MEXAL

PASSCOM

MANUALE WEBAPI

23/07/2025 v2.26

PASSEPARTOUT®
Software gestionale

WEBAPI v2.26

Sommario

1. INTRODUZIONE

1.1. COSA È

WebApi è, come suggerisce il nome, una collezione di API Passepartout basate su tecnologia Web tramite protocollo https. Il servizio è fruibile utilizzando richieste https secondo lo stile architetturale RESTful: **R**epresentational **S**tate **T**ransfer. Sono implementati tutti i metodi CRUD (**C**reate, **R**ead, **U**pdate, **D**elete) per mezzo dei metodi http **POST**, **GET**, **PUT**, **DELETE**.

Secondo il paradigma REST il servizio è fortemente orientato alle entità (gli archivi del gestionale) le quali saranno raggiungibili mediante path specifici.

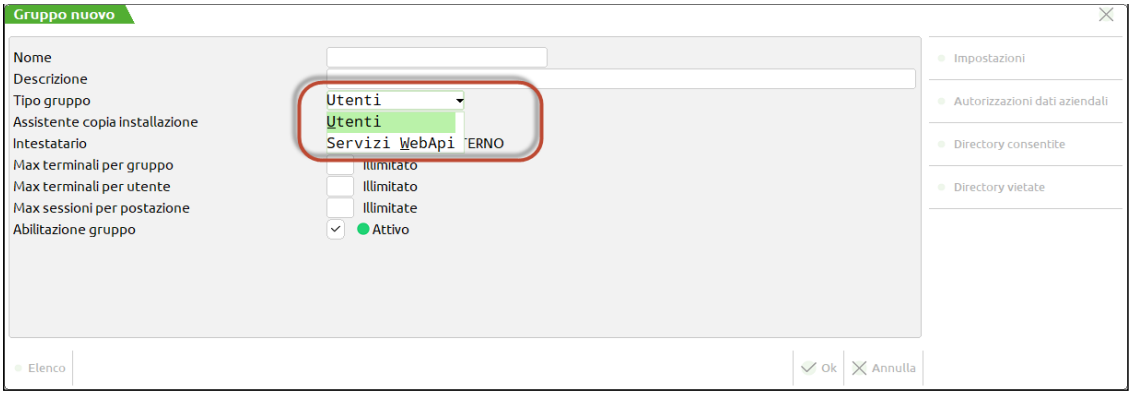
1.2. GLOSSARIO

Di seguito la terminologia ed il significato dei termini utilizzati in questo manuale.

- *risorsa*: oggetto presente sul gestionale Mexal/Passcom su cui possono essere effettuate operazioni
- *end-point*: entità alla fine di una connessione
- *entità*: risorsa. Entità, risorsa ed end-point verranno trattati come sinonimi all'interno di questo manuale
- *http*: **H**ypertext **T**ransfer **P**rotocol
- *https*: **H**ypertext **T**ransfer **P**rotocol over **S**ecure Socket Layer. Nel corso di questo manuale potrà capitare di riferirsi indistintamente ad http o https per definire alcuni concetti. WebAPI gestisce solo richieste tramite protocollo sicuro https.
- *http request*: azione (POST, GET, PUT, DELETE) fatta su una risorsa identificata da un dato url
- *header*: parte della *http request* in cui vengono impostati metadati necessari al completamento della richiesta stessa
- *body*: corpo di una *http request* in cui sarà presente il dato da scambiare
- *query string*: è la parte di un URL che contiene dei dati da passare in input ad un servizio
- *JSON*: **J**ava**S**cript **O**bject **N**otation, è un formato standard adatto all'interscambio di dati tra applicazioni client/server
- *installazione*: ci si riferisce al programma gestionale Mexal/Passcom
- *ambiente live*: installazione presente in server farm Passepartout (cloud)
- *ambiente locale*: installazione presente su un server locale alla rete del cliente (housing)
- *front-end:* parte di un programma responsabile dell'acquisizione dei dati di ingresso
- *back-end:* parte di un programma che elabora i dati acquisiti
- *cookie:* è una stringa di testo di piccole dimensioni inviata da un web server ad un web client e poi rimandata indietro dal client al server; tipicamente è utilizzata per implementare meccanismi di identificazione di un client presso un server

1.3. COME SI USA

WebApi è disponibile sia in ambiente locale che live dopo aver attivato la suite MDS tramite YouPass. L'attivazione del servizio abiliterà in Mexal/Passcom un gruppo particolare di utenti "Servizi WebApi". Solo gli utenti appartenenti a gruppi di questo tipo potranno utilizzare solo ed esclusivamente i servizi WebApi.



WebApi locale: è possibile accedere su installazione locale specificando l'indirizzo del server più la porta su cui è in ascolto il servizio. Il servizio deve essere abilitato lato server Mexal/Passcom tramite il menu *Servizi->Configurazioni->Configurazione moduli->WebApi*. In questa videata sarà possibile verificare la corretta abilitazione del modulo e la porta assegnata al servizio. L'accesso sarà possibile specificando nell'header della richiesta https le credenziali di accesso più altri parametri per identificare l'azienda e la data di esercizio.

WebApi Live: in ambiente live, se attivato da codice contratto, il servizio è già disponibile e raggiungibile tramite l'indirizzo https://services.passepartout.cloud/. In questo caso l'autenticazione avviene sempre tramite credenziali di accesso nell'header della richiesta https (come per le installazioni locali) ma usando come *base address* **https://services.passepartout.cloud/webapi/risorse** (vedi sezione dedicata)

Il gruppo Servizi WebAPI gestisce un numero massimo di 3 utenti: ogni utente del gruppo Servizi WebApi avrà a disposizione un pool di massimo 5 servizi contemporanei con un time-out di 5 minuti per ogni servizio (a parte il primo che rimane sempre attivo). Questo vuol dire che lo stesso utente WebAPI potrà accedere contemporaneamente solo da 5 client differenti (o 5 richieste concorrenti dallo stesso client), le successive richieste verranno rifiutate fino a che uno dei servizi del pool non si libera.

GESTIONE USO ESCLUSIVO

Lato gestionale (Mexal/Passcom) in tutte le parti del programma in cui è richiesto l'uso esclusivo (es. LiveUpdate fase di switch-to, anagrafica azienda, configurazione moduli..etc), la procedura chiederà la chiusura di eventuali processi WebAPI attivi. Questo si rende necessario in quanto, a differenza di altri terminali con interfaccia o di quelli senza interfaccia ma che si chiudono per time-out, i servizi WebAPI per loro natura non hanno interfaccia e hanno sicuramente un processo che rimane sempre attivo per velocizzare le connessioni. Se il processo WebAPI sta eseguendo una elaborazione si chiuderà non appena l'elaborazione sarà terminata, mentre se il processo è in uno stato *idle* verrà immediatamente chiuso.

In questo modo la procedura ad uso esclusivo può continuare senza interruzioni legate alla presenza di altri terminali.

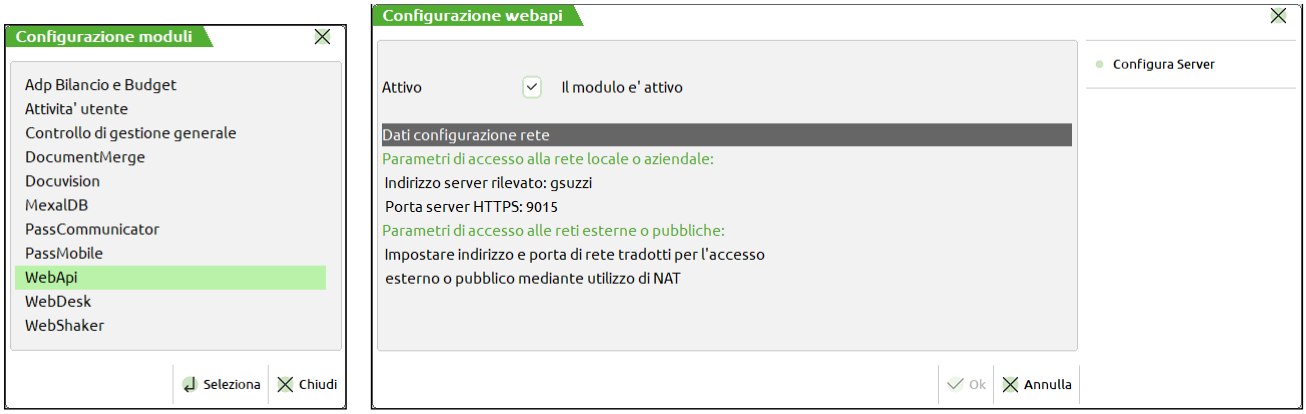
2. FORMATO DELLA HTTP REQUEST

Il formato dei dati è di tipo JSON e lo standard JSON prevede che la codifica del charset sia di tipo utf-8.

Eventuali dati non codificati correttamente in utf-8 generano un errore. E' necessario quindi usare opportune librerie di encoding utf-8 per essere sicuri di scambiare dati con WebAPI sempre usando la codifica corretta.

2.1. INSTALLAZIONI LOCALI

Per utilizzare WebApi in installazioni locali è necessario attivare il modulo dal menu Servizi - Configurazioni - Configurazione moduli – WebApi.



Le informazioni necessarie per l'accesso sono:

URL = <indirizzo_server>
PORTA = <porta_server_https>
USER = <utente_di_mexal_passcom>
PWD = <password_utente_mexal_passcom>

Se la richiesta prevede l'utilizzo di un comando che coinvolge una determinata azienda, è necessario fornire anche le seguenti informazioni:

AZIE = <azienda>
SOTTOAZ = <sottoazienda> (opzionale)
ANNO_AZ = <anno_di_apertura>
MAGAZZINO=<numero_magazzino_default> (opzionale)

NOTA: se non viene utilizzato il numero di magazzino, tutte le operazioni che coinvolgono il magazzino utilizzeranno come magazzino di default il numero 1.

Se sull'installazione locale è attivo anche il controllo delle credenziali di sistema operativo è necessario utilizzare anche queste credenziali

USER_SYS = <utente_di_sistema>
PWD_SYS = <password_utente_di_sistema>

L'indirizzo per raggiungere WebApi in locale dovrà essere formato in questo modo:

https://URL:PORTA/webapi/risorse/

Esempi:

PUT https://miodominio:9004/webapi/risorse/clienti/codice_cliente (aggiorna il cliente: dati nel corpo della richiesta)

POST https://miodominio:9004/webapi/risorse/articoli (crea un nuovo articolo: dati nel corpo della richiesta)

GET https://miodominio:9004/webapi/risorse/articoli (legge tutti gli articoli)

DELETE https://miodominio:9004/webapi/risorse/articoli/codice_articolo (cancella un articolo)

Le credenziali di accesso USER e PWD devono essere formattate in questo modo:

- concatenare USER e PWD formando la seguente stringa: "USER:PWD" es. "ADMIN:MEXAL"
- fare un encoding in base64 della stringa ottenuta al passo precedente

Nell'header della richiesta https dovranno poi essere inserite le informazioni ottenute dai passi precedenti in questo modo:

```
'Authorization': 'Passepartout <credenziali_b64>'
'Content-type': 'application/json'
'Coordinate-Gestionale': 'Azienda=<AZIE> SottoAzienda=<SOTTOAZ> Anno=<ANNO_AZ> Magazzino=<MAGAZZINO>'
```

Nel caso l'installazione gestisca anche l'ulteriore autenticazione tramite credenziali di sistema operativo (login=1) devono essere aggiunte nella chiave 'Authorization' codificate in base64 analogamente a quanto fatto per le credenziali dell'utente Mexal/Passcom.

L'header in questo caso avrà una struttura di questo tipo:

```
'Authorization': 'Passepartout <credenziali_b64> <credenziali_sistema_b64>'
'Content-type': 'application/json'
'Coordinate-Gestionale': 'Azienda=<AZIE> SottoAzienda=<SOTTOAZ> Anno=<ANNO_AZ> Magazzino=<MAGAZZINO>'
```

Il Content-type (opzionale solo per comandi GET e DELETE) prevede come charset di default utf-8 e non è necessario specificarlo. Se però si vuole specificare anche il charset è necessario che sia utf-8: 'Content-type': 'application/json;charset=utf-8'. Qualsiasi altro tipo di charset verrà rifiutato con un errore.

Di seguito due esempi di come deve essere formattato un header di una richiesta https su installazioni locali.

Caso di autenticazione con solo le credenziali dell'utente dell'installazione:

```
{
        'Authorization': 'Passepartout QURNSU46TUVYQUw=',
        'Content-type': 'application/json',
        'Coordinate-Gestionale': 'Azienda=DEM Anno=2021 Magazzino=2'
}
```

Caso di autenticazione in cui sono necessarie anche le credenziali di sistema operativo (login=1):

```
{
        'Authorization': 'Passepartout QURNSU46TUVYQUw= QURNSU46QURNSU4=',
        'Content-type': 'application/json',
        'Coordinate-Gestionale': 'Azienda=DEM Anno=2021'
}
```

Nel caso di installazioni locali, l'accesso alle risorse tramite protocollo sicuro https avviene utilizzando un certificato autoprodotto (self-signed). Le librerie utilizzate dall'applicazione client dovranno quindi poter gestire questo caso.

In alternativa, se non fosse possibile utilizzare i certificati self-signed, è necessario dotare l'installazione di certificati validi (non self signed). Questo è possibile farlo inserendo i certificati tramite il menu Servizi - Configurazioni - Configurazione moduli – WebApi nella sezione Configurazione Server (in modo del tutto analogo a quanto avviene per WebDesk).

2.2. INSTALLAZIONI LIVE

Per poter accedere in ambiente Live, i parametri necessari per l'accesso sono i seguenti:

https://services.passepartout.cloud/webapi/risorse

DOMINIO = <dominio_della_installazione_live>
USER = <utente_di_mexal_passcom>
PWD=<password_utente_mexal_passcom>

Se la richiesta prevede l'utilizzo di un comando che coinvolge una determinata azienda, è necessario fornire anche le seguenti informazioni:

***AZIE** = <azienda>*
***SOTTOAZ** = <sottoazienda> (opzionale)*
***ANNO_AZ** = <anno_di_apertura>*
***MAGAZZINO**=<numero_magazzino_default> (opzionale)*

Come per le installazioni locali, le credenziali di accesso USER e PWD devono essere formattate in questo modo:

- concatenare USER e PWD formando la seguente stringa: "USER:PWD" (es. "ADMIN:ADMIN")
- fare un encoding in base64 delle credenziali

Nell'header della richiesta https dovranno essere inserite le informazioni ottenute dai passi precedenti, dove rispetto alle installazioni locali nell'elemento Authorization va specificato anche il dominio nella forma *Dominio=<MIO_DOMINIO>*

'Authorization': 'Passepartout <credenziali_b64> Dominio=<DOMINIO>'
'Content-type': 'application/json'
'Coordinate-Gestionale': 'Azienda=<AZIE> SottoAzienda=<SOTTOAZ> Anno=<ANNO_AZ>',

Di seguito un esempio di come deve essere formattato un header di una richiesta https in ambiente live.

```
{
        'Authorization': 'Passepartout QURNSA46QQ== Dominio=MIODOMINIO',
        'Content-type': 'application/json',
        'Coordinate-Gestionale': 'Azienda=DEM Anno=2021'
}
```

3. COMANDI E SCAMBIO DEI DATI

Tutti gli esempi e le descrizioni dei comandi da questo punto in poi avranno lo stesso formato, sia per installazioni locali che live, lasciando sottointeso il *base address*. Per semplicità quindi tratteremo il path in questo modo:

https://base_address/webapi/risorse/<nome_risorsa>

dove *base_address* potrà essere:

- <indirizzo_server>:<porta_https> nel caso di installazioni locali
- services.passepartout.cloud nel caso di installazioni live

Per tutti i casi di autenticazione citati, l'interazione con Mexal/Passcom avviene mediante i seguenti metodi http:

POST: utilizzato esclusivamente per creare una nuova entità (es. creazione di un nuovo cliente)

GET: utilizzato per leggere una entità o una collezione di entità (es. la lista di tutti i clienti)

PUT: utilizzato esclusivamente per aggiornare una entità (es. modifica di una descrizione di un articolo)

DELETE: utilizzato per cancellare una entità (es. eliminazione di un cliente)

I metodi POST e PUT prevedono anche un *payload* contenuto nel *body* della richiesta https in formato JSON in cui saranno inseriti i dati da inserire o aggiornare, mentre per i comandi di tipo GET e DELETE un eventuale contenuto viene ignorato.

