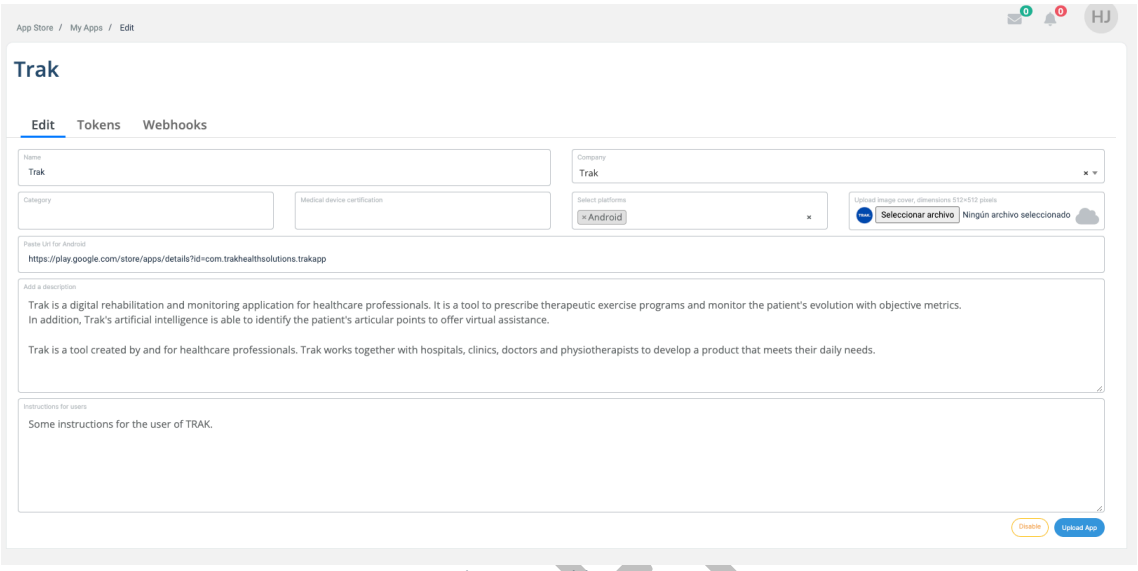

Figure 6. Editing an app

Client Id and Secret

Client IDs and secrets serve as essential credentials for authentication and secure communication between third-party systems and our platform. These credentials are fundamental in establishing a trusted connection, ensuring that only authorized entities can access specific resources.

Creating New Credentials

To obtain a new client ID and secret, TPS providers can initiate the process by clicking the "Create" button.

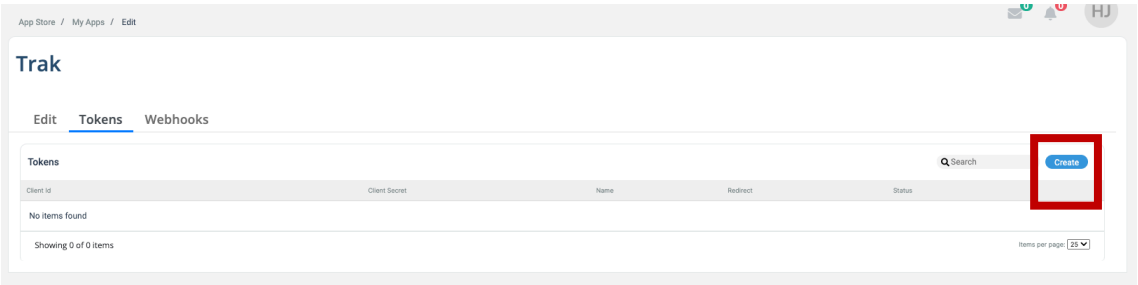

Figure 7. Client id and secret management

CONFIDE

In the pop-up form, they are prompted to enter the "Name" for this. After filling out these fields, clicking "Create" will generate new credentials.

