

**Identificación: API-FP-WA-V3.0.0**

Contenido

[Control del documento 3](#_Toc134480340)

[Control de cambios 3](#_Toc134480341)

[SERVICIOS 4](#_Toc134480342)

[Servicio Register 4](#_Toc134480343)

[Servicio Login 7](#_Toc134480344)

[Servicio Logout 9](#_Toc134480345)

[Servicio Qr 10](#_Toc134480346)

[Servicio Message 11](#_Toc134480347)

[Servicio User 14](#_Toc134480348)

[Servicio Users 15](#_Toc134480349)

[Servicio DisableUser 16](#_Toc134480350)

[Servicio EnableUser 18](#_Toc134480351)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Control del documento | | | | |
|  | **Nombre** | **Cargo** | **Área** | **Fecha** |
| **Responsable** | VASSALLO, Mario Nicolás | Developer | desarrollo | 10/03/2023 |

|  |  |  |  |
| --- | --- | --- | --- |
| Control de cambios | | | |
| **Versión Nº** | **Fecha** | **Descripción del cambio** | **Solicito** |
| V1.1.0 | 17/03/2023 | se pueden hacer envíos a múltiples números | Fredy |
| V2.0.0 | 25/03/2023 | se pueden conectar múltiples cuentas | Fredy |
| V3.0.0 |  | Detallado en doc: [Requeriomets wsp](https://docs.google.com/document/d/11bc3GGK5fMq2jTnyEc2PcmCWOAjxbHPrGbtc8sqcPF0/edit?usp=share_link) | Fredy |

SERVICIOS**:**

# Servicio Register

**1 -** Descripción

Este service maneja la creación de nuevos usuarios en una base de datos. Verifica si el nombre de usuario ya existe, hashea la contraseña del usuario, inserta los datos del usuario en la base de datos y responde con el resultado de la operación.

**2 -** Notas Aclaratorias:

Requiere una conexión establecida a una base de datos. Puedes encontrar el script SQL necesario para crear esta tabla en el proyecto. Además, utiliza variables de entorno a través de un archivo. env en el directorio raíz para configurar la conexión a la base de datos.

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location 'http://localhost:3000/register' \

--header 'Content-Type: application/json' \

--data '{

"username": "Usuario",

"password": "123456"

}'

● Ambiente Testing: No habilitado.

● Parámetros: username, password

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/register**

○ Nombre: username

○ Detalle: nombre del usuario a crear

○ Tipo: varchar

○ Ejemplo: “Nico”

○ Nombre: password

○ Detalle: password del usuario

○ Tipo: varchar

○ Ejemplo “123456”

**5.** Parámetros de Salida**:**

○ Nombre: id

○ Detalle: numero id

○ Tipo: int

○ Ejemplo: 32

○ Nombre: username

○ Detalle: usuario creado en el sistema

○ Tipo: varchar

○ Ejemplo “Usuario”

○ Nombre: password

○ Detalle: contraseña

○ Tipo: varchar

○ Ejemplo: "$2b$10$.vCettM4M2ALFIksW/ZNyeD1kvOQRcLQ3vahMYYMIWMZJTWuwlgYu"

○ Nombre: client

○ Detalle: cliente asignado para qr

○ Tipo: varchar

○ Ejemplo: client1

○ Nombre: num\_msgs

○ Detalle: cantidad de mensaje autorizado por usuario

○ Tipo: int

○ Ejemplo: 50

○ Nombre: created\_at

○ Detalle: fecha y hora de la creación del usuario

○ Tipo: date

○ Ejemplo: "2023-05-08T21:04:49.517Z"

○ Nombre: updated\_at

○ Detalle: fecha y hora de la actualización del usuario

○ Tipo: date

○ Ejemplo: "2023-05-08T21:04:49.517Z"

○ Nombre: disabled

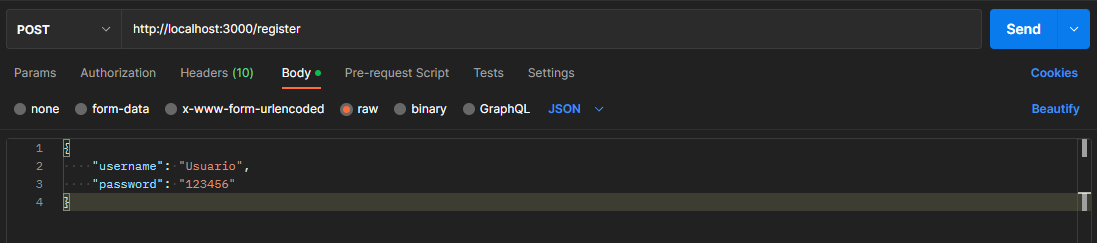
○ Detalle: determina si el usuario este habilitado