3.1. TIPO DI COMANDI E SPAZIO DEI NOMI

Mexal/Passcom prevede due tipi di comandi:

- Comandi che richiedono necessariamente l'apertura di una azienda in un dato anno
- Comandi sovra-aziendali che possono essere eseguiti anche senza l'accesso ad una particolare azienda

Nel primo caso, nell'header di tutte le richieste https devono essere specificati anche i parametri di accesso alla particolare azienda (o sottoazienda), come indicato nel capitolo 2.

Nel secondo caso tali parametri possono essere omessi in quanto non verranno controllati dal servizio WebAPI.

Tutti i nomi sono in alfabeto minuscolo senza indicazione sul formato del dato ed eventuali nomi composti saranno separati dal carattere *underscore* (_).

I campi di tipo *array* possono arrivare ad un massimo di 3 livelli. Ogni elemento di un campo di tipo array è identificato a sua volta da un elemento di tipo array in cui sono specificati gli indici dei vari livelli ed il valore in ultima posizione.

Si tratta quindi di array misti in cui i primi valori sono sempre interi (indici), l'ultimo valore può essere intero o stringa a seconda del tipo di dato.

Ad esempio per un campo array di 3 livelli, per identificare un elemento, la struttura sarà di questo tipo:

```
"campo_array": [
        [1,2,1,"a"],        // valore "a" in posizione 1 nel primo livello, 2 nel secondo e 1 nel terzo
        [2,1,4,"b"]         // valore "b" in posizione 2 nel primo livello, 1 nel secondo e 4 nel terzo
]
```

Per i nomi delle variabili, il tipo di dato ed il significato usare il parametro il metodo GET con il parametro *info=true* nella query string.

3.2. METODO GET

Secondo l'architettura REST i comandi di tipo GET non devono specificare alcun contenuto nel corpo della richiesta http. In ogni caso anche se presente deve essere ignorato.

Per capire quali sono le entità accessibili è disponibile una entità *HELP* utilizzabile con metodo *GET* che restituisce un JSON con tutte le entità ed i metodi disponibili.

GET https://base_address/webapi/risorse/help

Verrà tornato un JSON con l'elenco di tutti gli end-point gestiti. Di default verranno restituiti il path, la descrizione ed il metodo disponibile. Se si vogliono ulteriori informazioni è possibile usare il parametro *extended* in query string

GET https://base_address/webapi/risorse/help?extended=true

```
{
   "next": true/false,                          // specifica se la risorsa gestisce il parametro next
   "filter": true/false,                        // specifica se la risorsa gestisce i filtri
   "regexp": "<path_risorsa>",                  // path e la regular expression associata alla risorsa
   "descrizione": "<descrizione_risorsa>",      // descrizione della risorsa
   "coordinate": true/false,                    // specifica se per l'accesso alla risorsa servono le coordinate aziendali
   "versione": "<versione_gestionale>",         // specifica da che versione contenitore (Mexal/Passcom) è disponibile la risorsa
   "method": "<metodo_http>",                   // specifica il metodo http disponibile per la
   "max": true/false,                           // specifica se la risorsa gestisce il parametro max
   "chiavi": [ <elenco_chiavi> ],               // specifica eventuali chiavi per referenziare gli elementi di una risorsa
   "location": true/false,                      // specifica se la risorsa ritorna un tag location nell'header
   "fields": true/false                         // specifica se la risorsa gestisce il parametro fields
}
```

Facendo GET su un'entità senza specificare altri parametri, il JSON di risposta conterrà tutti i record con tutti i campi dell'archivio.

GET https://base_address/webapi/risorse/articoli

Se invece si vuole recuperare il singolo record bisogna specificare l'ID/Codice della particolare entità

GET https://base_address/webapi/risorse/articoli/11AA55

Nota: la differenza sostanziale tra i comandi GET che tornano una collezione di elementi rispetto a quelli che tornano un singolo elemento sta nel fatto che nel primo caso il risultato potrebbe avere alcuni campi in meno. Inoltre gli array sono tutti al massimo di 2 dimensioni, mentre nel secondo caso gli array possono avere fino a un massimo di 3 dimensioni. Per i dettagli sui campi disponibili usare info=true in query string.

Nel caso si voglia ridurre la dimensione del payload della risposta per rendere le request più efficienti (soprattutto in caso di risposte molto corpose), è possibile chiedere al servizio WebAPI di comprimere i dati. È sufficiente indicare nell'header della request la chiave *Accept-Encoding:gzip* come da standard http.

In questo caso il body del response sarà un archivio compresso nel formato gzip contenente il JSON con i dati. Sarà quindi poi cura del client decomprimere i dati ricevuti.

Esempio di autenticazione in installazione locale

```
{
        'Authorization': 'Passepartout QURNSU46TUVYQUw=',
        'Content-type': 'application/json',
```

```
        'Coordinate-Gestionale': 'Azienda=DEM Anno=2021 Magazzino=2'
        'Accept-Encoding': 'gzip'
}
```

QUERY STRING

È possibile utilizzare dei filtri o impostare parametri particolari usando le *query string.*

- ***fields***
  Per specificare i soli campi che si vuole vengano restituiti bisogna utilizzare l'attributo *fields*

GET https://base_address/webapi/risorse/articoli/11AA55?fields=codice,descrizione

Nota: è sempre consigliato estrarre i soli campi necessari in modo da velocizzare l'elaborazione lato server, soprattutto nel caso che la risorsa da gestire sia molto corposa.

- ***max***
  È possibile specificare il numero massimo di record restituiti nel caso si richieda una collezione di risorse. In *query string* sarà necessario utilizzare l'attributo *max*

GET https://base_address/webapi/risorse/articoli?fields=codice,descrizione&max=3

La risposta, oltre ai record letti, in questo caso contiene anche un elemento che identifica quale sarà il successivo record da cui partire per eventuali successive letture.


```
{
   [
      {
         "codice": "123ABC",
         "descrizione": "Pacco pignoni"
      },
      {
         "codice": "123456",
         "descrizione": "Cambio SRAM"
      },
      {
         "codice": "ABCDEF",
         "descrizione": "Gomma MAXISS ARDENT"
      }
   ],
   "next": "0F45AD69"
}
```


Il metodo GET legge sempre tutti i record che il server riesce a processare entro 30s oppure entro una dimensione massima di 50MB. Se il risultato viola una di queste 2 condizioni, il servizio WebAPI inizierà a paginare restituendo nel response l'elemento ***next*** indicante il prossimo record da cui partire per completare la lettura.

Se in query-string è utilizzato il parametro ***max*** questo sarà sempre prioritario, a patto che il risultato non violi una delle 2 condizioni sopra citate (es. se ho 10.000 record, ma ne voglio leggere solo 1.000 alla volta impostando max=1000, se il sistema impiega più di 30s o il payload della response cuba più di 50MB, vengono restituiti in ogni caso i soli record che soddisfano entrambe le condizioni, ad esempio 700 e non 1000)

- ***next***
  Il parametro next permette di specificare il punto di partenza per il recupero dei record successivi in una lista di risultati. Ogni volta che una risposta GET restituisce un valore next, questo valore indica il record da cui partire per la successiva richiesta.

GET
https://base_address/webapi/risorse/articoli?fields=codice,descrizione&next=57415f65626537663962352d313534382d313166302d623566662d666632366238356565393038-1101000000000000

I valori next sono gestiti tramite un sistema di semafori per evitare conflitti tra richieste concorrenti. Ogni richiesta è identificata da un **session ID** univoco, che viene concatenato alla chiave del record per determinare il punto di partenza della query. È possibile riutilizzare lo stesso valore di next più volte fino a quando l'intero archivio non è stato iterato. Tuttavia, quando viene raggiunta l'ultima chiave next, il semaforo viene resettato e, se vengono inviate chiavi next precedenti, la richiesta restituirà un errore.

Si consiglia di utilizzare immediatamente i valori di next restituiti, senza conservarli per un uso futuro, poiché potrebbero scadere o diventare obsoleti. Utilizzare tempestivamente le chiavi next garantisce l'integrità e l'affidabilità del processo di recupero dei dati.

- ***info***
  Per ogni risorsa è possibile ottenere un elenco di tutti i campi disponibili ed il loro significato. È sufficiente specificare nel path della query string l'attributo ***info***=*true*, in questo caso tutte le altre query string (se presenti) verranno ignorate.

GET https://base_address/webapi/risorse/articoli?info=true

- ***encoding***
  nel caso di risorse che prevedano nel codice caratteri speciali (nello specifico slash "/" e backslash "\") e che quindi andrebbero a comporre il path della risorsa, non è possibile inserire il codice in chiaro nel path ma deve essere codificato. In questo modo vengono superati problemi di sicurezza relativi alla presenza di questi 2 caratteri (nello specifico soprattutto per quanto riguarda il backslash non è stato possibile gestirlo con l'url-encoding standard in quanto viene in ogni caso rifiutato). Non tutte le risorse possono essere gestite con encoding: tramite l'HELP è possibile sapere se una risorsa ha la possibilità o meno di fare encoding delle proprie chiavi valutando il valore del campo *encodable: true/false*.

  Le codifiche disponibili sono 2:

  - base32: deve essere indicata in query string con *encoding=base32*
  - hex: deve essere indicata in query string con *encoding=hex*

GET https://base_address/webapi/risorse/articoli/F4QFY===?encoding=base32

dove la stringa *F4QFY===* è la codifica in base32 del codice articolo "/ \".

GET https://base_address/webapi/risorse/articoli/2f205c?encoding=hex

dove la stringa *2f205c* è la codifica in esadecimale del codice articolo "/ \".

Nel caso di inserimento (POST) di un codice che contiene caratteri *slash* o *backslash*, nel tag *location* della *response* risultante, il path, non potendo essere rappresentato con url-encoding di questi caratteri, viene automaticamente proposto con *encoding=hex.* In tutti gli altri casi invece eventuali caratteri speciali vengono codificati con url-encoding come avviene normalmente.

Location: /webapi/risorse/articoli/415254492f53?encoding=hex

- ***L'encoding si applica solo a codici di tipo alfanumerico (come ad esempio i codici articolo). Per codici di tipo numerico (es. id ubicazioni) non è previsto alcun encoding ed il parametro encoding, se specificato, viene ignorato***
- ***L'encoding deve essere utilizzato tutte le volte che si deve referenziare una entità che presenti slash e backslash nel proprio codice (tipicamente la risorsa articoli) quindi anche per i metodi PUT e DELETE***
- ***Se si vuole continuare ad usare l'url-encoding è possibile farlo (fermo restando che slash e backslash non saranno gestiti) ma non in combinazione con un eventuale encoding***

Nel caso di GET su una collezione di entità è possibile impostare anche filtri più evoluti oltre all'utilizzo di quelli specificati in *query string* (*fields*, *max* e *next).*

I filtri dovranno essere impostati nel corpo della richiesta in formato JSON. Dato però che, come precedentemente indicato, il metodo GET non prevede l'utilizzo di alcun contenuto nel body della http request, i filtri saranno utilizzati sfruttando il metodo POST.

Ogni entità ha a disposizione una risorsa denominata *ricerca* su cui sarà possibile specificare opportuni filtri nel body della request.

POST https://base_address/webapi/risorse/articoli/ricerca (vedi entità RICERCA)

Ulteriori parametri in query string specifici per particolari tipi di risorse (es. ubicazioni o liste di prelievo) sono dettagliati nei relativi capitoli.

3.3. METODO POST

Il metodo POST deve essere utilizzato esclusivamente per la creazione di nuove entità. Nel path non deve essere mai specificato il codice della risorsa da creare. Nel body della richiesta https deve essere presente un JSON con i dati da inserire. Nel caso debba essere specificato un codice per la creazione della risorsa (vedi anagrafica articoli) bisogna aggiungerlo nel body della richiesta https.

Se la creazione avviene con successo verrà tornato un *http status code 201 (Created).*

POST https://base_address/webapi/risorse/clienti

```
{
   "codice": "501.AUTO", //codice creato in automatico sul mastro indicato
   "ragione_sociale": "Cliente di prova",
   "tp_nazionalita": "I",
   // altri campi obbligatori
}
```

Creazione con codice specificato dall'utente

POST https://base_address/webapi/risorse/articoli

```
{
   "codice": "ARTICOLO1",
   "descrizione": "Articolo di prova"
   //altri campi obbligatori
}
```

Il *response* di una *http request* conterrà nell'header della risposta un tag *location* comprendente il path della nuova entità appena creata.

Creazione di un articolo

POST https://base_address/webapi/risorse/articoli

Body della richiesta https:

```
{
   "codice": "ARTICOLO1",
   "descrizione": "Articolo di prova"
   //altri campi obbligatori
}
```

Header della risposta https:

```
{
```

```
        // altri tag
        'Content-type': 'application/json; charset=utf-8',
        'Location': 'https://base_address/webapi/risorse/articoli/ARTICOLO1',
}
```

3.4. METODO PUT

Il metodo PUT deve essere utilizzato per modificare/aggiornare delle risorse già esistenti. In questo caso è obbligatorio specificare sempre il codice della risorsa nel PATH, mentre i dati devono essere specificati nel body della *http request*. Non è previsto alcun contenuto nel *body* della risposta https. In caso di successo verrà solamente restituito un *http status code 204 (No Content)*.

PUT https://base_address/webapi/risorse/articoli/11AA55

Body della richiesta https:

```
{
   "descrizione": "Descrizione modificata"
   //altri campi
}
```

3.5. METODO DELETE

Il metodo DELETE deve essere utilizzato per eliminare una risorsa esistente. In questo caso è obbligatorio specificare sempre il codice della risorsa nel PATH. Non è previsto alcun contenuto nel *body* della risposta https. In caso di successo verrà solamente restituito un *http status code 204 (No Content)*.

DELETE https://base_address/webapi/risorse/articoli/11AA55

3.6. CONTROLLO SULLA VALIDITÀ DEI DATI

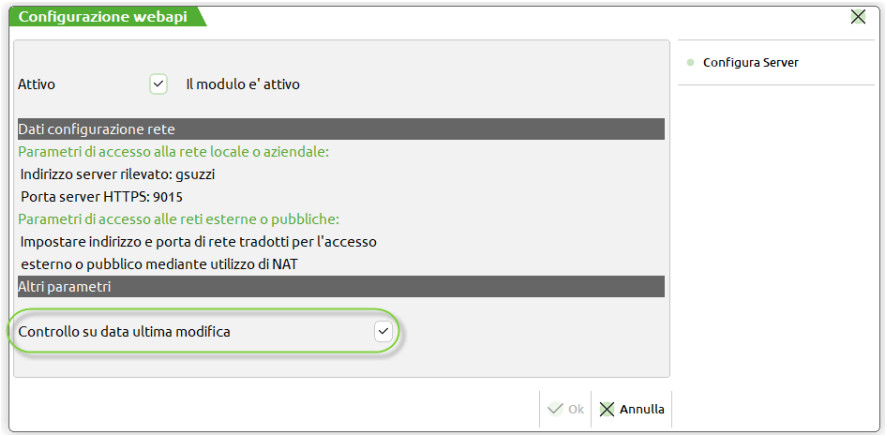
In fase di revisione di una risorsa (PUT o POST nel caso di trasformazione documenti) può capitare che i dati precedentemente letti da una GET non siano più validi perché modificati nel frattempo da altri attori. Per evitare che in fase di revisione si creino disallineamenti (es. leggo un documento con 3 righe, nel frattempo lato gestionale ne viene aggiunta una quarta, vado in revisione con le 3 righe lette in origine e di fatto cancello la quarta) è presente un controllo sulla data di ultima modifica.

È necessario specificare nel payload, oltre ai dati da modificare, anche le date di ultima modifica delle risorse che si stanno revisionando. Nel caso dei documenti, dato che un documento può derivare da più documenti di origine, il campo è di tipo array e deve essere specificata una data di ultima modifica per ogni testata presente.

Ricapitolando quindi, i campi per il controllo sulla data di ultima modifica saranno:

1. *dt_ult_mod_orig*: array di testate che deve essere utilizzato quando si sta effettuando una trasformazione documento con metodo POST
2. *data_ult_mod*: campo scalare che deve essere valorizzato in tutti gli altri casi

Il parametro è gestito lato gestionale alla voce di menu Servizi – Configurazioni – Configurazione moduli – WebAPI



3.7. ENTITÀ RICERCA

La risorsa *ricerca* è una particolare risorsa comune a tutti i gruppi di risorse (es. clienti, articoli, documenti, etc.).