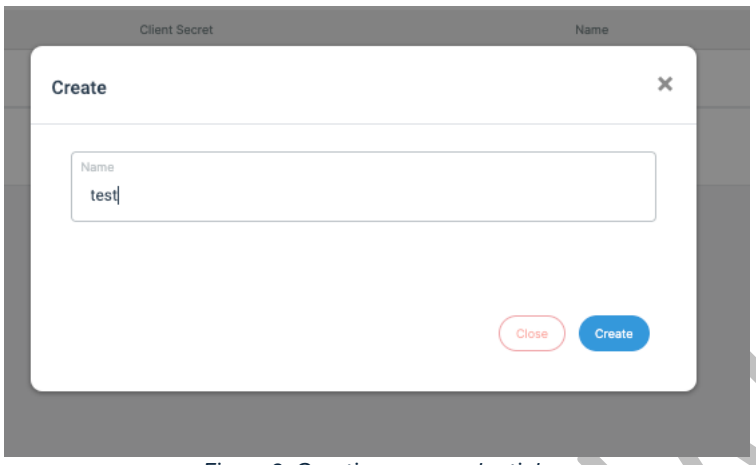

Figure 8. Creating new credentials

Credentials listing

Once created, the new client ID and its associated secret are listed in the credentials section of the interface. This list allows users to easily view, manage, and reference their various credentials.

Each entry in the list displays the name associated with the credentials and provides options to see the secret, edit or delete them, giving users full control over their API keys and ensuring they can update their settings as their integration requirements evolve.