○ Tipo: boolean

○ Ejemplo: null

**6 -** Ejemplo de Condición de éxito**:**

Entrada:



Salida:

{

"id": 32,

"username": "Usuario",

"password": "$2b$10$.vCettM4M2ALFIksW/ZNyeD1kvOQRcLQ3vahMYYMIWMZJTWuwlgYu",

"client": null,

"num\_msgs": 50,

"created\_at": "2023-05-08T21:04:49.517Z",

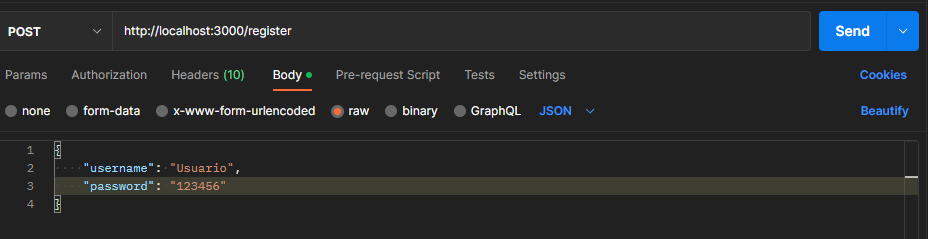
"updated\_at": "2023-05-08T21:04:49.518Z",

"disabled": null

}

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:



Salida:

El nombre de usuario ya está en uso

# Servicio Login

**1 -** Descripción

Este service brinda un conjunto de funciones que permiten la autenticación segura de usuarios, el control de acceso y la administración de usuarios en un sistema

**2 -** Notas Aclaratorias:

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location 'http://localhost:3000/login' \

--header 'Content-Type: application/json' \

--header 'Cookie: connect.sid=s%3AR8sNXPJPYcWOGpiH3EOp1ef80aHnAN4h.Ce0X4exHIO4Bq2fmzyVZVlQsH11yrM4%2BfmIaFMxyRXk' \

--data '{

"username": "Usuario",

"password": "123456"

}'

● Ambiente Testing: No habilitado.

● Parámetros: username, password

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/login**

○ Nombre: username

○ Detalle: nombre del usuario a crear

○ Tipo: varchar

○ Ejemplo: “Usuario”

○ Nombre: password

○ Detalle: password del usuario

○ Tipo: varchar

○ Ejemplo “123456”

**5.** Parámetros de Salida**:**

○ Nombre: respuesta

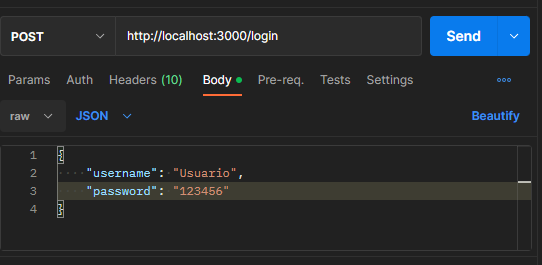
○ Detalle: información de logeo

○ Tipo: varchar

○ Ejemplo: “Inicio de sesión exitoso”

**6 -** Ejemplo de Condición de éxito**:**

Entrada:

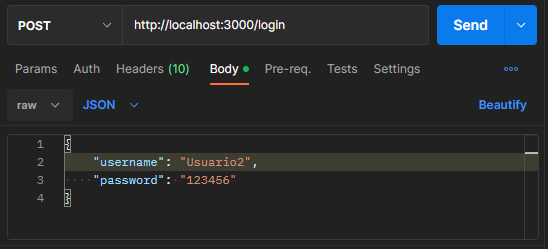


Salida:

Inicio de sesión exitoso

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:



Salida:

Nombre de usuario o contraseña incorrectos

# Servicio Logout

**1 -** Descripción

Este service brinda al usuario poder salir del sistema.

**2 -** Notas Aclaratorias:

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location --request POST 'http://localhost:3000/logout' \

--header 'Cookie: connect.sid=s%3AVZZtZUmkOT-K-VTq4Ys8oldQuQx7M9A2.vDpgS%2FEokbTPcqT2%2F0Wz4e5T4NWQQ0xCMFO%2Fanls82w'

● Ambiente Testing: No habilitado.

● Parámetros:

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/logout**

**5.** Parámetros de Salida**:**

○ Nombre: respuesta

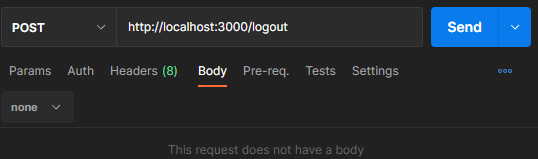
○ Detalle: información de deslogeo

○ Tipo: varchar

○ Ejemplo: “Sesión cerrada de Usuario exitosamente”

**6 -** Ejemplo de Condición de éxito**:**

Entrada:



Salida:

Sesión cerrada de Usuario exitosamente

**7 -** Ejemplo de Condición de fracaso**:**

# Servicio Qr

**1 -** Descripción

Este service brinda una imagen QR para sincronizar WhatsApp del usuario con el sistema .