Questa particolare risorsa può essere utilizzata solo mediante metodo POST. Il significato è: “crea una ricerca su questa collezione di risorse”. In questo modo è possibile ottenere un risultato come se fosse una GET in cui sono specificati filtri particolari nel *body* della *request* oltre quelli specificati in query string. Come per il comando GET è possibile utilizzare i filtri in query string (*fields, max e next*).

I filtri devono essere specificati secondo questo pattern

```
{
   "filtri":[
      {
         "campo":"<nome_campo>",
         "indice1":<indice1_campo_array>, // facoltativo: serve nel caso il campo sia un array. Nel caso di liste, gli array sono di
                                                massimo 2 livelli quindi possono essere specificati solo 2 indici
         "indice2":<indice2_campo_array>,
         "condizione": "=,<,<=,>,>=,<>,contiene,inzia_per",
         "case_insensitive": true/false      // default: false
         "valore": "<valore da testare>" oppure [<valori in logica OR tra loro>]
      },
     // altri elementi in AND
   ]
}
```

In tutti i casi in cui il confronto avviene su stringhe, può essere indicato se la ricerca debba essere case sensitive o meno indicando anche il parametro “*case_insensitive*” (default: false)

POST https://base_address/webapi/risorse/articoli/ricerca?fields=codice,descrizione

Esempio body della richiesta https

```
{
   "filtri":[
      {
         "campo":"tipologia",
         "condizione": "=",
         "valore": "A"
      },
      {
         "campo":"cos_ult",
         "condizione": ">",
         "valore": 100
      },
      {
         "campo":"descrizione",
         "condizione": "contiene",
         "valore": ["BICI","MTB","BDC"]
      }
```

```
  ]
}
```

I filtri sono tutti in logica AND tra loro. Nell'esempio verranno restituiti solo gli articoli di tipo merce **AND** costo ultimo anno corrente maggiore di 100 **AND** solo quelli con la descrizione che contiene le parole BICI **OR** MTB **OR** BDC. Del risultato saranno visualizzati solo i campi codice e descrizione.

3.8. ENTITÀ ALLEGATI

Solo per gli archivi che hanno allegati (es. immagini articoli) è disponibile una entità dedicata per l'accesso ai documenti.

GET https://base_address/webapi/risorse/<categoria_risorsa>/<codice_risorsa>/allegati/<tipo_allegato>

Dove tipo allegato è il tipo di file che si vuole referenziare. Nell'header della risposta sarà specificato il content-type corretto per il file restituito (ad esempio image/jpeg per le immagini).

Esempio di letture delle tre immagini articolo (icona, catalogo, immagine):

Lettura dell'icona di un articolo

GET https://base_address/webapi/risorse/articoli/FUELEX97/allegati/icona

Lettura dell'immagine di catalogo

GET https://base_address/webapi/risorse/articoli/FUELEX97/allegati/immagine-catalogo

Lettura dell'immagine articolo

GET https://base_address/webapi/risorse/articoli/FUELEX97/allegati/immagine

3.9. ENTITÀ MYDB

Per quanto riguarda l'accesso agli archivi MyDB sono disponibili tutti i metodi REST con la stessa logica delle altre entità, ma vale la pena entrare più nel dettaglio in quando si tratta di una risorsa particolare.

La risorsa mydb non è altro che un contenitore (un po' come succede con la risorsa documenti) e quindi non è direttamente accessibile: è necessario specificare sempre nel path anche l'archivio mydb sul quale si intende operare.

Il path sarà quindi di questo tipo

https://base_address/webapi/risorse/mydb/MIA_APP@mydb

dove MIA_APP è l'app Passbuilder in cui è presente l'archivio MyDB su cui voglio operare, e mydb il nome dell'archivio MyDB

Facendo una GET verrà restituita una lista di tutti i record dell'archivio MyDB indicato, mentre specificando l'id verrà restituito il singolo record

Lista record

GET https://base_address/webapi/risorse/mydb/MIA_APP@mydb

Lettura del record 42

GET https://base_address/webapi/risorse/mydb/MIA_APP@mydb/42

In entrambi i casi i dati restituiti avranno un formato di questo tipo (per le liste ovviamente saranno array di oggetti):

```
{
   "nome_archivio": "MIA_APP@mydb",
   "sigla_doc": "",
   "id": 42,
   "numero_campi": 5,
   "dati_campi": [],          // array contenente i valori dei campi dell'archivio MyDB
   "tipi_campi": [],          // array contenente il tipo dei campi
   "etichette_campi": [],     // array contenente le etichette dei campi MyDB
   "riportabil_doc": [],      // array contenente i flag sui campi riportabili
   "annullato": "",
   "bloccato": "",
   "utente_ult_mod": "198187508",
   "data_ult_mod": "20211217 113408"
}
```

Se si vuole interrogare un archivio MyDB collegato ad un documento va specificata anche la sigla documento.

Nel caso di richiesta della lista di tutti i record collegati ad un particolare tipo di documenti (es. OC) bisogna specificare la sigla documento come entità

GET https://base_address/webapi/risorse/mydb/MIA_APP@mydb/OC

Nel caso di singolo record l'id del record va specificato come sotto entità

GET https://base_address/webapi/risorse/mydb/MIA_APP@mydb/OC/42

I dati restituiti avranno valorizzato anche l'elemento "sigla_doc"

```
{
   "nome_archivio": "MIA_APP@mydb",
   "sigla_doc": "OC",
   "id": 42,
   "numero_campi": 2,
   "dati_campi": [],
   "tipi_campi": [],
   "etichette_campi": [],
   "riportabil_doc": [],
   "annullato": "",
   "bloccato": "",
   "utente_ult_mod": "626892910",
   "data_ult_mod": "20211217 122426"
}
```

I restanti metodi (POST,PUT,DELETE) seguono la solita logica.

Esempio di creazione di un nuovo record su un MYDB di un campo collegato ai documenti OC

POST https://base_address/webapi/risorse/mydb/MIA_APP@mydb/OC

Body della request:

```
{
   "id":0,
```

```
  "numero_campi": 2,
  "dati_campi": [
    [
      1,
      "C000OC001.000008"        // codice del documento (vedi manuale MYDB)
    ],
    [
      2,
      "valore_campo"
    ]
  ]
}
```

Nota: l'id del record deve essere presente e valorizzato a 0 dentro il body della request

3.10. ENTITÀ LISTE-PRELIEVO

L'entità liste di prelievo è una struttura master-detail, per cui è presente una testata ed n righe ad essa associata (come per i documenti). Sono quindi presenti end-point per accedere alla lista nel suo complesso ed altri per accedere solo alle righe:

- liste

  https://base_address/webapi/risorse/liste-prelievo

- righe

  https://base_address/webapi/risorse/liste-prelievo/righe

Come per i documenti, solo la singola lista può essere modificata (PUT) specificando tutte le righe a cui si vogliono apportare modifiche. Ogni riga d'ordine è identificata dalla chiave composta da sigla, serie, numero e id_riga.

Sempre come per i documenti, è possibile modificare solo i campi di testata specificando in query string il parametro *solo_testata=true*.

La peculiarità di questa entità risiede nel fatto che è possibile inserire un altro livello di dettaglio: le righe di picking. Per ogni riga della lista è infatti possibile "creare" ulteriori righe di prelievo (picking) ad essa associata.

Per ogni record (riga) della lista il campo "id_riga" identifica a quale riga d'ordine è associato. Se la riga è una riga di picking l'elemento "prog_riga" identifica il progressivo di picking relativo alla riga d'ordine, altrimenti se per la riga non è presente l'elemento "prog_riga" si intende che si tratta della riga d'ordine.

Nel caso si voglia inserire una nuova riga di picking, è sufficiente fare una PUT referenziando la riga d'ordine su cui si vuole agire ed impostando il progressivo di riga a -1. In questo caso è necessario specificare, oltre ai dati di chiave, solo le quantità.

NOTA: in caso di aggiornamento con PUT è necessario specificare sempre tutti i campi chiave (sigla, serie, numero, id_riga, prog_riga)

GESTIONE DEI LOTTI

Le variabili per l'identificazione dei lotti sulle righe sono le seguenti:

id_lotto: valorizzata solo per righe di picking (prog_riga>=1)

id_lotto_ord: valorizzata solo per righe ordine (prog_riga i-esima non presente)

pos_righe_lotto: posizione in cui iniziano i lotti della riga d'ordine i-esima nell'array dei lotti (id_lotto_ord)

nr_righe_lotto: numero di lotti gestiti dalla i-esima riga

dec_qta_lo_ord: decimali quantità lotto riga ordine

qta_lotto_ord: quantità lotto riga ordine

nr_colli_lo_ord: numero colli lotto della riga ordine

qta_tg_lo_ord: quantità taglia del lotto della riga ordine

Questi campi servono perché, sebbene tutte le righe siano omogenee, le righe d'ordine possono avere la gestione a multi lotto: è quindi necessario poter leggere tutti i lotti della riga (il meccanismo è identico a quello dei documenti: per ogni riga c'è l'indice per l'accesso all'array dei lotti + offset). Questo per quanto riguarda l'end-point liste-prelievo.

Per l'end-point liste-prelievo/**righe** invece il lotto è singolo e compare nell'array id_lotto con lo stesso indice della riga di picking. Per le righe d'ordine invece id_lotto=0 e sono valorizzati gli array id_lotto_ord.

ESEMPI DI REQUEST SU LISTE-PRELIEVO

Lettura delle testate di tutte le liste di prelievo

GET https://base_address/webapi/risorse/liste-prelievo

Lettura del dettaglio (testata+righe) di una lista di prelievo

GET https://base_address/webapi/risorse/liste-prelievo/5

Ricerca di una lista di prelievo (solo testate, filtri nel JSON)

POST https://base_address/webapi/risorse/liste-prelievo/ricerca

Modifica di una lista di prelievo (testate + righe)

PUT https://base_address/webapi/risorse/liste-prelievo/3

Con la PUT è possibile modificare anche i soli campi di testata omettendo le righe se viene specificato in query stringa il parametro solo_testata = true come avviene nei documenti

PUT https://base_address/webapi/risorse/liste-prelievo/3?solo_testata=true

Lettura di tutte le righe di tutte le liste di prelievo

GET https://base_address/webapi/risorse/liste-prelievo/righe

Ricerca di specifiche righe (impostare il filtro nel JSON)

POST https://base_address/webapi/risorse/liste-prelievo/righe/ricerca

Cancellazione di una lista (solo id lista) o singole righe (parametri di filtro in query string)

DELETE https://base_address/webapi/risorse/liste-prelievo/5?sigla=OC&serie=1&numero=339&id_riga=4&prog_riga=2

ESEMPIO DI INSERIMENTO DI UNA RIGA DI PICKING

Consideriamo la seguente GET sulla lista di prelievo 4 avente una sola riga d'ordine

GET https://base_address/webapi/risorse/liste-prelievo/4

```
{
   "id": 4,
```

```
  "data_creazione": "20220610",
  "stato": "C",
  "descrizione": "Lista Prelievo del 10/06/2022",
  "tipo": "V",
  "sigla": [
    [
      1,
      "OC"
    ]
  ],
  "sottoazienda": [],
  "serie": [
    [
      1,
      1
    ]
  ],
  "numero": [
    [
      1,
      330
    ]
  ],
  "id_riga": [
    [
      1,
      1
    ]
  ],
  "prog_riga": [],
  "codice_articolo": [
    [
      1,
      "ATT001"
    ]
  ],
  "data_documento": [
    [
      1,
      "20220302"
    ]
  ],
  "nr_riga_art": [
    [
      1,
      49
    ]
  ],
  "id_magazzino": [
    [
      1,
      1
    ]
  ],
  "id_ubicazione": [],
  "dt_sca_doc": [],
  "tp_um_articolo": [
    [
      1,
      "1"
    ]
  ],
  "um": [
```

```
      [
        1,
        "PZ"
      ]
    ],
    "nr_colli": [],
    "quantita": [
      [
        1,
        1.0
      ]
    ],
    "decimali_qta": [
      [
        1,
        2
      ]
    ],
    "coeff_conv": [],
    "id_lotto": [],
    "cod_taglia": [],
    "numero_taglia": [],
    "qta_taglia": [],
    "id_alias": [],
    "cod_conto": [
      [
        1,
        "501.00022"
      ]
    ],
    "cod_agente": [
      [
        1,
        "601.00006"
      ]
    ],
    "cod_vettore": [],
    "conto_ind_sped": [],
    "ind_spedizione": [],
    "nr_righe_lista": 0
}
```


Per aggiungere una riga di picking relativa alla riga d'ordine è necessario fare una PUT indicando, oltre alla chiave per identificare la riga (silga+serie+numero+id_riga+prog_riga), la quantità di picking (in questo caso "quantita", se l'articolo è gestito a colli o taglie è necessario utilizzare i campi dedicati) e impostando il progressivo di riga a **-1** ("prog_riga":[[1,-1]]


```
{
    "sigla": [
      [
        1,
        "OC"
      ]
    ],
    "serie": [
      [
        1,
        1
      ]
    ],
    "numero": [
      [
```

```
            1,
            330
        ]
    ],
    "id_riga": [
        [
            1,
            1
        ]
    ],
    "prog_riga": [
        [1,-1]
    ],
    "quantita": [
        [
            1,
            2
        ]
    ]
}
```


3.11. ENTITÀ UBICAZIONI

L'entità delle ubicazioni viene gestita in modo del tutto analogo alle altre risorse, ad eccezione di un particolare filtro che deve essere applicato in query string.

Per tutti i metodi (GET,POST,PUT,DELETE) l'entità *ubicazioni* è raggiungibile al seguente path

https://base_address/webapi/risorse/ubicazioni

Per la GET della singola entità, oltre a specificare l'id (obbligatorio), sono previsti anche i seguenti campi facoltativi da inserire in query string per la gestione dei progressivi delle ubicazioni: *cod_art_progr, dettlotti*

cod_art_progr:

questo parametro può assumere i seguenti valori:

- Nessun valore: vengono ritornati tutti i progressivi degli articoli in quella ubicazione
- Codice articolo: vengono ritornati tutti i progressivi dell'articolo specificato in quella ubicazione

Nel caso di caratteri speciali (compresi slash e backslash) devono essere passati con url-encoding

Se il parametro non è presente vengono ritornati solo i campi dell'anagrafica dell'ubicazione senza i progressivi.

dettlotti:

vincolato alla presenza di cod_art_progr, permette di escludere o meno i lotti utilizzando se valorizzato rispettivamente a *true* o *false*

Esempio di lettura di tutti i progressivi di una ubicazione

GET https://base_address/webapi/risorse/ubicazioni/76?cod_art_progr

Esempio di lettura dei progressivi di un articolo in una ubicazione

GET https://base_address/webapi/risorse/ubicazioni/76?cod_art_progr=FUELEX97

Esempio di lettura dei progressivi con anche i dettagli dei lotti

GET https://base_address/webapi/risorse/ubicazioni/76?cod_art_progr&dettlotti=true

3.12. ENTITÀ DOCUMENTI

L'end-point dei documenti (preventivi, ordini e documenti di magazzino) contiene entità complesse, composte generalmente da una o più testate ed *n* righe. La lettura della lista dei documenti ritorna una collection di sole testate, mentre la lettura delle righe di una particolare famiglia di documenti (es. documenti/movimenti-magazzino/righe) ritorna tutte le righe di quella famiglia che potranno poi essere filtrate per documento di origine tramite opportuna POST sull'end-pont di ricerca.

Nel caso invece si referenzi il singolo documento è possibile avere il dettaglio complessivo di testate e righe, e solo in questo caso è possibile fare modifiche al documento ed alle sue righe tramite metodi PUT e POST.

Per il singolo end-point (es. ordini-clienti/OC+1+1) le righe del documento sono rappresentate da variabili di tipo array. Per ogni variabile, l'indice dell'array oltre a referenziare la riga i-esima ne rappresenta la sua posizione a video in Mexal/Passcom.