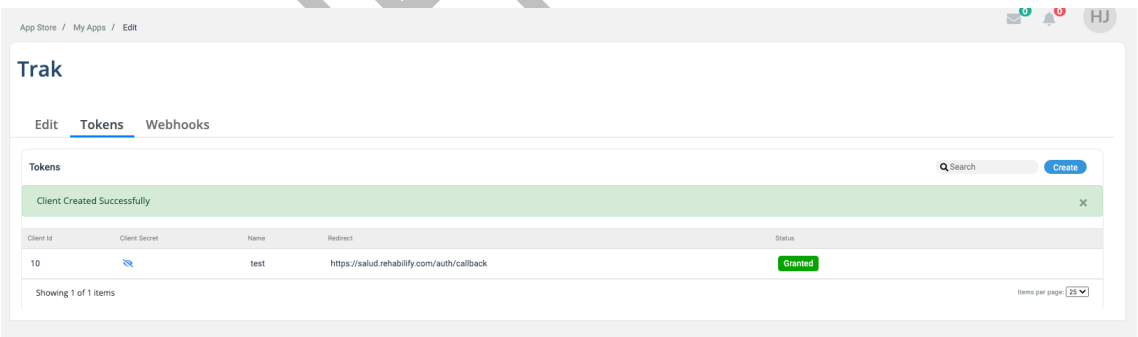

Figure 9. Credentials list

CONFIDENTIAL

On each credential, by clicking on the pencil you can update it

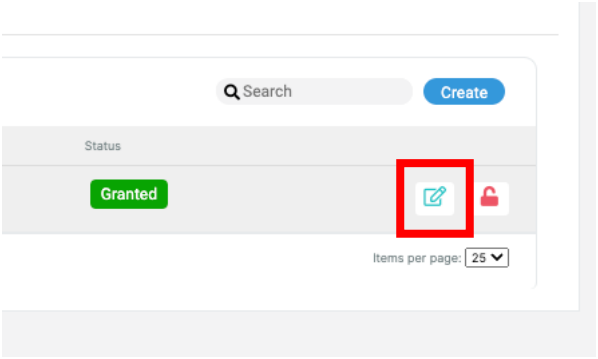

Figure 10. Edit credential

Changing the name or revoke it

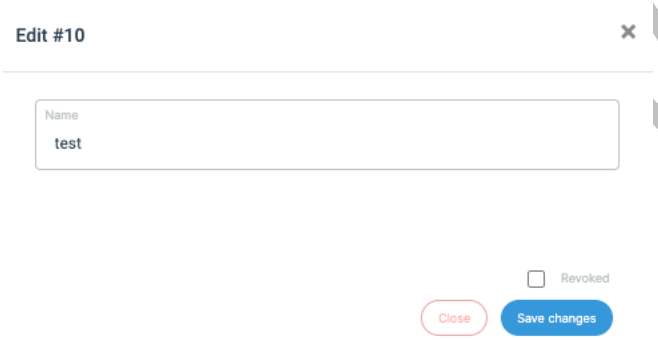

Figure 11. Form to edit the credential

Or also you can directly revoke/enable it by clicking on the lock icon

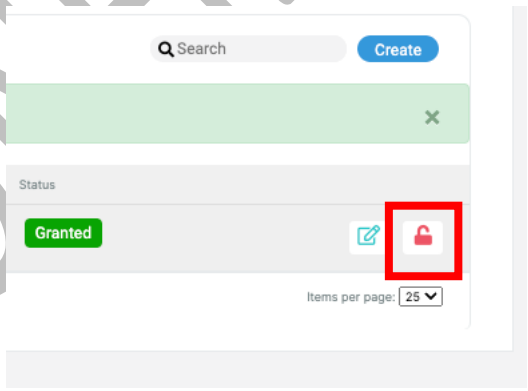

Figure 12. Revoke credential

CONFIDENTIAL

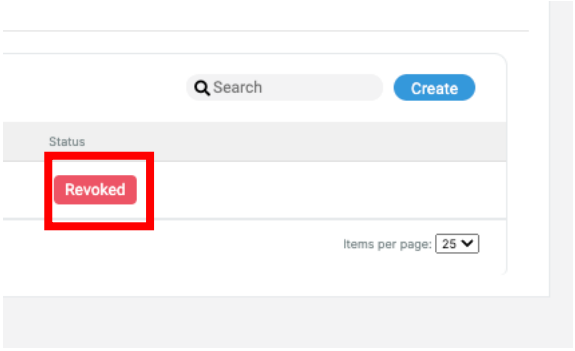

Figure 13. Revoked credential

CONFIDENTIAL

Webhooks

A webhook is a crucial tool at Rehabilify, facilitating real-time communication between our systems and external applications. When significant events occur—such as the creation of a patient profile, initiation of a video call, or assignment of advice to a patient—the webhook automatically sends a notification to a predefined URL. This enables external systems, such as a hospital's Health Information System (HIS) or third-party solutions, to respond immediately and update their records without manual intervention. Integrating webhooks into Rehabilify enhances efficiency and coordination, ensuring that all involved systems are continuously updated with the latest information.

Signature

All requests sent to your webhook endpoints will be signed to verify that the incoming traffic to your servers originates from Rehabilify. To sign our requests, we utilize a Hash-based Message Authentication Code (HMAC). Each webhook that is created will receive a secret, which can be found at any of the CRUD webhook endpoints. When a request is made to your webhook endpoint, a signature is generated by hashing the request body sent with the shared secret. This signature is transmitted in the HTTP 'Signature' header, enabling the client to verify that it was made with the same secret they have access to. To verify the signature, the client can generate a hash signature using the same algorithm and check if the values match.

Example of signature verification:

hash_hmac('sha256', payloadJson, secret)

Health Status

To ensure efficient use of our resources here at Rehabilify, the status of the webhooks you create will be monitored. As long as the endpoints you provide return a successful HTTP status code in a timely manner, your webhook will remain active and in good standing.

If the application receiving the webhook does not send a response with a 2xx status code, the request will be considered failed. A call will also be deemed failed if the remote application does not respond within 3 seconds, although this default timeout can be adjusted at the time of webhook creation.

If a webhook call fails, we will attempt to resend the call two more times. You can set the default number of retry attempts at the time of creating the webhook.

CONFIDENTIAL

To prevent overwhelming the remote application, we will wait a period between each attempt. By default, we wait 10 seconds between the first and second attempt, 100 seconds between the second and third, 1000 seconds between the third and fourth, and so forth. The maximum wait time is 100,000 seconds, which is approximately 27 hours.

Creating a webhook subscription

To subscribe to a webhook, navigate to the Webhooks tab within the application you wish to subscribe, and then click the "Create" button. This will initiate the process for setting up a new webhook subscription.