**2 -** Notas Aclaratorias:

En el Front la imagen generada debe de actualizarse cada cierto tiempo, también debe de tomar el valor de client y pasarlo por la url.

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: GET

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location 'http://localhost:3000/qrcode/client3' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4'

● Ambiente Testing: No habilitado.

● Parámetros:

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4**. Parámetros de Entrada**:**

○ **/qrcode/client?**

**5.** Parámetros de Salida**:**

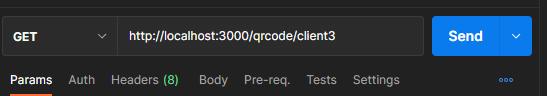
○ Nombre: qr

○ Detalle: imagen de qr

○ Tipo: imagen

**6 -** Ejemplo de Condición de éxito**:**

Entrada:

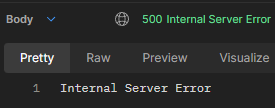


Salida:



**7 -** Ejemplo de Condición de fracaso**:**

Salida:



# Servicio Message

**1 -** Descripción

Este service permite enviar mensajes a través de WhatsApp, aplicando límites diarios para mantener un control sobre la cantidad de mensajes que un usuario puede enviar en un día. También se encarga de almacenar los mensajes en una base de datos y enviarlos a través de la plataforma correspondiente.

**2 -** Notas Aclaratorias:

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location 'http://localhost:3000/sendMessage' \

--header 'Content-Type: application/json' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4' \

--data '{

"numbers": [

"5492615011946"

],

"message": "Hola, mando msj como Nico"

}'

● Ambiente Testing: No habilitado.

● Parámetros: numbers, message

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/** **sendMessage**

○ Nombre: numbers

○ Detalle: número o números a enviar mensaje

○ Tipo: array

○ Ejemplo: "5492615011946"

○ Nombre: message

○ Detalle: mensaje a enviar

○ Tipo: varchar

○ Ejemplo "Hola, mando msj como Nico"

**5.** Parámetros de Salida**:**

○ Nombre: success

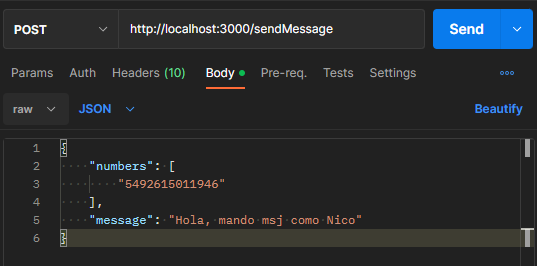
○ Detalle: Información de mensaje

○ Tipo: varchar

○ Ejemplo: "Mensajes enviados con éxito"

**6 -** Ejemplo de Condición de éxito**:**

Entrada:



Salida:

{

"success": "Mensajes enviados con éxito"

}

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:



Salida:

{

"error": "Los números proporcionados no son válidos"

}

# Servicio User

**1 -** Descripción

Este service muestra **el** usuario conectado.

**2 -** Notas Aclaratorias:

Solo muestra el usuario en esa sesión.

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: GET

● Observaciones: Se sugiere probar con Postman.

●cURL:

curl --location 'http://localhost:3000/user' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4'

● Ambiente Testing: No habilitado.

● Parámetros:

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/** **user**

**5.** Parámetros de Salida**:**

○ Nombre: on

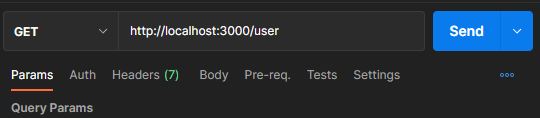
○ Detalle: Información de mensaje

○ Tipo: varchar

○ Ejemplo: "Usuario"

**6 -** Ejemplo de Condición de éxito**:**

Entrada:



Salida:

{

"On": "Usuario"

}

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:

Salida:

# Servicio Users

**1 -** Descripción

Este service muestra **los** usuarios conectados.

**2 -** Notas Aclaratorias:

Solo muestra los usuarios que estén en el sistema.

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: GET

● Observaciones: Se sugiere probar con Postman.

●cURL

curl --location 'http://localhost:3000/users' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4'

● Ambiente Testing: No habilitado.