Ad esempio se a video ho 2 righe, in prima posizione la riga con id=4 ed in seconda posizione la riga con id=1 (questo può capitare quando si modificano documenti, inserendo o eliminando righe), in WebAPI l'array degli *id_riga* sarà composto in questa maniera

```
{
   "id_riga": [
      [              // prima riga a video
         1,
         4
      ],
      [
         2,          // seconda riga a video
         1
      ]
   ]
}
```

L'ordine degli elementi può anche essere scambiato, l'importante è che gli indici rimangano coerenti.

Utilizzando lo stesso esempio potremmo referenziare le stesse righe cambiando l'ordine dei due elementi degli array ma mantenendo coerenza tra indice di riga e *id_riga*.

```
{
   "id_riga": [
      [              // seconda riga a video
         2,
         1
      ],
      [
         1,          // prima riga a video
         4
      ]
   ]
}
```

In caso di creazione o modifica di un ordine è importante che gli indici di riga mantengano l'ordinamento corretto altrimenti si modifica l'ordine in cui compaiono le righe sul documento in Mexal/Passcom. In generale non bisogna utilizzare l'id_riga come ordinamento.

In caso di revisione di un documento (PUT) bisogna sempre indicare tutti gli identificativi di riga (*id_riga*) ed il tipo di riga (*tp_riga*) anche se la riga n-esima non è da revisionare. Eventuali id omessi, di fatto elimineranno dal documento la riga corrispondente.

Nel caso si voglia modificare (PUT) solo la testata del documento è necessario passare in query string il parametro *solo_testata=true*

PUT https://base_address/webapi/risorse/documenti/ordini-cllienti/OC+1+1?solo_testata=true

In questo modo anche se gli *id_riga* vengono omessi, verranno sempre preservate le righe senza correre il rischio di eliminarle.

Nota sulla gestione delle taglie

Se il parametro di magazzino "Gestione a taglie" è impostato a No, le variabili della quantità a taglie nei diversi archivi è di tipo scalare e riporta sempre il valore 0. Es. quantità di riga documento di magazzino "qta_taglia": 0

Se il parametro di magazzino "Gestione a taglie" impostato a Si, le variabili della quantità a taglie nei diversi archivi è di tipo array. Nel caso l'articolo non gestisca le taglie viene esposta la variabile come array vuoto Es. quantità di riga documento di magazzino "qta_taglia": [], altrimenti è sempre di tipo array ma valorizzato.

Di seguito sono indicate alcune operazioni che possono essere svolte tramite WebAPI sui documenti, come ad esempio la copia di una riga o la trasformazione da un documento ad uno di ordine superiore.

COPIA RIGHE DOCUMENTI

Per tutti gli end-point raggruppati sotto l'entità *documenti* è possibile duplicare una riga specificando solo il suo riferimento. La copia di una riga è possibile solo nei seguenti casi:

- PUT (revisione di un documento)
- POST solo in trasformazione (sigla_trasformazione=xx)

La procedura si occuperà di prendere tutti i dati di riga dalla riga di origine. Gli unici parametri obbligatori sono *id_riga* (impostato a 0 perché si sta creando una nuova riga) e *tp_riga* in cui deve comunque essere specificato il tipo di riga che si andrà a copiare.

Per specificare la riga da cui copiare i dati dovrà essere indicato l'indice di riga di origine nel campo *idx_riga_copia* ed il corrispondente id_riga dovrà essere impostato a 0. In questo modo la procedura crea una nuova riga (*id_riga*=0) ma copiandola da una pre-esistente nel documento (*idx_riga_copia*=n).

Nel caso di copia durante la trasformazione di un documento composto da più testate, la riga da copiare deve essere sempre inserita nel punto di rottura tra una testata e la successiva (o comunque tra le righe della stessa testata). Questo perché, per un vincolo tecnico, i riferimenti alle testate (*id_rif_testata*) devono essere progressivi. Ad esempio, con 2 testate ognuna con 2 righe, se voglio inserire una riga copiandola da una riga della prima testata, non è possibile inserirla in ultima posizione, ma deve essere inserita in prima, seconda o terza: in questo modo ovviamente devono essere aggiornati tutti gli indici degli array per i riferimenti alle righe.

Esempio di aggiunta di una nuova riga in revisione di un documento (PUT)

```
{
   "sigla": "OC",
   "serie": 1,
   "numero": 358,
   "cod_conto": "501.00001",
   "data_documento": "20221231",
   "id_riga": [
      [
         1,
         1
      ],
```

```
      [
         2,
         2
      ],
      [
         3,          // aggiungo una nuova riga
         0
      ]
   ],
   "tp_riga": [
      [
         1,
         "R"
      ],
      [
         2,
         "R"
      ],
      [
         3,
         "R"         // bisogna comunque specificare il tipo della nuova riga
      ]
   ],
   "idx_riga_copia": [
      [
         3,          // indico che la riga 3 sarà una copia esatta della prima riga
         1
      ]
   ],
   "data_ult_mod": "20230227 145327"
}
```

Non è possibile fare una copia della copia, cioè impostare come idx_riga_copia un indice di una riga che è essa stessa copia di un'altra. La copia deve essere sempre riferita ad un riga pre-esistente nel documento.

TRASFORMAZIONE DOCUMENTI

Per tutte le risorse sotto l'entità documenti è possibile specificare in query string in fase di creazione (POST) l'eventuale sigla documento in cui si vuole trasformare il documento senza doversi preoccupare di gestire la trasformazione "a mano" eliminando il documento di origine o gestendo i residui delle righe. Eventuali moduli di stampa dovranno essere specificati nel campo dedicato *cod_modulo*.

Per le trasformazioni di documenti, se si vogliono gestire anche i residui di riga, è necessario valorizzare le seguenti variabili di riga (array) a seconda che la riga sia gestita a quantità, taglia, lotto, colli.

"nr_colli_trs": "Numero colli trasformato"
"quantita_trs": "Quantita' / Peso Lordo trasformato"
"qta_taglia_trs": "Quantita' per taglia trasformato"
"qta_lotto_trs": "Lotti - Quantita' trasformato"
"nr_colli_lo_trs": "Lotti - Colli trasformato"
"qta_tg_lo_trs":"Lotti - Quantita' per taglia trasformato"

"ubi_su_res":"S/N" indica se l'ubicazione deve essere impostata anche sul residuo (il default dipende da quello che è impostato nel documento di origine S se era già presente una ubicazione N se non era presente)

Le quantità a residuo devono essere intese (come in emissione/revisione documenti) come le quantità che verranno evase e la differenza andrà a residuo. Ad esempio se la prima riga ha quantità 10, se viene valorizzata

```
"quantita_trs":[
        [1,3]
]
```

si intende che, della prima riga con indice 1, vengono evase 3 quantità e 7 rimangono a residuo nel documento originale.

Dovranno sempre essere indicati tutti gli indici delle righe che si vogliono trasformare e, nel caso di gestione dei residui, le righe a residuo devono essere lasciate con lo stato originale (tipicamente “S”) mentre quelle che devono essere totalmente evase devono essere impostate in evadibili “E”.

In tutti i documenti da trasformare devono essere indicati anche i riferimenti al/ai documento/i di origine valorizzando i seguenti campi array:

sigla_doc_orig

serie_doc_orig

numero_doc_orig

data_doc_orig

sigla_ordine

serie_ordine

numero_ordine

data_ordine

per le trasformazioni su end-point degli ordini, preventivi o matrici i riferimenti agli ordini possono essere omessi mentre per i documenti dell’end-point *movimenti-magazzino* bisogna indicare anche i riferimenti agli ordini

Ad esempio se si vuole fare una fattura (FT) partendo da 2 bolle (BC) differenti è necessario valorizzare anche gli elementi "sigla_doc_orig”, "serie_doc_orig", ”numero_doc_orig", ”data_doc_orig” indicando per ogni riga il suo riferimento di origine come nell’esempio che segue

POST https://base_address/webapi/risorse/documenti/movimenti-magazzino?sigla_trasformazione=FT

```
{
   "sigla":"FT",
   "serie":1,
   "numero":0,
   "cod_conto":”501.00002”,
   "id_riga":[
      [
         1,
         1
      ],
      [
         2,
         1
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ],
      [
         2,
         "R"
      ]
   ],
```

```
  "data_documento": "20220523",
  "codice_articolo":[
    [
      1,
      "ATT001"
    ],
    [
      2,
      "ATT003"
    ]
  ],
  "quantita":[
    [
      1,
      3
    ],
    [
      2,
      4
    ]
  ],
  "cod_iva":[
    [
      1,
      "23"
    ],
    [
      2,
      "23"
    ]
  ],
  "sigla_doc_orig":[
      [1,"BC"],
      [2,"BC"]
  ],
  "serie_doc_orig":[
      [1,1],
      [2,1]
  ],
  "numero_doc_orig":[
      [1,2],
      [2,12]
  ],
  "id_rif_testata":[
      [1,1],
      [2,2]
  ],
  "data_doc_orig":[
    [1,"20220520"],
    [2,"20220523"]
  ],
  "dt_ult_mod_orig": [
        [1,“20230101 070555”],     // metadati letti dalle precedenti GET
        [2,“20230102 080512”]      // metadati letti dalle precedenti GET
  ]
}
```

Se nel body del POST è specificata una sigla differente viene ignorata, la query string sigla_trasformazione è prioritaria.

ESEMPIO1: TRASFORMAZIONE DA PIÙ ORDINI FORNITORE A BOLLA FORNITORE CON RESIDUO

Nel caso si voglia gestire un residuo e non “consumare” il documento di origine lasciandolo aperto per trasformazioni successive, è possibile specificare la quantità che deve essere trasferita nel documento trasformato (es. BF) e lasciare il residuo nei documenti di origine (es. OF).

Nell’esempio che segue viene fatta una trasformazione in bolla di carico (BF) partendo da due ordini fornitore (OF) con una riga ognuno rispettivamente con quantità 20 e 30. Nella bolla di carico verranno “spezzate” queste quantità in 5 e 6. Il risultato sarà quindi quello di avere una BF con 2 righe rispettivamente con quantità di 5 e 6 e le OF parzialmente evase verranno modificate con un residuo di 15 e 24.

POST https://base_address/webapi/risorse/documenti/movimenti-magazzino?sigla_trasformazione=BF

```
{
   "sigla":"BF",
   "serie":1,
   "numero":14,                          // il numero per le BF è obbligatorio non può essere automatico (0)
   "data_documento": "20220809",
   "cod_conto":"601.00001",
   "id_riga":[
      [
         1,
         1
      ],
      [
         2,
         1
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ],
      [
         2,
         "R"
      ]
   ],
   "tipo_stato_riga":[
      [                  // nel caso di gestone di un residuo lo stato deve rimanere quello del documento originale
         1,"S"
      ],
      [
         2,"S"
      ]
   ],
   "quantita":[  // quantita del documento di origine (facoltativo verranno prese in automatico dal documento di origine)
      [1,20],
      [2,30]
   ],
   "quantita_trs":[  // se voglio gestire i residui per ogni riga devo specificare la quantità che andrà a finire in BF
      [1,5],
      [2,6]
   ],
   "qta_lotto_trs":[
   ],
   "pos_righe_lotto": [
   ],
   "nr_righe_lotto": [
   ],
   "cod_iva":[
```

```
    [
        1,
        "22"
    ],
    [
        2,
        "22"
    ]
],
"sigla_doc_orig":[
    [1,"OF"],
    [2,"OF"]
],
"serie_doc_orig":[
   [1,1],
   [2,1]
],
"numero_doc_orig":[
    [1,337],
    [2,401]
],
"data_doc_orig":[
    [1,"20220809"],
    [2,"20220809"]
],
"id_rif_testata":[
    [1,1],
    [2,2]
],
"sigla_ordine": [
    [
        1,
        "OF"
    ],
    [
        2,
        "OF"
    ]
],
"serie_ordine": [
    [
        1,
        1
    ],
    [
        2,
        1
    ]
],
"numero_ordine": [
    [
        1,
        337
    ],
    [
        2,
        401
    ]
],
"data_ordine": [
    [
        1,
        "20220523"
```

```
      ],
      [
         2,
         "20220825"
      ]
   ],
   "dt_ult_mod_orig": [
         [1,“20230101 070555”],     // metadati letti dalle precedenti GET
         [2,“20230102 080512”]      // metadati letti dalle precedenti GET
   ]
}
```

ESEMPIO2: TRASFORMAZIONE DA OF A BF CON RESIDUO E GESTIONE DEI LOTTI

Può esistere la circostanza per cui un documento in origine non abbia lotti (si pensi ad un ordine fatto al fornitore) e poi solo in fase di trasformazione in bolla di carico vengano assegnati uno o più lotti.

Nell’esempio che segue si trasforma in BF partendo da due ordini al fornitore:

- OF337: ordine con articolo senza lotti
- OF401: ordine con articolo gestito a lotti

Nella BF risultante per la riga riferita all’ordine senza lotti, verrà gestito un residuo assegnando anche i lotti, mentre per la riga riferita all’ordine OF401 (con lotti) viene gestito un residuo riferito ai lotti già presenti.

POST https://base_address/webapi/risorse/documenti/movimenti-magazzino?sigla_trasformazione=BF

```
{
   "sigla":"BF",
   "serie":1,
   "numero":13,
   "data_documento": "20220809",
   "cod_conto":"601.00001",
   "id_riga":[
      [
         1,
         1
      ],
      [
         2,
         1
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ],
      [
         2,
         "R"
      ]
   ],
   "tipo_stato_riga":[
      [
         1,"S"
      ],
      [
```

```
        2,"S"
    ]
],
"quantita":[      // la quantità del documento di origine è facoltativa ( è stata specificata solo per scopi didattici, viene
    [1,20],       // recuperata automaticamente dai riferimenti dei doc di origine)
    [2,30]
],
"quantita_trs":[
    [1,5],        // per la prima riga metto in bolla solo 5 pezzi (15 andranno a residuo)
    [2,6]         // per la seconda riga metto in bolla solo 6 pezzi (24 andranno a residuo)
],
"qta_lotto_trs":[ // per ogni lotto già esistente indico le quantità per la gestione del residuo
    [4,2],         // NOTA BENE: se la riga di origine non aveva lotti non devo specificare nessun residuo
    [5,2],        // ma devo specificare direttamente la quantità totale del lotto (vedi campo qta_lotto)
    [6,2]         // in questo caso l'ordine OF337 non aveva lotti quindi ometto i riferimenti ai lotti
],
"pos_righe_lotto": [ // per ogni riga indico a quale posizione dell'array dei lotti (campo id_lotto) iniziano i propri lotti
    [1,1],        // per la riga 1 i lotti partono dall'indice 1
    [2,4]         // per la riga 2 i lotti partono dall'indice 4
],
"nr_righe_lotto": [ // per ogni riga indico quanti lotti gestisce: in questo caso 3 lotti per riga
    [1,3],
    [2,3]
],
"sigla_doc_orig":[
    [1,"OF"],
    [2,"OF"]
],
"serie_doc_orig":[
    [1,1]
    [2,1]
],
"numero_doc_orig":[
    [1,337],
    [2,401]
],
"data_doc_orig":[
    [1,"20220809"],
    [2,"20220809"]
],
"id_rif_testata":[
    [1,1],
    [2,2]
],
"sigla_ordine": [
    [
        1,
        "OF"
    ],
    [
        2,
        "OF"
    ]
],
"serie_ordine": [
    [
        1,
        1
    ],
    [
        2,
        1
    ]
```

```
  ],
  "numero_ordine": [
    [
      1,
      337
    ],
    [
      2,
      401
    ]
  ],
  "data_ordine": [
    [
      1,
      "20220523"
    ],
    [
      2,
      "20220825"
    ]
  ],
  "id_lotto":[ // array di tutti i lotti presenti nel documento
    [1,175],
    [2,174],
    [3,173],
    [4,173],
    [5,174],
    [6,175]
  ],
  "qta_lotto":[ // quantità dei lotti.
    [1,1],     // NOTA BENE: se la riga di origine non aveva lotti le quantità sono quelle totali che andranno a finire in bolla
    [2,2],     // e bisogna omettere la gestione del residuo (vedi campo qta_lotto_trs)
    [3,2],     // In questo caso i primi 3 elementi dell'array sono riferiti alla prima riga (che non aveva lotti) e quindi devo
    [4,15],    // specificare i totali che andranno in BF (in sostanza la somma deve essere uguale a quantita_trs (5) )
    [5,5],     // per i lotti riferiti alla seconda riga (che invece già aveva i lotti) devo indicare il totale dei lotti (la somma deve
    [6,10]     // essere uguale al campo quantita (30) )
  ],
  "dt_ult_mod_orig": [
        [1,"20230101 070555"],    // metadati letti dalle precedenti GET
        [2,"20230102 080512"]     // metadati letti dalle precedenti GET
  ]
}
```

Trasformazione indicando anche una ubicazione diversa rispetto al documento di origine

Nel caso di trasformazione con residuo si voglia specificare anche una ubicazione (es. ordine al fornitore (OF) di nuovi articoli che poi verranno ubicati solo in fase di ricezione merce (BF)), questa di default non viene impostata sul residuo ("ubi_su_res": "N").