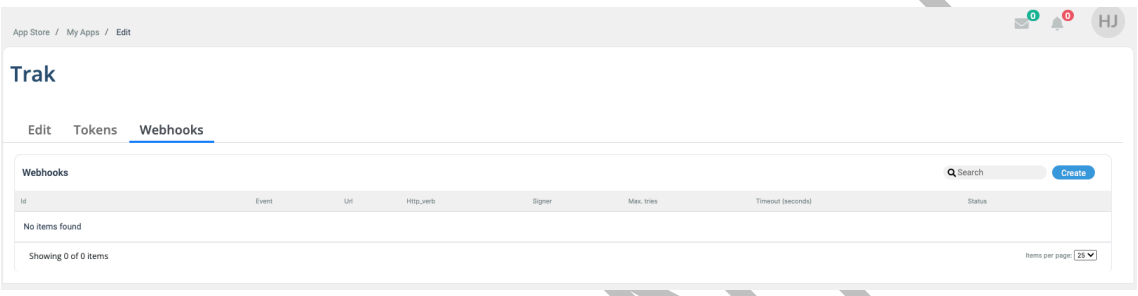

Figure 14. Webhooks tab

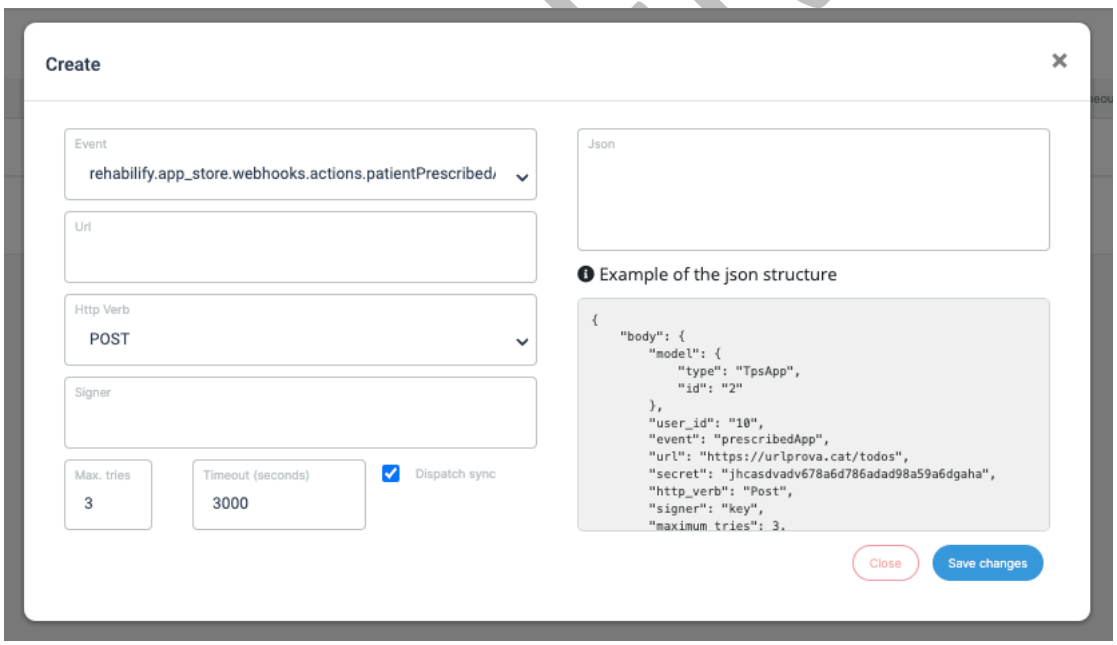

Figure 15. Subscribing to an event

Event Selection: select the specific event you want to subscribe to. For instance, if you want to track when a patient is prescribed a new application, you might select patientPrescribedApp.

Webhook URL: Enter the URL where you want the webhook payloads to be sent. This URL will be the endpoint that receives the JSON data triggered by the selected event.

HTTP Verb: Choose the HTTP method that the webhook will use to send data. Typically, this is a POST request, which is suitable for transmitting data securely to your endpoint.

Signature: Define the passphrase that will be used to sign the webhook data. This field is critical as it ensures the security of the transmitted data by verifying that the messages received are authentic and from a verified source. The signature helps prevent unauthorized access and manipulation of data between your endpoint and the platform.

Max Tries: Enter the maximum number of retry attempts for the webhook if the initial try fails. For example, setting it to 3 means the system will attempt to resend the webhook up to three times in case of failure.

Timeout: Set the timeout duration in seconds. This defines how long the system should wait for a response from your endpoint before considering the attempt as failed.

Dispatch Sync: If this option is available and checked, the webhook will be sent synchronously, awaiting a response before proceeding. If unchecked, it will dispatch asynchronously using a queue system, usually within a few seconds after the event.

Webhook events

TPS Prescribed

The patientPrescribedApp event will be sent when a patient is assigned to a third-party solution (TPS). This notification allows the TPS to undertake necessary actions to integrate this user into their system. Such actions might include registering the user within their system, sending them a welcome email, or any other steps required to fully onboard the patient.

Body	{
	"event": "patientPrescribedApp",
	"app_id" : 1,
	"app_name": "Doole",
	"user_id" : 10,
	"executed_at": "Y-m-d H:i:s"
	}

Params explanation	**event:** This is a string that identifies the type of event that triggered the webhook. In this case, the event is "patientPrescribedApp", indicating that the webhook is being called because an application was prescribed to a patient.

CONFIDENTIAL

	app_id: This is an integer that uniquely identifies the third-party application that has been assigned to the patient. **app_name:** A string representing the name of the third-party application **user_id:** This is an integer that uniquely identifies the Rehabilify user (patient) to whom the app has been prescribed. **executed_at**: This field represents the timestamp of when the event was executed.