● Parámetros:

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/** **users**

**5.** Parámetros de Salida**:**

○ Nombre: users

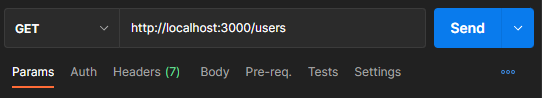
○ Detalle: Información de mensaje

○ Tipo: varchar

○ Ejemplo: "Usuario", "Luigi"

**6 -** Ejemplo de Condición de éxito**:**

Entrada:



Salida:

{

"users": [

"Usuario",

"Luigi"

]

}

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:

Salida:

# Servicio DisableUser

**1 -** Descripción

Este service inhabilita a un usuario.

**2 -** Notas Aclaratorias:

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL

curl --location 'http://localhost:3000/disableUser' \

--header 'Content-Type: application/json' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4' \

--data '{

"username": "Usuario"

}'

● Ambiente Testing: No habilitado.

● Parámetros: username

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/** **disableUser**

○ Nombre: username

○ Detalle: nombre del usuario

○ Tipo: varchar

○ Ejemplo: "Usuario”

**5.** Parámetros de Salida**:**

○ Nombre: respuesta

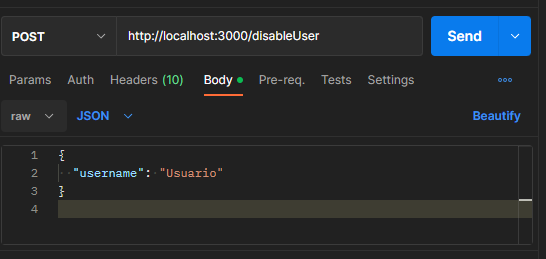
○ Detalle: Información de mensaje

○ Tipo: varchar

○ Ejemplo: "Usuario inhabilitado correctamente"

**6 -** Ejemplo de Condición de éxito**:**

Entrada:

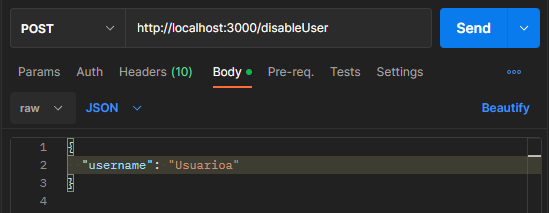


Salida:

Usuario inhabilitado correctamente

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:



Salida:

Usuario no encontrado

# Servicio EnableUser

**1 -** Descripción

Este service habilitar a un usuario bloqueado.

**2 -** Notas Aclaratorias:

**3 -** Reglas de operación**:**

● Tipo servicio: REST

● Método soportado: POST

● Observaciones: Se sugiere probar con Postman.

●cURL

curl --location 'http://localhost:3000/disableUser' \

--header 'Content-Type: application/json' \

--header 'Cookie: connect.sid=s%3AYNS1z7T5Z550MQtakODjeXaeV7ehXxE-.WNBSYWDsj45OkykKSR8Kz1wdO9yZM5t4nRop9fZmrg4' \

--data '{

"username": "Usuario"

}'

● Ambiente Testing: No habilitado.

● Parámetros: username

● Ambiente Producción: No habilitado.

● URL producción: No habilitado.

● URL repositorio: <https://github.com/facil-pos/ApiWA>

**4.** Parámetros de Entrada**:**

○ **/** **enableUser**

○ Nombre: username

○ Detalle: nombre del usuario

○ Tipo: varchar

○ Ejemplo: "Usuario”

**5.** Parámetros de Salida**:**

○ Nombre: respuesta

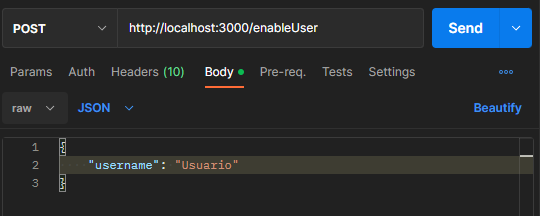
○ Detalle: Información de mensaje

○ Tipo: varchar

○ Ejemplo: " Usuario habilitado correctamente"

***6 -*** Ejemplo de Condición de éxito**:**

Entrada:

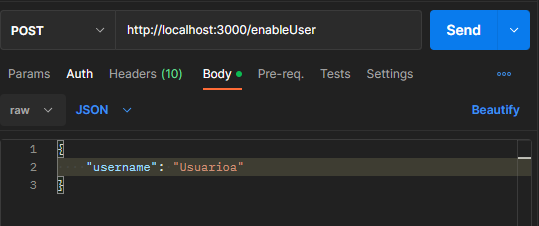


Salida:

Usuario habilitado correctamente

**7 -** Ejemplo de Condición de fracaso**:**

Entrada:



Salida:

Usuario no encontrado