Nel caso si voglia impostare l'id ubicazione anche nel documento di origine, oltre a valorizzare l'id nel campo "*id_ubicazione*" (si prenda come riferimento l'esempio precedente), è possibile specificare nel campo "*ubi_su_res*" se l'ubicazione impostata deve essere riportata anche sul residuo ("*ubi_su_res*":"S") oppure deve essere impostata esclusivamente sul documento trasformato ("*ubi_su_res*" : "N"). Il campo "*ubi_su_res*" non è un dato presente in archivio, ma è un parametro di servizio specifico nel caso di trasformazione documenti.

ESEMPIO3: TRASFORMAZIONE SPECIFICANDO UN PRELIEVO MERCE DA UBICAZIONI DIVERSE

Il caso è quello di un ordine cliente (OC) in cui non è specificata nessuna ubicazione per l'articolo (oppure è impostata una ubicazione di default) e si vuole prelevare parte della merce da una ubicazione e parte da un'altra.

Nell'esempio che segue si fa riferimento ad un OC con una riga gestita su 3 lotti differenti: 30 pezzi divisi in 15, 5 e 10 pezzi per ogni lotto rispettivamente. Non è specificata nessuna ubicazione. Si vuole trasformare il documento in una BC in cui si evadono parzialmente solo le quantità del primo lotto prelevando 3pz dall'ubicazione 2704 e 5pz dall'ubicazione 2708.

Creazione ordine OC1/401 con lotti

POST https://base_address/webapi/risorse/documenti/ordini-clienti

```
{
   "sigla":"OC",
   "serie":1,
   "numero":401,
   "data_documento": "20220825",
   "cod_conto":"501.00001",
   "id_riga":[
      [
         1,
         1
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ]
   ],
   "codice_articolo":[
      [
         1,
         "RUOCARB005"
      ]
   ],
   "quantita":[
      [
         1,
         30
      ]
   ],
   "pos_righe_lotto": [
      [1,1]          // indice da cui partire per referenziare i lotti della riga 1
   ],
   "nr_righe_lotto": [
      [1,3]          // numero di lotti gestiti dalla riga 1
   ],
   "id_lotto":[      // id dei tre lotti gestiti
      [1,173],
      [2,174],
      [3,175]
   ],
   "qta_lotto":[
      [1,15],        // quantità per ogni lotto
      [2,5],
      [3,10]
   ]
}
```

In questo caso, dato che l'ubicazione è una proprietà in rapporto 1:1 con la riga, se vogliamo assegnare più ubicazioni alla stessa merce è necessario che nel documento di destinazione si aggiunga una riga in più. Questa riga non può essere gestita dall'algoritmo di trasformazione che si basa esclusivamente sulla riga originale del documento OC. Deve quindi essere una riga copiata dalla riga 1 e completamente evasa (in modo che non risulti nell' OC originale) e, dato che la sua quantità viene presa dal lotto 173, deve essere aggiustata la quantità totale in modo che la trasformazione utilizzi dati coerenti.

POST https://base_address/webapi/risorse/documenti/movimenti-magazzino?sigla_trasformazione=BC

```
{
   "sigla": "BC",
   "serie": 1,
   "numero": 0,     // 0=numerazione automatica
   "data_documento": "20220809",
   "cod_conto": "501.00001",
   "id_riga": [
      [
         1,
         1
      ],
      [                 // nuova riga che dovrà comparire solo in bolla a cui sarà assegnata la seconda ubicazione di prelievo merce
         2,
         0
      ]
   ],
   "tp_riga": [
      [
         1,
         "R"
      ],
      [
         2,
         "R"
      ]
   ],
   "idx_riga_copia": [        // la nuova riga sarà una copia identica della prima
      [
         2,
         1
      ]
    ]
   "tipo_stato_riga": [       // la prima riga è quella originale e dato che deve essere gestito un residuo deve rimanere in stato S
      [                       // la seconda riga è quella aggiunta in BC e quindi deve essere completamente evasa (E) in modo
         1,                   // che non rimanga traccia sul documento di origine
         "S"
      ],
      [
         2,
         "E"
      ]
   ],
   "quantita": [              // la quantità totale di riga, che normalmente è quella del documento di origine, in questo caso deve
      [                       // essere abbassata coerentemente con quella della riga aggiunta (30-5). In questo modo la gestione
         1,                   // del residuo sulla prima riga opera con la quantità effettivamente disponibile.
         25
      ],
      [
         2,
         5
      ]
   ],
   "quantita_trs": [
      [                       // la quantità che deve andare nel documento trasformato (trs): in questo caso si prelevano solo 3pz
         1,                   // dalla prima riga
         3
      ]
   ],
```

```
"qta_lotto_trs": [          // dato che l'articolo è gestito a lotti devo indicare anche il/i lotti da cui prelevare la merce: la somma
  [                         // deve essere uguale alla quantita_trs. In questo caso viene evasa tutta la quantità solo dal primo
    1,                      // lotto
    3
  ]
],
"pos_righe_lotto": [
  [
    1,
    1
  ],
  [
    2,
    4
  ]
],
"nr_righe_lotto": [
  [
    1,
    3
  ],
  [
    2,                      // la nuova riga aggiunta deve avere anche il/i lotti da cui prelevare la merce. Anche in questo caso
    1                       // la merce è prelevata solo dal primo lotto, quindi di fatto è una riga con un solo lotto
  ]
],
"sigla_doc_orig": [         // il documento di origine è lo stesso per entrambe le righe
  [
    1,
    "OC"
  ]
],
"serie_doc_orig": [
  [
    1,
    1
  ]
],
"numero_doc_orig": [
  [
    1,
    401
  ]
],
"data_doc_orig": [
  [
    1,
    "20220809"
  ]
],
"id_riga_orig": [           // l'indice di riga di origine è il medesimo per entrambe le righe
  [
    1,
    1
  ],
  [
    2,
    1
  ]
],
"id_rif_testata": [         // entrambe le righe sono riferite alla stessa testata: il riferimento alla testata della riga copiata
  [                         // si può anche omettere perché la copia prenderà lo stesso riferimento di testata della riga origine
    1,1                     // ATTENZIONE però all'ordine: le righe copiate devono essere ordinate per testata di appartenenza
```

```
    ]                       // altrimenti verrà dato un errore in fase esecuzione della procedura
  ],
  "sigla_ordine": [
    [
      1,
      "OC"
    ]
  ],
  "serie_ordine": [
    [
      1, 1
    ]
  ],
  "numero_ordine": [
    [
      1,
      401
    ]
  ],
  "data_ordine": [
    [
      1,
      "20220523"
    ]
  ],
  "id_ubicazione": [        // per ogni riga specifico l'ubicazione di prelievo merce: 5 pz dall'ubicazione 2704 e 3 pz dalla 2708
    [
      1,
      2704
    ],
    [
      2,
      2708
    ]
  ],
  "ubi_su_res": [           // non voglio che l'ubicazione sia cambiata anche sul documento di origine quindi imposto "N"
    [
      1,
      "N"
    ],
    [
      2,
      "N"
    ]
  ],
  "id_lotto": [             // oltre ai lotti gestiti sulla riga originale bisogna aggiungere anche il/i lotti referenziati dalla riga
    [                       // aggiunta. In questo caso la riga evade solo quantità prese dal lotto 173 e quindi viene aggiunto
      1,                    // solo questo lotto
      173
    ],
    [
      2,
      174
    ],
    [
      3,
      175
    ],
    [
      4,
      173
    ]
  ],
```

```
  "qta_lotto": [              // la quantità totale dei lotti deve essere modifica in modo che sia coerente con quella della quantità
    [                         // totale. Dato che 5pz vengono "spostati" sul lotto 4 (173) a partire dal lotto 1 (173) bisogna
      1,                      // abbassare la quantità del lotto (15 - 5)
      10
    ],
    [
      2,
      5
    ],
    [
      3,
      10
    ],
    [
      4,
      5
    ]
  ],
  "dt_ult_mod_orig": [
        [1,"20230101 070555"]      // metadati letti dalle precedenti GET
  ]
}
```

3.13. ENTITÀ PRIMA NOTA

L'end-point è strutturato come quello dei documenti ed è suddiviso in testate e righe. Sono implementati tutti i metodi CRUD per la lettura e la scrittura.

I documenti di prima nota sono referenziabili usando una chiave composta dai seguenti campi:

- Data di registrazione (stringa)
- Numero progressivo di registrazione di primanota (1 – 16. 777. 214) (numerico)
- Causale del documento (stringa)
- Registro di protocollo (stringa)
- Serie di protocollo (numerico)
- Numero di protocollazione (numerico)
- Numero del documento (numerico)
- Data del documento (stringa)

Se il numero progressivo di registrazione è omesso (0) allora i restanti campi diventano obbligatori.

Esempio di inserimenti di un nuovo documento di prima nota (con i soli dati obbligatori):

POST webapi/risorse/prima-nota

```
{
  "data_registr": "20230302",
  "descrizione": "bella",
  "cau_contabile": "FE",
  "rstro_prot_iva": "V",
  "serie_prot": 1,
  "nr_protocollo": 5,
  "nr_documento": 0,
  "data_documento": "20201231",
  "codice_conto": [
    [
      1,
      "501.00001"
    ],
    [
```

```
            2,
            "501.00001"
        ]
    ],
    "importo_riga": [
        [
            1,
            6666
        ],
        [
            2,
            -6666
        ]
    ]
}
```

Esempio per GET/PUT/DELETE con solo il numero progressivo

GET/PUT/DELETE webapi/risorse/prima-nota/20210103+5022

Esempio per GET/PUT/DELETE senza specificare il numero progressivo

GET/PUT/DELETE webapi/risorse/prima-nota/20210103+0+FR+A+1+5+74344+20201234

La ricerca viene effettuata con metodo POST su end-point ricerca

POST webapi/risorse/prima-nota/ricerca

Come per l'end-point dei documenti è presente anche il dettaglio delle righe per le quali è possibile solo la lettura e la ricerca

GET webapi/risorse/prima-nota/righe

POST webapi/risorse/prima-nota/righe/ricerca

NOTA: nel caso di revisione di un documento (PUT), la procedura prima esegue una lettura, poi cancella il documento e lo riscrive con le modifiche. È quindi possibile che vengano restituiti degli errori sulla cancellazione in caso il documento non possa essere cancellato: 6001 - errore gestionale [documento senza record di scadenzario, doc. non cancellabile]

3.14. ENTITÀ REFERENTI

L'end-point *referenti* è diviso tra clienti e fornitori:

- risorse/referenti/clienti
- risorse/referenti/fornitori

Sono disponibili tutti i metodi CRUD per la creazione, lettura, aggiornamento, cancellazione e ricerca.

La particolarità di questo *end-point* sta nella richiesta della singola entità, non basta il codice cliente ma anche l'identificativo della referente ad esso associato. La chiave quindi si compone con <codice cliente>+<id>.

Es. get del referente 1 del cliente 501.00022

GET https://base_address/webapi/risorse/referenti/clienti/501.00022+1

Es. get del referente 3 del fornitore 601.00005

GET https://base_address/webapi/risorse/referenti/fornitori/601.00005+3

3.15. ENTITÀ ANAGRAFICA UNICA

Per l'end-point *anagrafica-unica* sono disponibili tutti i metodi CRUD.

Le anagrafiche sono identificate da un id univoco (*id_anagrafica*) o ricercate tramite codice fiscale (*anag_cod_fis* univoco per ogni anagrafica) dall'end-point *ricerca*.

Dato che si tratta di un archivio sovraziendale non è necessario specificare le Coordinate-Gestionale nell'header della request.

L'anagrafica unica prevede anche il concetto di storicizzazione: le diverse sezioni che compongono una anagrafica possono essere storicizzate ad una certa data di validità (data di storicizzazioni fino a cui è valido il particolare blocco di informazioni).

È quindi disponibile anche un end-point in cui sono presenti tutte le storicizzazioni accessibile solo con GET per la lista e POST per la ricerca: *anagrafica-unica/storico*

Ricapitolando quindi gli end-point disponibili sono 2:

anagrafica-unica: in cui sono presenti tutte le anagrafiche in linea (tutti i metodi CRUD)

anagrafica-unica/storico: in cui sono presenti tutte le storicizzazioni dei vari blocchi di informazioni (solo GET e POST per la ricerca)

LETTURA DI TUTTE LE ANAGRAFICHE IN LINEA

GET https://base_address/webapi/risorse/anagrafica-unica

LETTURA DI UNA ANAGRAFICA IN LINEA

Per leggere una singola anagrafica si deve fornire l'id mentre per leggere i dati storicizzati ad una tal data deve essere passata come chiave anche la data di validità.

Esempio di lettura con id

GET https://base_address/webapi/risorse/anagrafica-unica/6

Esempio di lettura per leggere un dato storicizzato alla *data_validita*

GET https://base_address/webapi/risorse/anagrafica-unica/6+20230601

Se la data validità, che è sempre opzionale, viene omessa viene utilizzata la data di apertura della sessione WebAPI che, se fatta nell'anno corrente, corrisponde alla data odierna (in questo caso le Coordinate-Gestionale devono essere presenti nell'header)

INSERIMENTO DI UNA NUOVA ANAGRAFICA

POST https://base_address/webapi/risorse/anagrafica-unica

Nel body è necessario valorizzare obbligatoriamente i seguenti campi:

- anag_cod_fis
- anag_rag_soc
- anag_piva

Nel tag Location viene restituita come al solito l'id per referenziare la risorsa appena creata

MODIFICA DI UNA ANAGRAFICA IN LINEA

La modifica, senza nessun parametro aggiuntivo, individua l'anagrafica esclusivamente tramite id e viene effettuata sull'ultima revisione.

PUT https://base_address/webapi/risorse/anagrafica-unica/6

Esempio di body per la modifica della ragiona sociale

```
{
   "anag_rag_soc":"Nuova Ragione Sociale"
}
```

Casi particolari:

- persona fisica:
  - anag_tipo_sogg: "M/F"

  in questo caso i campi *anag_cognome* e *anag_nome* vengono presi in considerazione
- società di capitali di persone/capitali o ente non commerciale:
  - anag_tipo_sogg: "P/C/E"

  in questo caso viene utilizzato il campo *anag_rag_soc* e si può valorizzare anche la struttura di variabili relative al Legale Rappresentante (*leg_*****): se è specificato l'id di una anagrafica esistente in *leg_id_anag* vengono letti direttamente i campi della anagrafica referenziata dall'id. Se *leg_id_anag=0* vengono utilizzati i valori impostati nei campi *leg_*. In modo analogo se è specificato un id per la Società Dichiarante nella variabile *leg_soc_id_anag* verranno utilizzati il codice fiscale e la ragione sociale dell'anagrafica referenziata dall'id, altrimenti verranno utilizzati i valori dei campi *leg_soc_cod_fiscale, leg_soc_rag_sociale*

MODIFICA DI UNA ANAGRAFICA CON STORICIZZAZIONE

Nel caso si voglia effettuare una modifica storicizzando l'anagrafica alla data della modifica, è possibile specificare in query string il parametro *storicizza*.