TPS Unprescribed

The ` patientUnprescribedApp ` event will be sent when a patient is unassigned from a third-party solution (TPS). This notification enables the TPS to execute necessary actions, such as deregistering the patient from their platform. This helps ensure that the patient's data and access are promptly and securely managed in response to their change in status.

Body	{
	"event": "patientUnprescribedApp",
	"app_id" : 1,
	"app_name": "Doole",
	"user_id" : 10,
	"executed_at": "Y-m-d H:i:s"
	}

Params explanation	**event:** This is a string that identifies the type of event that triggered the webhook. In this case, the event is "patientUnprescribedApp", indicating that the webhook is being called because an application was unprescribed to a patient. **app_id:** This is an integer that uniquely identifies the third-party application that has been assigned to the patient. **app_name:** A string representing the name of the third-party application **user_id:** This is an integer that uniquely identifies the Rehabilify user (patient) to whom the app has been prescribed. **executed_at**: This field represents the timestamp of when the event was executed.

CONFIDENTIAL

TPS Program Assigned

The **tpsProgramAssigned** event is a specific type of notification used within Rehabilify to indicate to a third-party solution (TPS) that a patient has been assigned to a particular program. This could be any type of program—from a simple exercise routine to a more comprehensive treatment or rehabilitation program. When this event is triggered, it signifies that the patient has been formally enrolled in a program tailored to meet their specific health needs or goals.

As a result, the TPS can take appropriate actions, such as incorporating this patient into their system for the required program or exercise. This integration ensures that the patient's care plan is updated and managed effectively, facilitating seamless communication between the primary health system and the third-party provider.

Body	{
	"event": "tpsProgramAssigned",
	"app_id" : 1,
	"app_name": "Doole",
	"user_id" : 10,
	"program_name": "Exercise",
	"program_id": 417
	"executed_at": "Y-m-d H:i:s"
	}

Params explanation	**event:** This is a string that identifies the type of event that triggered the webhook. In this case, the event is "tpsProgramAssigned", indicating that the webhook is being called because a TPS program has been assigned to a patient **app_id:** This is an integer that uniquely identifies the third-party application that has been assigned to the patient. **app_name:** A string representing the name of the third-party application **user_id:** This is an integer that uniquely identifies the Rehabilify user (patient) to whom the app has been prescribed. **program_name:** The name of the program being assigned, it can be configurable, for example "Exercise" or "Therapy group". This tells the type or category of the program. **program_id:** An integer that uniquely identifies the program being assigned.

CONFIDENTIAL

	executed_at: This field represents the timestamp of when the event was executed.

TPS Program Unassigned

The **tpsProgramUnassigned** event functions as a specific type of notification within a system, signaling to a third-party solution (TPS) that a patient has been unassigned or removed from a particular program. This event can occur in various contexts, such as when a patient's treatment plan is updated or changed, when a patient completes a program, or when it's determined that the program is no longer suitable for the patient's needs.

When the **tpsProgramUnassigned** event is triggered, it signifies that the patient is no longer enrolled in the specific program, which might necessitate several follow-up actions by the TPS:

- Update Records: The TPS may need to update their records to reflect that the patient is no longer participating in the program. This ensures that all data remains accurate and up-to-date.

- Cease Activities: Any ongoing activities or interventions associated with the program should be ceased to prevent any unnecessary or unsuitable treatment.

Body	{
	"event": "tpsProgramUnassigned",
	"app_id" : 1,
	"app_name": "Doole",
	"user_id" : 10,
	"program_name": "Exercise",
	"program_id": 417
	"executed_at": "Y-m-d H:i:s"
	}

Params explanation	**event:** This is a string that identifies the type of event that triggered the webhook. In this case, the event is "tpsProgramUnassigned", indicating that the webhook is being called because a TPS program has been unassigned to a patient **app_id:** This is an integer that uniquely identifies the third-party application that has been assigned to the patient. **app_name:** A string representing the name of the third-party application

CONFIDENTIAL

	user_id: This is an integer that uniquely identifies the Rehabilify user (patient) to whom the app has been prescribed. **program_name:** The name of the program being assigned, it can be configurable, for example "Exercise" or "Therapy group". This tells the type or category of the program. **program_id:** An integer that uniquely identifies the program being assigned. **executed_at**: This field represents the timestamp of when the event was executed.

CONFIDENTIAL

APIs

In today's interconnected digital landscape, APIs play a pivotal role in facilitating seamless communication and integration across diverse systems and applications. Our APIs provide the foundational building blocks that allow external systems and developers to interact efficiently with our platform. By offering a set of clear and robust interfaces, we enable a wide range of functionalities—from managing user data and processing transactions to enabling real-time updates and interactions. This section is designed to guide you through the various APIs we offer, providing all the necessary information to understand their functions, how to integrate them into your applications, and best practices for maximizing their potential. Whether you're looking to automate processes, enhance functionality, or integrate new services, our APIs are your gateway to expanding your capabilities in a secure and efficient manner.

Obtaining a Client Token

This API endpoint is designed to retrieve an access token from our system, which is necessary for making subsequent API requests, such as obtaining an access token for a patient user. This token ensures that all communications between client applications and our server are secure and authenticated.

Endpoint Details	https://*********/oauth/token
Method	POST
Params	grant_type: The action to be performed, in this case: client_credentials. client_id: The client's identifier. client_secret: The client's secret key.

Response	{
	"token_type": "Bearer",
	"expires_in": 1200,
	"access_token": *********
	}

Explanation	token_type: The type of token, which is Bearer. expires_in: The expiration time of the token in seconds access_token: The actual token string, which is a JWT (JSON Web Token) that should be used in the Authorization header for subsequent requests to the API.

Add Tasks

This endpoint is enabled for adding exercises or tasks to a user's profile. It is designed to facilitate the integration of personalized fitness or therapeutic routines directly into the user's daily or weekly schedule. By utilizing this endpoint, external applications can seamlessly post new activities that are tailored to the user's health needs and goals.

CONFIDENTIAL

Endpoint Details	https://*********/api/v1/center/programable-plays
Method	POST

Request Params	{
	"data": [
	{
	"programable_id": "ID|NULL",
	"origin": {
	"type": "Exercise",
	"id": "exercise_id"
	},
	"user": {
	"id": "user_id"
	},
	"scheduled_date": "Y-m-d H:i:s",
	"details": "{
	"title": "Exercise Example",
	"url": "https://tpsserver.com/presignedURL"
	}"
	}
	]
	}

Response	{
	"success": [
	{
	"programable_play_id": 68,
	"programable_id": "ID|NULL",
	"origin": {
	"type": "Exercise",
	"id": "Exercise_id"
	},
	"user": {
	"id": "user_id",
	},
	"scheduled_date": "Y-m-d H:i:s",
	"details": "{
	"title": "Exercise Example",
	"url": "https://tpsserver.com/presignedURL"
	}"
	}
	],
	"failed": [
	{
	"programable_id": "ID|NULL",
	"origin": {
	"type": "Exercise",

CONFIDENTIAL

	"id": "exercise_id"
	},
	"user": {
	"id": "user_id",
	},
	"scheduled_date": "Y-m-d H:i:s",
	"details": "{
	"title": "Exercise Example",
	"url": "https://tpsserver.com/presignedURL"
	}"
	"errors": [
	{
	"campo": "programable_id",
	"error": [
	"programmable_id has not been found on database",
	]
	}
	]
	}
	]
	}

Explanation	Each **success** object corresponds to an exercise that has been successfully added, while each **failed** object contains details about what went wrong, helping in troubleshooting and ensuring data integrity.

Remove Tasks

This endpoint enables the deletion of exercises or tasks for a users. It allows for the precise management and timely removal of content that is no longer needed or relevant, ensuring that the user interface remains streamlined and effective.

Endpoint Details	https://*********/api/v1/center/programable-plays
Method	DELETE

Params	{
	"data": [
	{
	"programable_play_id": "ID|Required, this identifier is obtained from the API that saves the programable: api/v1/client/programable-plays POST"
	}
	]
	}

Response	{ "success": [

CONFIDENTIAL

	{
	"programable_play_id": "ID"
	}
	],
	"failed": [
	{
	"programable_play_id": "ID generated when the record was created in Rehabilify",
	"errors": [
	{
	"field": "programable_play_id",
	"error": [
	"The programable_play_id is not found in the database"
	]
	}
	]
	}
	]
	}

Explanation	Each object in the **success** array corresponds to a programable play that has been successfully deleted, while each object in the **failed** array details what went wrong, particularly if the specified ID does not exist in the database.