È possibile operare in due modalità:

1) inserire o modificare una storicizzazione esistente: *storicizza=S*

PUT https://base_address/webapi/risorse/anagrafica-unica/6?storicizza=S

2) storicizzare i dati in linea e contestualmente modificare i dati in linea: *storicizza=R*

PUT https://base_address/webapi/risorse/anagrafica-unica/6?storicizza=S

In entrambi i casi nel body del JSON devono essere specificate anche la date di validità del dato storicizzato e l'eventuale nota. I dati storicizzabili sono divisi per gruppi ed ogni gruppo ha tre campi *xxx_stor_data, xxx_stor_iniz, xxx_stor_nota.*

I gruppi storicizzabili sono:

3) i dati anagrafici di testata (anag_xxx)
4) i dati sulla residenza (res_xxx)
5) i dati sul domicilio (dom_xxx)

6) i dati sulla residenza estera (est_xxx)
7) i dati sul legale rappresentante (leg_xxx)
8) i dati sulla persona fisica (pf_xxx)
9) i dati sulla persona giuridica (pg_xxx)
10) i dati sulla nota anagrafica (nota)

1) Nel caso di modifica o inserimento nuova storicizzazione (*storicizza=S*) deve essere indicata la data di fine validità (**xxx_stor_data**)
2) Nel caso di revisione e storicizzazione (storicizza=R) deve essere indicata la data di inizio validità della nuova revisione (**xxx_stor_iniz**: la procedura calcolerà in automatico la data di storicizzazione)

Es. storicizzazione di una anagrafica modificando l'indirizzo di residenza

PUT https://base_address/webapi/risorse/anagrafica-unica/6?storicizza=S

```
{
   "res_indir":"Via Roma, 77",
   "res_stor_data":"20230810",
   "res_stor_nota":"vecchio indirizzo valido fino ad agosto"
}
```

Dove la data deve essere intesa come una data di validità: in questo caso si intende che l'indirizzo "via Roma, 77" era valido fino al 10 Agosto 2023.

Es. storicizzazione di una anagrafica con revisione dei dati in linea

PUT https://base_address/webapi/risorse/anagrafica-unica/6?storicizza=R

```
{
   "res_indir":"Via Roma, 78",
   "res_stor_iniz":"20230811",
   "res_stor_nota":"vecchio indirizzo valido fino ad agosto"
}
```

In questo caso si intende che l'indirizzo verrà "attualizzato" con "via Roma, 78" a partire dall'11/08/2023 e contestualmente l'indirizzo che c'era prima verrà storicizzato al 10/08/2023.

Nota: le date xxx_stro_iniz e xxx_stor_data sono mutuamente esclusive e se specificate entrambe verrà considerata solo la data inerente la particolare storicizzazione (S o R)

CANCELLAZIONE DI UNA ANAGRAFICA

La cancellazione individua l'anagrafica esclusivamente tramite id e una data di validità. Questo perché non è possibile cancellare l'intera anagrafica da WebAPI ma solo eventuali storicizzazioni.

DELETE https://base_address/webapi/risorse/anagrafica-unica/6+20230201

Nota: la DELETE elimina una intera storicizzazione alla data, quindi se più blocchi di informazioni hanno la stessa data di storicizzazione verranno eliminati tutti

RICERCA DI UNA ANGRAFICA IN LINEA

Anche per l'end-point anagrafica unica è disponibile l'entità *ricerca* su cui usare filtri tramite metodo POST (vedi capitolo dedicato)

LISTA DEI DATI STORICI

L'end-point *anagrafica-unica/storico* contiene tutte le storicizzazioni dei vari blocchi di informazioni delle varie anagrafiche.

GET https://base_address/webapi/risorse/anagrafica-unica/storico

Ogni elemento contiene le variazioni esclusivamente di un particolare blocco di informazioni. Dato che verranno in ogni caso riportati tutti i campi della struttura anagrafica, ma saranno valorizzati solo i campi del particolare blocco di informazioni, la data di storicizzazione sarà quella indicata nel campo *data_fine_valid*

Ogni record restituito dalla lista (e anche dalla ricerca) avrà un campo *tp_dati* che specifica quale tipo di dati è stato modificato e può assumere i seguenti valori:

- **A**: anag_xxx -> Dati anagrafici
- **R**: res_xxx -> Residenza anagrafica
- **I**: dom_xxx -> Domicilio fiscale
- **E**: est_xxx -> Residenza estera
- **L**: leg_xxx -> Legale rappresentante
- **T**: docum_xxx -> Documenti di identificazione
- **F**: pf_xxx -> Ulteriori dati persona fisica
- **G**: pg_xxx -> Ulteriori dati persona giuridica
- **N**: nota_xxx -> Nota anagrafica

.

Facendo una GET a questa data si avrà la fotografia dell'anagrafica completa a quel periodo.

RICERCA DEI DATI STORICI

Come di consueto facendo un POST su *anagrafica-unica/storico/ricerca* si potrà specificare un filtro di ricerca

POST https://base_address/webapi/risorse/anagrafica-unica/storico/ricerca

Questo può essere utile nel caso di dover ricostruire delle immagini storicizzate a seguito di modifiche.

Ad esempio può essere utile capire se una particolare anagrafica ha subito dei cambiamenti nei dati storici. In questo caso, specificando nel filtro l'id dell'anagrafica ed una data di ultima modifica, verrà tornata la lista di tutte le storicizzazioni che hanno subito dei cambiamenti.

```
{
  "filtri":[
            {
                "campo":"data_ult_mod",
                "condizione":">",
                "valore":"20230101 000000"
            },
            {
                "campo":"id_anagrafica",
                "condizione":"=",
                "valore":25
            }
```

```
        ]
}
```

Per avere la situazione della anagrafica aggiornata ad una delle date di storicizzazione ritornate, bisognerà poi fare una GET sull'end-point principale per ottenere l'anagrafica nel suo complesso.

GET https://base_address/webapi/risorse/anagrafica-unica/25+20230506

3.16. ENTITÀ AZIENDE

Tramite l'end-point *aziende* è possibile avere la lista di tutte le aziende presenti sull'installazione e di crearne una nuova.

Il comando, essendo sovra-aziendale, non necessiterà di specificare in header il campo Coordinate-Gestionale.

LISTA DELLE AZIENDE

È accessibile tramite metodo GET

GET https://base_address/webapi/risorse/aziende

INSERIMENTO DI UNA NUOVA AZIENDA

Tramite metodo POST sull'end-point aziende è possibile creare una nuova azienda collegandola ad un soggetto già presente in anagrafica unica. Questo vincolo quindi impone che prima della creazione dell'azienda, siano usate le API per la creazione del soggetto in anagrafica unica.

POST https://base_address/webapi/risorse/aziende

Esempio di body della request

```
{
   "in_sigla_azienda" : "MI2",
   "in_data_inizio_gestione" : "20230301",
   "in_gest_impresa_professionista" : "P",
   "in_gestione_fiscale" : "Y",
   "in_contabilita_cassa" : "V",
   "in_id_anagrafica" : 12,
   "in_mese_inizio_anno_contabile" : 1
}
```

Dove:

- "in_sigla_azienda": sigla azienda di tre caratteri alfanumerici
- "in_data_inizio_gestione": da di creazione dell'azienda nel formato AAAAMMGG
- "in_gest_impresa_professionista": tipo di gestione, può assumere i seguenti valori
    - P = professionista
    - I = impresa
- "in_gestione_fiscale" : tipo di gestione fiscale, può assumere i seguenti valori
    - O= ordinaria
    - S= semplificata
    - U = super minimo
    - I = solo attività istituzionale

  - Y= forfetaria 2015
- "in_contabilita_cassa": tipo di contabilità per cassa, può assumer i seguenti valori
  - S = reale
  - V = virtuale
- "in_id_anagrafica" : id dell'anagrafica unica a cui deve essere collegata l'azienda
- "in_mese_inizio_anno_contabile": mese in cui deve iniziare l'anno contabile

Prima della creazione dell'azienda l'anagrafica unica a cui deve essere collegata l'azienda deve già esistere.

Nota: il campo in_contabilita_cassa deve essere specificato esclusivamente se in_gest_impresa_professionista=I e in_gestione_fiscale=S. In tutti gli altri casi non viene accettato e la procedura darà un errore.

3.17. ENTITÀ TIPI LOTTI E MATRICOLE

L'end-point si trova in *dati-generali/tipi-lotti-matricole*.

La particolarità di questo end-point risiede del fato che la creazione può avvenire solo con codifica automatica specificando nel campo tipo_lotto la chiave "**".

Esempio di POST con campi personalizzati

POST https://base_address/dati-generali/tipi-lotti-matricole


```
{
    "tipo_lotto": "**",
    "stato": "C",
    "descrizione": "Test creazione",
    "chiave_univoca": "S",
    "auto_prel_tipo": "L",
    "auto_prel_qta": "S",
    "auto_prel_peso": "S",
    "auto_prel_spez": "S",
    "auto_prel_qres": "N",
    "tipo_ctrl_caric": "A",
    "auto_prel_gest": "E",
    "qta_insuf_scar": "D",
    "tipo_lott_matr": "M",
    "cod_tipo_lott": [
        [
            1,
            "**aa"
        ],
        [
            2,
            "**bb"
        ]
    ],
    "etichetta": [
        [
            1,
            "ANNO"
        ],
        [
            2,
            "CONTATORE"
        ]
    ],
    "tipo": [
        [
            1,
            "N"
        ],
        [
```

```
                2,
                "I"
            ]
        ],
        "dimensione": [
            [
                1,
                4
            ],
            [
                2,
                7
            ]
        ],
        "gest_obbl": [
            [
                1,
                "S"
            ],
            [
                2,
                "S"
            ]
        ],
        "riga_ord_visual": [
            [
                1,
                1
            ],
            [
                2,
                2
            ]
        ],
        "valore_iniziale": [
            [
                2,
                "1"
            ]
        ],
        "nr_elem_cod_ute": [
            [
                2,
                128
            ]
        ],
        "data_obbl": "N",
        "giorni_autocod": 0,
        "giorni_manuale": 0,
```

```
    "giorni_soglia": 0
}
```

Gli altri metodi (PUT,GET,DELETE) seguono la classica logica REST

3.18. WEBAPI PER LA RICHIESTA DI SERVIZI

In tutti i casi in cui l'operazione che si vuole effettuare sul gestionale non identifichi una risorsa ben precisa, ma piuttosto richieda l'esecuzione di una operazione o un insieme di operazioni che dovranno essere svolte da Mexal/Passcom, l'approccio RESTful non è più sufficiente e bisogna adottare un paradigma differente.

WebAPI mette a disposizione un ulteriore canale per la gestione di questi casi identificato con il nome *servizi.*

Le *http request* in questo caso saranno tutte di tipo *POST* e richiederanno l'esecuzione di un servizio.

Il path deve essere quindi formato in questo modo

POST https://base_address/webapi/servizi

Tutti i dati relativi all'esecuzione del particolare comando dovranno essere specificati nel *body* della *request*.

COLLAGE SERVER REMOTO

Questo comando permette l'esecuzione di un Collage Server Remoto contenuto in una Passapp esattamente come avviene con WebShaker (di fatto WebShaker è a tutti gli effetti una "costola" di WebAPI).

In questo caso è necessario specificare i dati necessari all'individuazione dell'etichetta Collage da eseguire all'interno della Passapp indicando *codice_app*, *nome_collage* e *etichetta_collage*

Esempio di esecuzione di un Collage Server Remoto

Richiesta

POST https://base_address/webapi/servizi

```
{
   "cmd": "esec_collage_server_remoto",
   "codice_app": "924148WEBSHAKER",
   "nome_collage": "colws",
   "etichetta_collage": "WAPI"
   "dati": {<eventuali dati da inviare>}
}
```

Risposta

```
{
        <risultato dell'elaborazione>
}
```

SVILUPPO DISTINTA BASE

Servizio per lo sviluppo della distinta base di un articolo

Nel caso di articolo non a taglie è necessario specificare la quantità come primo elemento dell'array "quantita_taglie".

Richiesta

POST https://base_address/webapi/servizi

```
{
```

```
  "cmd": "sviluppo_distinta_base",
  "dati": {
    "codice_articolo": "<stringa_codice_articolo>",
    "codice_art_padre": "<stringa_codice_art_padre>",
    "codice_cliente": "<stringa_codice_cliente>",
    "data_documento":<stringa_data_documento>,
    "codice":<numero_bolla>,
    "cod_sottobolla":<numero_sottobolla>,
    "nr_rif_pf":<numero_rif_prodotto_finito>,
    "id_riga_oc": <id_riga_ordine>,
    "quantita_taglie": [
          [1,n],
         ..
         [32,n]
       ]
  }
}
```

AVANZAMENTO DI PRODUZIONE

Servizio per avanzare la produzione creando automaticamente i relativi documenti di carico (CL) e scarico (SL)

Richiesta

POST https://base_address/webapi/servizi

```
{
   "cmd":"avanzamento_produzione",
   "dati":{
     "data_documento":"20220907",
     "codice": <numero_bolla>,
     "cod_sottobolla": <numero_sttobolla>,
     "nr_rif_pf": <numero_rif_prodotto_finito>,
     "tp_operazione": "S/C",  //Scarico o Carico
     "nr_fase": <numero_fase>,
     "id_lotto": [
     ],
     "qta_lotto": [
     ],
     "qta_agg_carico":1,
     "nr_colli_lotto": [],
     "qta_tg_lotto": [],
     "nr_decimali_qta": [
     ],
     "id_lotto_sa":[],
     "qta_lotto_sa":[],
     "nr_colli_lo_sa":[],
     "qta_tg_lo_sa":[],
     "nr_dev_qta_sa":[],
     "nr_righe_lo_sa":[],
     "nr_righe_lotto": 0
   }
}
```

SPEZZA RIGA BOLLE DI LAVORAZIONE

Funzione che permette di spezzare la quantità di riga del prodotto finito, suddividendo in due righe le quantità totali. È utilizzata per effettuare gli avanzamenti di produzione parziali.

Come per lo sviluppo della distinta base, nel caso di articolo non a taglie è necessario specificare la quantità come primo elemento dell'array "quantita_taglie".

Richiesta

POST https://base_address/webapi/servizi

```
{
   "cmd": "spezzariga_bl",
   "dati": {
      "codice": <numero_bolla>,
      "cod_sottobolla": <numero_sottobolla>,
      "codice_pf": "<codice_articolo>",
      "nr_rif_pf": <numero_rif_prodotto_finito>,
      "qta": [
         [
            1,
            2
         ]
      ],
      "stato_riga_orig": "<stato_riga_origine>",
      "stato_riga_dest": "<stato_riga_destinazione>"
      "solo_bl": "S/N"
   }
}
```

CONDIZIONI DOCUMENTO

Servizio che permette di ottenere, se presenti, prezzo sconto e provvigione in base ai parametri impostati.

POST https://base_address/webapi/servizi

```
{
   "cmd": "condizioni_documento",
   "dati": {
      "tipo": 1,                          // 1: prezzo, sconto e provvigione 2:sconto e provvigione 3:solo provvigione
      "cod_conto": "501.00085",           // conto del cliente
      "codice_articolo": "NIPDSWERG001",
      "data_documento": "20211231",
      "sigla_documento": "OC",
      "prezzo":0,                 // se impostato e tipo=1/2  allora viene utilizzato come base per calcolo sconti e provvigioni
      "quantita": 23,             // valore per lo scaglione delle quantità
      "coefficiente": 0,          // se zero lavora in u.m. primaria dell'articolo se valorizzata in u.m. secondaria
      "valuta" : 1,
      "id_listino" : 1,           // se zero si usa il listino del cliente
      "sconto" : "",              // se impostato e tipo=3 allora viene utilizzato come base per calcolo provvigioni
      "id_cat_sconto" : 0,        // se impostata viene utilizzata questa al posto di quella del cliente
      "id_categoria_pr" : 0,      // se impostata viene utilizzata questa al posto di quella del cliente
      "extra" : ""                // se impostato viene utilizzata questa per il calcolo
   }
}
```

Risposta

```
{
   "prezzo": 0.0,
   "sconto": "",
   "provvigione": 0.0,
   "tp_provvigione": [],       // tipo provvigione (% o T=totale)
   "formula_pr": [],           // formula provvigione
   "codice_agente": [],
   "quota_pr": [],             // quota provvigione ripartita
```

```
    "cond_agente": [],          // condizione agente
    "modalita_pr": [],          // modalità provvigione
    "base_impon_calc": [],      // base imponibile di calcolo provvigione
    "utente_ult_mod": "",
    "data_ult_mod": ""
}
```

CALCOLO ESPOSIZIONE

Servizio che permette, dato un codice conto e una data di riferimento, di calcolare i dati realitvi all'esposizione di un cliente/fornitore.