Add variables and results

In the field of health and rehabilitation, the capability to dynamically add and manage variables such as joint angles, muscle strength, and range of motion is essential for delivering personalized care and monitoring patient progress. This endpoint allows developers to effectively integrate and manage these health-related variables and their corresponding results through our system.

Endpoint Details	https://*********/api/v1/tps/user/elements
Method	POST

Params	{
	"execution_model": {
	"id": 10, -> identificador del ejercicio en el TPS
	"name": "Ejercicio 1" -> nombre del ejercicio
	},
	"values": [
	{
	"ext_id": "abcdef",
	"user_id": 7,

CONFIDENTIAL

	"date": "2023-12-11 17:26:00",
	"element_id": 1,
	"value": "300"
	},
	{
	"ext_id": "abcdefg",
	"user_id": 7,
	"date": "2023-12-11 17:26:00",
	"element_id": 2,
	"value": "77"
	}
	]
	}

	execution_model **id:** exercise or task identifier in the TPS. It is used to group all the records that have been generated in the same exercise. **name:** exercise or task name in the TPS. **values** **ext_id:** variable id in TPS. **user_id:** user identifier on the Rehabilify platform. **date:** date of the value **element_id:** variable id in Rehabilify **value:** variable value

Response	{
	"message": "Data has been processed",
	"success": [],
	"failed": [
	{
	"ext_id": "abcdef",
	"user_id": 7,
	"date": "2023-12-11 17:26:00",
	"element_id": 1,
	"value": "300"
	},
	{
	"ext_id": "abcdefg",
	"user_id": 7,
	"date": "2023-12-11 17:26:00",
	"element_id": 2,
	"value": "77"
	}
	]
	}

CONFIDENTIAL

CONFIDENTIAL

Use case example

In this use case, we explore how a Third-Party Solution (TPS) that has already registered on our platform and set up an application can utilize webhooks to enhance its services.

Enrolling a patient

Once the application is enabled to be prescribed from the Rehabilify platform, a healthcare professional can proceed with prescribing it:

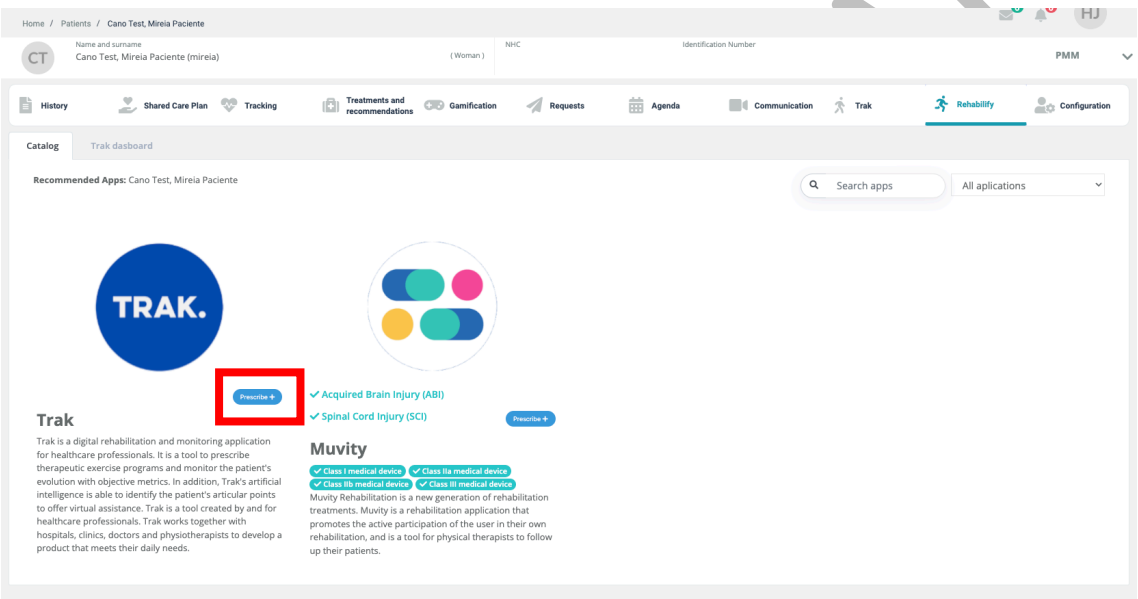

Figure 16. Prescribing a TPS

This TPS has actively subscribed to the patientPrescribedApp webhook event to ensure timely, real-time updates. Once a healthcare professional prescribes an application via the Rehabilify platform, the system immediately triggers the configured webhook associated with the patientPrescribedApp event. Upon receiving this notification, the TPS can swiftly integrate the patient into their solution. This seamless process facilitates immediate patient setup and customization of services based on the specific details of the prescription.