POST https://base_address/webapi/servizi

```
{
    "cmd":"calcolo_esposizione",
    "dati":{
        "in_codice_conto":"501.00022",
        "in_alla_data":"20230101"
    }
}
```

Risposta

```
{
    "saldo_cli_for": 51205.6,
    "valore_fido": 0.0,
    "val_ord_bolle": 170212.91,
    "val_ord_netto": 108925.91,
    "val_ord_iva": 132549.65,
    "val_bolle_netto": 61287.0,
    "val_bolle_iva": 74770.14,
    "esposizione": 0.0,
    "dt_esposizione": "20230101",
    "valore_rischio": 258525.39,
    "fuori_fido": 0.0
}
```

PROGRESSIVI DI TUTTE LE UBICAZIONI

Servizio che permette di ottenere i progressivi di tutte le ubicazioni per tutti i magazzini (Magazzino=0) o per il singolo magazzino selezionato (Magazzino=n)

```
{
    "cmd": "get_prog_ubicazioni",
    "cod_art_progr":"",
    "dettlotti":true
}
```

cod_art_progr e *dettlotti* hanno lo stesso significato della corrispondente chiamate REST per singola ubicazione:

- *cod_art_progr*: se presente vuoto ritorna il dettaglio di tutti gli articoli, se è specificato un articolo ritorna il dettaglio dell'articolo
- *dettlotti:* se presente e valorizzato a *true* torna il dettaglio anche dei lotti

Il magazzino deve essere impostato nell'header della *request* nella chiave *Coordinate-Gestionale*

Se la risposta presenta un elemento *next* significa che il servizio non è riuscito a fornire tutti i dati in una sola *request*. Bisogna fare una successiva *request* inserendo, oltre ai dati precedentemente indicati, anche il campo *next* esattamente come è indicato nella *response*.

Al momento è possibile filtrare il risultato esclusivamente per data di ultima modifica di un progressivo e per id. Il filtro deve quindi essere fatto usando il campo *dt_mod_ult_doc* e **non** il campo *data_ult_mod* (che invece si riferisce all'anagrafica dell'ubicazione). Sebbene il campo *dt_mod_ult_doc* sia un array, nel filtro il campo deve essere usato come scalare: il significato è "restituisci i soli record di ubicazione in cui almeno un progressivo soddisfa la condizione".

È possibile indicare i soli campi che si vuole vengano restituiti elencandoli nell'oggetto di tipo array "*campi*".

Di seguito un esempio di chiamata con filtro sulla data di ultima modifica

```
{
   "cmd": "get_prog_ubicazioni",
   "cod_art_progr":"",
   "dettlotti":true,
   "campi":[
      "id"
   ],
   "filtri":[
      {
         "campo":"dt_mod_ult_doc",
         "condizione":">",
         "valore":"20230501 162500"
      },
      {
         "campo":"id",
         "condizione":"=",
         "valore":4
      }

   ]
}
```

In questo esempio si chiede di avere tutti i progressivi con il dettaglio dei lotti, ma solo delle ubicazioni in cui è cambiato "qualche cosa" dopo il 1° maggio 2023 alle ore 16.25. Il risultato deve mostrare solo gli *id* delle ubicazioni coinvolte.

TOTALI DOCUMENTO

Servizio che permette di ottenere i totali del documento di magazzino specificato nei dati di input.

```
{
   "cmd": "totali_documento",
   "dati": {
      "in_sigla": "OC",
      "in_serie": 1,
      "in_numero": 400,
      "in_cod_conto": "501.00001"
   }
}
```

TOTALI DI RIGA

Servizio che permette di ottenere i totali di una riga di un documento

```
{
   "cmd": "totali_riga_documento",
   "dati": {
```

```
      "in_sigla": "OC",
      "in_serie": 1,
      "in_numero": 400,
      "in_id_riga": 1,
      "in_modo": 1, //
      "in_nr_dec_articolo":1 // numeo decimali dell'articolo
   }
}
```

in_modo indica la modalità di calcolo:

- 1 calcola per progressivi;
- 2 calcola per totale documento; la differenza di metodo è relazionata alle righe in um2 (unità di misura secondaria), ovvero, nel modo 1 si ottengono i valori (ad es. quantità e importo) di riga in unità di misura primaria e in valuta di gestione azienda, mentre invece, nel modo 2 gli stessi valori sono espressi in unità di misura secondaria (se gestita) ed in valuta del documento (se diversa da quella di valuta azienda).

ESPOSIZIONE DEL CLIENTE

Servizio che permette di ottenere i dati relativi all'esposizione di un cliente a partire da una particolare data

```
{
   "cmd":"calcolo_esposizione",
   "dati":{
      "in_codice_conto":"501.00031",
      "in_alla_data":"20230101"
   }
}
```

Esempio di risposta

```
{
   "saldo_cli_for": 5129.65,
   "valore_fido": 234.0,
   "val_ord_bolle": 23302500.03,
   "val_ord_netto": 12584.26,
   "val_ord_iva": 15355.99,
   "val_bolle_netto": 23289915.77,
   "val_bolle_iva": 28413697.21,
   "esposizione": 0.0,
   "dt_esposizione": "20230101",
   "valore_rischio": 28434182.85,
   "fuori_fido": 28433948.85,
   "rating_assegnato": "Incerto",
   "numero_insoluti": 0,
   "valore_insoluti": 0.0,
   "fatturato_net_ap": 2500.0,
   "fatturato_ld_ap": 2500.0,
   "fatturato_net_ac": 0.65,
   "fatturato_ld_ac": 0.65,
   "dt_doc_ult_gna": "20230828",
   "doc_ult_consegna": "BC  1/62    "
}
```

LISTA LOCALITÀ ITALIANE

Sono due servizi separati:

- uno per l'elenco delle province, località, cap e comune: **get_tabella_cap**
- uno per l'elenco degli indirizzi: **get_tabella_zone**

Get_tabella_cap

Accetta come parametri di ingresso:

- n_tipo_ricerca
- in_cap
- in_provincia
- in_comune
- in_localita

"in_tipo_ricerca" indica il tipo di informazione che si vuole richiedere:

- "P" : per ottenere il solo elenco delle province
- "C" : per ottenere l'elenco dei cap (ritorna sia la provincia che il codice cap). Inoltre può essere filtrato indicando la "provincia".
- "L" : per ottenere l'elenco delle località (ritorna provincia, cap e le località). Può essere filtrato indicando la "provincia" e/o il "cap".
- "M" : per ottenere l'elenco dei comuni (ritorna provincia, cap, localita e comune). Può essere filtrato indicando la "provincia e/o il "cap" e/o il "comune".

Esempio

```
{
   "cmd": "get_tabella_cap",
   "dati": {
      "in_tipo_ricerca": "L",
      "in_provincia": "RN",
      "in_comune":"Rimini",
      "in_cap":"47921",
      "in_localita":"Viserba"
   }
}
```

Per ottenere la lista di tutti i record senza nessun tipo di filtro è sufficiente fare la chiamata indicando solo il tipo = M

```
{
   "cmd": "get_tabella_cap",
   "dati": {
      "in_tipo_ricerca": "M"
  }
}
```

get_tabella_zone

Accetta come parametri:

- in_cap
- in_provincia

```
{
   "cmd": "get_tabella_zone",
   "dati": {
      "in_cap":"47921",
      "in_provincia": "RN"
   }
}
```

Nel caso si voglia la lista di tutte le zone senza nessun filtro è sufficiente invocare il comando senza filtri

```
{
   "cmd": "get_tabella_zone",
   "dati": {
   }
}
```

Entrambi i servizi, in quanto ritornano una grossa mole di dati, potrebbero paginare. In questo caso in fondo al json della risposta sarà presente un campo “next” con il valore del record successivo da cui partire nella successiva richiesta.

Il next dovrà essere inserito nella request allo stesso livello del campo *cmd*.

Esempio di next con le zone

```
{
   "cmd": "get_tabella_zone",
   "next": "89831",
   "dati": {
   }
}
```

CONTEGGIO NUMERO OPERAZIONI CONTABILI

Servizio che restituisce il conteggio del numero di operazioni contabili a fronte di una data di inizio e fine

```
{
   "cmd":"totali_nr_operazioni_contabili",
   "dati":{
      "in_da_data_op":"20230101",
      "in_a_data_op":"20230510"
   }
}
```

Se *in_da_data_op* e *in_a_data_op* sono vuote vengono restituite tutte le operazioni contabili

LETTURA PRATICA DICHIARATIVI

Servizio che restituisce le informazioni su una particolare pratica dei dichairativi

```
{
   "cmd": "leggi_pratica_dr",
   "dati": {
      "in_mod_pratica": "RPF",
      "in_pratica": "RMC",
      "in_interno": "1",
      "in_tp_soggetto": "D"
   }
}
```

LISTA PRATICHE DICHIARATIVI

Servizio che restituisce una collection di elementi con le informazioni su tutte le pratiche dei dichiarativi

```
{
   "cmd": "get_lista_pratiche_dr",
   "dati": {
   }
}
```

E' possibile filtrare le pratiche usando i seguenti filtri in ingresso in qualsiasi combinazione:

- **in_mod_pratica**: filtra le pratiche che appartengono al modello dichiarativo specificato.
- **in_da_pratica**: filtra le pratiche con numero pratica uguale o superiore a quella specificata.
- **in_da_interno**: filtra le pratiche con interno uguale o superiore a quello specificato (a parità di numero pratica)
- **in_a_pratica**: filtra le pratiche con numero pratica inferiore o uguale a quella specificata.
- **in_a_interno**: filtra le pratiche con interno inferiore o uguale a quello specificato (a parità di numero pratica)
- **in_cf_dichiarante**: filtra le pratiche relative al dichiarante con il codice fiscale indicato.
- **in_codice_stato_pratica**: serve per filtrare e ricercare le sole pratiche con lo stato specificato:
  - **N** - stato_pratica : (vuoto)
  - **T** - stato_pratica : Pratica terminata
  - **S** - stato_pratica : Pratica terminata - Solo stampa IMU
  - **I** - stato_pratica : Solo stampa IMU
  - **E** - stato_pratica : messaggio di errore relativo alla lettura della pratica
- **in_da_data**: filtra le pratiche che hanno la data ricevuta uguale o superiore a quella specificata.
- **in_a_data**: filtra le pratiche che hanno la data ricevuta inferiore o uguale a quella specificata.
- **in_codice_esito**: filtra le pratiche che hanno l'esito ricevuta specificato:
  - **N** - esito : (vuoto)
  - **A** - esito : Ricevuta accolta
  - **S** - esito : Ricevuta scartata
  - **X** - esito : Pratica inviata ma ricevuta non associata

Esempio di richiesta filtrata per stato pratica

```
{
  "cmd": "get_lista_pratiche_dr",
  "dati": {
    "in_mod_pratica": "RSC",
    "in_stato": "N"
  }
}
```

LISTA DELEGHE DICHIARATIVI

Servizio per leggere la lista delle deleghe dei pagamenti unificati F24. Le deleghe possono essere filtrate passando un elenco di pratiche o aziende, deleghe o un range di date etc., fino a d arrivare con precisione ad identificare una singola delega.

È possibile filtrare le deleghe usando i seguenti campi in qualsiasi combinazione:

- **in_azienda_pratica** : limita la ricerca alle sole aziende o alle sole pratiche. I valori accettati sono : **"A"** (sole aziende) o **"P"** (sole pratiche)
- **in_da_azienda** : filtra le deleghe che appartengono alla sigla azienda uguale o superiore a quella specificata
- **in_a_azienda** : filtra le deleghe che appartengono alla sigla azienda inferiore o uguale a quella specificata.
- **in_da_pratica** : filtra le deleghe che appartengono al numero pratica uguale o superiore a quella specificata.
- **in_a_pratica** : filtra le deleghe che appartengono al numero pratica inferiore o uguale a quella specificata.
- **in_da_interno** : filtra le deleghe che appartengono all'interno uguale o superiore a quella specificata (a parità di numero pratica)**{}**
- **in_a_interno** : filtra le deleghe che appartengono all'interno inferiore o uguale a quella specificata (a parità di numero pratica)
- **in_cf_dichiarante** : filtra le deleghe che appartengono alle pratiche relative al dichiarante con il codice fiscale indicato. (**solo pratiche, no aziende**)
- **in_mod_pratica** : filtra le deleghe che appartengono al modello dichiarativo specificato:
  - **730**
  - **RPF**
  - **RSP**
  - **RSC**
  - **RNC**
  - **CNM**
- **in_stato** : filtra le sole deleghe in stato "sospeso", se utilizzato accetta come valore al momento solo **"S"** (es.: "in_stato" : "S")
- **in_da_id_delega** : filtra le deleghe che hanno un ID uguale o superiore a quello specificato (formato : GG/MM/AAAA[/PR])
- **in_a_id_delega** : filtra le deleghe che hanno un ID inferiore o uguale a quello specificato (formato : GG/MM/AAAA[/PR]
- **in_da_data** : filtra le deleghe che hanno la data versamento uguale o superiore a quella specificata.
- **in_a_data** : filtra le deleghe che hanno la data versamento inferiore o uguale a quella specificata.
- **in_tipo** : filtra le deleghe nei tipi : "Ordinario", "Semplificato" o "Elide" (quest'ultimo ad oggi non è implementato, ma è stato previsto per un futuro)
- **in_definitiva** : filtra le deleghe "definitive". Specificando "S" tornano le sole deleghe definitive, con "N" tornano solo le deleghe "non definitive"
- **in_accise** : filtra le deleghe con "accise". Specificando "S" tornano le sole deleghe che contengono accise, con "N" tornano tutte le deleghe ad esclusione di quelle contenenti "accise".
- **in_ravvedimento_operoso** : filtra le deleghe di "ravvedimento". Specificando "S" tornano le sole deleghe di ravvedimento, con "N" tornano tutte le deleghe ad esclusione di quelle di ravvedimento operoso.
- **in_avviso_bonario** : filtra le deleghe relative ad avvisi bonari. Specificando "S" tornano le sole deleghe relative ad avvisi bonari, con "N" tornano tutte le deleghe ad esclusione di quelle che fanno riferimento ad avvisi bonari.

- **in_codice_invio** : filtra le deleghe in base alla modalità di invio della delega. I valori consentiti sono : S - CBI Studio, P - CBI Personale, E - Entratel intermediari, O - Contribuente Fisco Online, T - Contribuente Entratel, Y - Entratel Studio, N – Cartaceo
- **in_maggiorazione**: filtra le deleghe in base al fatto che sia presente (S) o meno (N) una maggiorazione
- **in_data_ult_mod**: filtra tutte le deleghe che sono state modificate a partire dalla data specificata in poi

Oltre ai filtri sono presenti altri due campi di tipo array in cui specificare puntualmente pratiche o aziende:

- **in_elenco_pratiche** : permette di specificare una o più pratiche dichiarativi nelle quali ricercare le deleghe
- **in_elenco_aziende** : permette di specificare una o più aziende nelle quali ricercare le deleghe contabili e/o dichiarativi in caso di azienda collegata a pratica redditi.

Nota: per le sole pratiche redditi è possibile ricercare delle deleghe su qualunque pratica presente nel gestionale senza dover necessariamente indicarne una lista tramite l'etichetta "in_elenco_pratiche".

Quando si opera sulle aziende, al contrario, è sempre necessario indicare l'elenco dove ricercare, quindi per operare sulle aziende, l'etichetta "in_elenco_aziende" deve sempre essere presente.

La ricerca può essere effettuata fino ad indicare in modo puntuale la singola delega specifica usando il seguente campo array:

in_elenco_deleghe: permette di specificare una o più deleghe da ricercare per le quali si vogliono tutte le informazioni contenute.

In caso di richieste onerose possono essere restituiti risultati parziali e viene fornita la chiave *next* da cui partire per completare la lettura (paginazione)

ESEMPI

Esempio1: filtro di tutte le deleghe presenti sull'installazione

```
{
   "cmd": "get_lista_deleghe"
}
```

Esempio2: filtro di tutte le deleghe presenti in una pratica specifica.

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_pratiche": [
         {
            "in_mod_pratica": "RPF",
            "in_pratica": "026",
            "in_interno": "1"

                  }
      ]

      }
}
```

Esempio3: filtro di tutte le deleghe presenti in due pratiche di diversa tipologia

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_pratiche": [
         {
            "in_mod_pratica": "RPF",
            "in_pratica": "026",
            "in_interno": "1"
         },
         {
```

```
            "in_mod_pratica" : "RSC",
            "in_pratica" : "CIM",
            "in_interno" : "1"

                }
    ]
    }

}
```

Esempio 4: filtro di tutte le deleghe presenti in due pratiche di diversa tipologia con data versamento compresa nell'intervallo specificato.