Configuring exercises or characteristics

Once a patient from Rehabilify is enrolled in a Third-Party Solution (TPS), configuring their exercise regimen or characteristics can begin through two distinct methods: directly via the TPS and through webhooks.

Directly via TPS

Healthcare professionals can log into the TPS's website to directly manage and set up exercises or specific features for the patient. Once the exercises are configured,

the TPS utilize the Rehabilify's API to schedule these exercises on the Rehabilify platform.

Configuration via Webhooks

In Rehabilify, professionals have the ability to set up different 'verticals,' each aligned with a specific TPS. These verticals can organize the services and features that various third-party solutions offer. Within each vertical, healthcare professionals can associate specific exercise packages or settings with the patient's profile. Examples of these packages include "shoulder rehabilitation" for targeted physical recovery or "thigh strengthening" for muscle building. Additional features might involve varying levels of patient monitoring, such as "intense monitoring" for close supervision, suitable for acute conditions or post-operative recovery, and "normal monitoring" for less critical management. Once these packages or features are linked to a patient, a webhook automatically notifies the corresponding TPS using the tpsProgramAssigned event.

```
{
 "event": "tpsProgramAssigned",
 "app_id" : 1,
 "app_name": "Doole",
 "user_id" : 10,
 "program_name": "Exercise",
 "program_id": 417
 "executed_at": "Y-m-d H:i:s"
}
```

In the body of the webhook, the *program_id* parameter is the TPS id for the "program" or configuration. For example, if "Arm's rehabilitation" in the TPS has the id 417, the webhook will be sent with *program_id* 417.

Scheduling tasks into Rehabilify

Once a patient is enrolled in a Third-Party Solution (TPS) and their exercise regimen is configured, the next step involves the TPS scheduling specific tasks for the patient using Rehabilify's API. This is done by making a POST request to the API endpoint designated for adding programmable plays, such as exercises or activities. The tasks are then directly visible to the patient within their application, providing a clear and organized schedule of activities.

CONFIDENTIAL

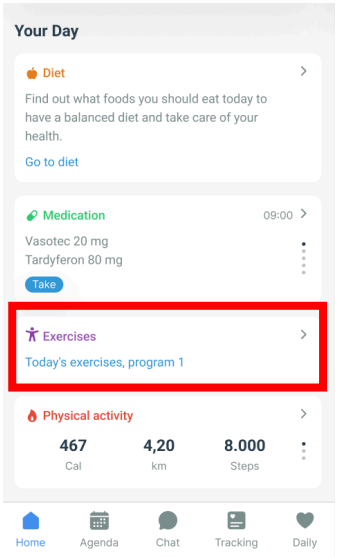

Figure 17. Tasks are visualized into the patient's app

In cases where the patient can perform the task directly from a web page, Rehabilify is prepared to open it within the application via iframe. To do this, the API for adding tasks, which as detailed above in the APIs section, is a POST call to https://*********/api/v1/center/programmable-plays, with the following structure:

```
{
 "data": [
  {
   "programable_id": "ID|NULL",
   "origin": {
    "type": "Exercise",
    "id": "exercise_id"
   },
   "user": {
    "id": "user_id"
   },
   "scheduled_date": "Y-m-d H:i:s",
   "details": "{
    "title": "Exercise Example",
    "url": "https://tpsserver.com/presignedURL"
   }"
  }
 ]
}
```

Within details, the TPS can put a title for the task, as well as adding the URL, preferably using signed URLs to increase security.

CONFIDENTIAL

Figure 19. TPS web page opened via iframe using presigned URL

Sending values to Rehabilfy

When the user completes their exercises or the TPS receives new data, the TPS can send the values to Rehabilify, so they can be processed and visualized by professionals. To do this, the TPS will use the "Add variables and results" API.

For security reasons, a Third-Party Solution (TPS) is not permitted to directly create "variable types" in Rehabilify. A variable type could be, for example, "Temperature" or "Oxygen Saturation." Instead, the TPS and the healthcare facility must collaborate to decide which variables will be integrated into Rehabilify. Once these variables have been agreed upon, a healthcare professional from the facility is responsible for creating these variables within the platform if they not exists. After these variables are established, their identifiers are shared with the TPS. This allows the TPS to correctly map these variables and utilize the IDs provided by Rehabilify when making API calls.

CONFIDENTIAL