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_pratiche": [
         {
            "in_mod_pratica": "RPF",
            "in_pratica": "026",
            "in_interno": "1"
         },
         {
            "in_mod_pratica": "RSC",
            "in_pratica": "CIM",
            "in_interno": "1"
      ],
      "in_da_data": "20230101",
      "in_a_data": "20230630"
      }
}
```

Esempio 5: filtro di tutte le deleghe presenti in una pratica specifica che sono in stato "sospeso".

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_pratiche": [
         {
            "in_mod_pratica": "RPF",
            "in_pratica": "026",
            "in_interno": "1"
         }
      ],
      "in_stato" : "S"
      }
}
```

Esempio 6: filtro di tutte le deleghe presenti in un'azienda.

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
     "in_elenco_aziende": [
        {
          "in_sigla": "A11"
        }
    ]
  }
}
```

Esempio 7: filtro di tutte le deleghe presenti in 2 aziende.

```
{
```

```
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_aziende": [
         {
            "in_sigla": "A11"
         },
         {
            "in_sigla" : "A04"
          }
      ]
       }
}
```

Esempio 8: filtro di tutte le deleghe presenti in 2 aziende con data versamento più vecchia e fino al 30/06/2023.

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_aziende": [
         {
            "in_sigla": "A11"
         },
         {
            "in_sigla" : "A04"
         }
      ],
       "in_a_data" : "20230630"
      }
}
```

Esempio 9: filtro di due deleghe specifiche contenute all'interno di una pratica.

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_elenco_pratiche": [
         {
            "in_mod_pratica": "RPF",
            "in_pratica": "026",
                     "in_interno": "1",
            "in_elenco_deleghe" : [
              {
                 "in_id_delega" : "13/12/2023/15"
              },
              {
                 "in_id_delega": "16/02/2016/50"
              }
            ]
         }
      ]
   }
}
```

Esempio 10: filtro di tutte le deleghe di tutte le Aziende

```
{
   "cmd": "get_lista_deleghe",
   "dati": {
      "in_azienda_pratica": "A"
   }
}
```

Esempio 11a: filtro di tutte le deleghe DA "id delega" A "id delega" contenute nell'azienda "A11"

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_da_id_delega": "01/09/2023",
    "in_a_id_delega": "31/12/2023",
    "in_elenco_aziende": [
      {
        "in_sigla": "A11"
      }
    ]
  }
}
```

Esempio 11b: filtro di tutte le deleghe DA "id delega" A "id delega" contenute nell'azienda "A11"

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_da_azienda": "A11",
    "in_a_azienda": "A11",
    "in_da_id_delega": "01/09/2023",
    "in_a_id_delega": "31/12/2023"
  }
}
```

Esempio 12: filtro di tutte le deleghe DA id "01/08/2021" A id "31/12/2021 contenute nelle aziende dalla "A01" alla "A11" e contenute nelle pratiche da "RPF/026/1" a "RPF/B01/1

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_da_azienda": "A01",
    "in_a_azienda": "A11",
    "in_mod_pratica": "RPF",
    "in_da_pratica": "026",
    "in_da_interno": "1",
    "in_a_pratica": "B01",
    "in_a_interno": "1",
    "in_da_id_delega": "01/08/2021",
    "in_a_id_delega": "31/12/2021"
  }
}
```

Esempio 13: filtro di tutte le deleghe contenute nelle aziende dalla "A01" alla "A11" e tutte le deleghe contenute in tutte le pratiche

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_azienda_pratica": "P",
    "in_da_azienda": "A01",
    "in_a_azienda": "A11"
  }
}
```

Esempio 14: paginazione

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_azienda_pratica": "P"
  }
```

```
}
```

l'elaborazione si interrompe perchè è stato raggiunto uno dei limiti previsti. Quindi viene fornita la chiave per continuare con una nuova richiesta: "next": "P-730-001-1-D-2016-12-16-07-001-6". Per ottenere gli ulteriori dati dive essere fatta una richiesta forendo anche questo campo

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_azienda_pratica": "P"
  },
  "next": "P-730-001-1-D-2016-12-16-07-001-6"
}
```

Esempio 15: filtro di tutte le deleghe per data di ultima modifica. Vengono restituite tutte le deleghe a partuire dal 15 Aprile 2025 alle ore 9:00 in poi.

```
{
  "cmd": "get_lista_deleghe",
  "dati": {
    "in_data_ult_mod": "20250415 090000""
  }
}
```

SBLOCCO DELEGHE

Servizio che permette, dato un elenco di pratiche e deleghe, lo sblocco della delega che si trova in stato “sospeso”

Esempio di chiamata

```
{
  "cmd": "cambia_stato_delega",
  "dati": {
    "in_elenco_pratiche": [
      {
        "in_mod_pratica": "RPF",
        "in_pratica": "A02",
        "in_interno": "1",
        "in_elenco_deleghe": [
          {
            "in_id_delega": "02/01/2023/77",
            "in_numero_modulo": "001"
          }
        ]
      }
    ]
  }
}
```

Nell’array in_elenco_pratiche devono essere elencate le varie pratiche, e per ogni pratica nell’arrya in_elenco_deleghe devono essere lencati gli oggetti con i riferimenti per sbloccare la particolare delega (in_id_delega+id_numero_modulo)

In caso di errore verrà ritornato un codice di errore ed il realtivo errore:

"codice_esito"

0 - nessun errore

1 - delega in uso su altro terminale

2 - Delega non e' sospesa. Put non eseguita.

LETTURA REGISTRO CANCELLATI

Servizio per la lettura dei registri con l'elenco dei record eliminati sui vari end-point.

```
{
   "cmd": "get_registro_operazioni_utente",
   "dati": {
      "in_archivi": [
         <elenco_archivi>
      ],
      "in_tipo_operazione":"C",
      "in_storico":true,
      "in_data_ric_elab": "20240411 120000"
   }
}
```

Al momento è possibile richiedere solo i cancellati, quindi *in_tipo_operazione* può assumere solo C come valore.

In_storico, se impostato a true, viene indicato se il dato cancellato è presente nello storico o meno. L'informazione è disponibile solo per:

- Movimenti di magazzino (bolle)
- Ordini clienti
- Lotti su ordini clienti
- Lotti su movimenti di magazzino (lotti su bolle)

In_data_ric_elab è facoltativo (se omesso prende tutti i record cancellati per gli archivi specificati). Se viene invece inserita la *data_ric_elab* ritornata dall'ultima richiesta effettuata mostrerà solo i cancellati da quel momento in poi.

Il campo *in_archivi* è un array in cui dovranno essere specificati gli end-point dei quali si vuole avere la lista dei cancellati.

Attualmente gli end-point gestiti sono elencati di seguito:

- articoli
- alias-articoli
- clienti
- fornitori
- prima-nota
- movimenti-magazzino
- scadenzario
- anagrafica-unica
- anagrafica-unica-storico
- preventivi
- ordini-clienti
- indirizzi-spedizione
- liste-prelievo
- ubicazioni
- particolarita
- ordini-fornitori
- ordini-matrici
- lotti
- distinte-base
- lavorazione-bolle
- impegni

Esempio di richiesta senza specificare nessuna data

Richiesta

```
{
   "cmd": "get_registro_operazioni_utente",
   "dati": {
      "in_archivi": [
         "clienti",
         "articoli",
         "ordini-clienti",
         "movimenti-magazzino",
```

```
            "prima-nota",
            "anagrafica-unica"
        ],
        "in_tipo_operazione":"C"
    }
}
```

Risposta

```
{
    "data_ora_ultima_lettura": "20240411 121914",
    "registro_operazioni": [
        {
            "archivio": "clienti",
            "elenco_cancellati": []
        },
        {
            "archivio": "articoli",
            "elenco_cancellati": [
                {
                    "codice": "&"UN TEST\\CATTIVO",
                    "data_operazione": "20240410 121914"
                }
            ]
        },
        {
            "archivio": "ordini-clienti",
            "elenco_cancellati": []
        },
        {
            "archivio": "movimenti-magazzino",
            "elenco_cancellati": []
        },
        {
            "archivio": "prima-nota",
            "elenco_cancellati": []
        },
        {
            "archivio": "anagrafica-unica",
            "elenco_cancellati": []
        }
    ]
}
```

Questo indica che al momento della chiamata (20240411 121914) risulta cancellato un articolo.

Nota bene: la data non è quella in cui è stato eliminato il record ma quella in cui viene fatta la richiesta

Esempio di richiesta con data di ultima lettura
Richiesta

```
{
    "cmd": "get_registro_operazioni_utente",
    "dati": {
        "in_archivi": [
            "clienti",
            "articoli",
            "ordini-clienti",
            "movimenti-magazzino",
            "prima-nota",
            "anagrafica-unica"
        ],
        "in_tipo_operazione":"C",
        "in_data_ora_ultima_lettura": "20240411 121914"
    }
}
```

Risposta

```
{
    "data_ora_ultima_lettura": "20240411 122700",
    "registro_operazioni": [
```

```
      {
         "archivio": "clienti",
         "elenco_cancellati": []
      },
      {
         "archivio": "articoli",
         "elenco_cancellati": []
      },
      {
         "archivio": "ordini-clienti",
         "elenco_cancellati": []
      },
      {
         "archivio": "movimenti-magazzino",
         "elenco_cancellati": []
      },
      {
         "archivio": "prima-nota",
         "elenco_cancellati": []
      },
      {
         "archivio": "anagrafica-unica",
         "elenco_cancellati": []
      }
   ]
}
```

In questo caso passando come dato anche la data di ultima lettura ricevuta dal comando precedente non viene riportato alcun cancellato in quanto già processato dalla precedente richiesta.

Nota: in caso di ripristino di dati aziendali verrà restituito un errore 400 indicando che il registro non è più consistente.

LETTURA DICHIARAZIONI IVA ANNUALI

Servizio per richiedere la lista delle dichiarazioni iva.

```
{
   "cmd": "leggi_dichiarazione_iva_annuale",
   "dati":  {
         "in_da_azienda" : "A01",
         "in_a_azienda" : "A22",
         "in_da_data_versamento":"20240701",
         "in_a_data_versamento":"20240801"
   }
}
```

Dove *in_da_data_versamento* e *in_a_data_versamento* sono facoltative:

- se "in_da_data_versamento" non viene specificato, il servizio lo imposterà in automatico alla data 01/01/<primo anno gestito dell'azienda>
- se "in_a_data_versamento" non viene specificato, il servizio lo imposterà in automatico alla data 31/12/<ultimo anno gestito dell'azienda>

Nota bene: le aziende vengono aperte con anno terminale indicato nell'header della richiesta, mentre l'azienda indicata nell'header viene ignorata in quanto è un comando sovraziendale.

LETTURA COMUNICAZIONI LIPE

Servizio per richiedere la lettura delle dichiarazioni LIPE.

```
{
   "cmd": "leggi_comunicazioni_lipe",
   "dati":  {
         "in_da_azienda" : "A01",
         "in_a_azienda" : "A22",
         "in_da_data_versamento":"20240701",
         "in_a_data_versamento":"20240801",
         "in_trimestre":2
   }
```

```
}
```

Dove *in_da_data_versamento*, *in_a_data_versamento* e *in_trimestre* sono facoltativi:

- se "in_da_data_versamento" non viene specificato, il servizio lo imposterà in automatico alla data 01/01/<primo anno gestito dell'azienda>
- se "in_a_data_versamento" non viene specificato, il servizio lo imposterà in automatico alla data 31/12/<ultimo anno gestito dell'azienda>
- "in_trimestre" può assumere valori numerici (1,2,3,4,-1) e se viene omesso o impostato a -1 ritorna tutti i trimestri

Nota bene: verranno mostrate solo comunicazioni per cui è stata impostata una data di versamento.

Le aziende vengono aperte con anno terminale indicato nell'header della richiesta, mentre l'azienda indicata nell'header viene ignorata in quanto è un comando sovraziendale.

SERVIZI PER LA STAMPA PDF DI DOCUMENTI

I seguenti servizi lanciano una stampa PDF lato server del documento specificato nei campi chiave, ed il risultato viene tornato nella response con content-type *application/pdf*

Il campo *cmd* può assumere uno dei seguenti valori a seconda del tipo di documento di cui si vuole la stampa:

- stampa_ordine_cliente
- stampa_ordine_fornitore
- stampa_preventivo
- stampa_ordine_matrice
- stampa_movimento_magazzino

Nel campo dati devono essere specificati i campi chiave per identificare il documento:

- sigla
- serie
- numero
- cod_conto (obbligatorio solo per i movimenti di magazzino)
- cod_modulo (facoltativio se si vuole specificare un modulo di stampa personalizzato)

Esempi

Stampa di un ordine

```
{
   "cmd": "stampa_ordine_cliente",
   "dati":{
      "sigla":"OC",
      "serie":1,
      "numero":401,
      "cod_modulo":"A"

   }
}
```

Stampa di un movimento di magazzino

```
{
   "cmd": "stampa_movimento_magazzino",
   "dati":{
      "sigla":"BC",
      "serie":1,
      "numero":1,
      "cod_conto":"501.00001"
   }
}
```

SERVIZIO PER INSERIMENTO NUOVE RIGHE

Dato che l'inserimento di nuove righe a seguito di una PUT o anche in fase di trasformazione con POST non restituisce l'id delle righe appena create è stato inserito un servizio dedicato al solo inserimento di nuove righe all'interno di un documento.

Il servizio accetta i seguenti comandi, uno per ogni tipo di documento:

- ins_righe_ordine_cliente
- ins_righe_ordine_fornitore
- ins_righe_preventivo
- ins_righe_ordine_matrice
- ins_righe_movimento_magazzino

Per i documenti di tipo ordine e preventivo i campi chiave sono solo sigla, serie e numero.
Per i documenti di tipo movimenti di magazzino bisogna fornire anche il codice del conto (cod_conto)

Per i documenti di tipo ordine, oltre ai campi obbligatori per i dettagli di riga (es. codice, cod_iva etc…), va indicato anche l'id del magazzino per ogni riga (id_mag_riga).
Per i documenti di tipo movimento di magazzino, oltre ai campi obbligatori, va indicato anche l'id del magazzino di testata (id_magazzino).
Inoltre, per il funzionamento intrinseco di WebAPI, è necessario specificare anche i campi sui documenti di origine. Sono obbligatori nella richiesta ma ininfluenti per quanto riguarda il risultato dell'operazione quindi si possono lasciare vuoti:

"sigla_ordine": []
"serie_ordine": []
"numero_ordine": []
"sigla_doc_orig": []
"serie_doc_orig": []
"numero_doc_orig": []
"id_rif_testata": []

La posizione in cui inserire la nuova riga è specificata dall'indice dell'array. Ad esempio se si ha un documento con 3 righe e si volesse inserire 1 riga tra la seconda (quindi di fatto in posizione 3) è necessario specificare come indice 3.
Se si vogliono inserire più righe, la posizione di inserimento sarà sempre riferita al documento originale. Sempre considerando l'esempio con 3 righe, volendo inserire una riga tra la seconda e la terza (posizione 3) e un'altra in fondo (posizione 4) è necessario specificare come indici 3 e 4. Sarà poi la procedura a restituire gli indici e gli id risultanti dall'operazione: in questo caso indice 3 per la prima riga inserita ed indice 5 per la seconda.

Esempio:

Doc originale	Indice di riga origine	Doc con inserimento righe	Indice di riga risultante
Riga1	1	Riga1	1
Riga2	2	Riga2	2
		Nuova Riga3	3
Riga3	3	Riga3	4
		Nuova Riga4	5

Esempio di inserimento di tre righe in un ordine in posizione 2,3 e 6 in un documento in cui sono già presenti 5 righe:

```
{
   "cmd": "ins_righe_ordine_cliente",
   "dati": {
      "sigla": "OC",
      "serie": 1,
      "numero": 22,
      "id_riga": [
         [
            2,
            0
         ],
         [
            3,
            0
         ],
         [
            6,
            0
         ]

      ],
      "tp_riga": [
         [
            2,
            "R"
         ],
         [
            3,
            "R"
         ],
         [
            6,
            "R"
         ]
      ],
      "codice_articolo": [
```

```
        [
          2,
          "ATT002"
        ],
        [
          3,
          "ATT002"
        ],
        [
          6,
          "ATT002"
        ]
      ],
      "quantita": [
        [
          2,
          1.0
        ],
        [
          3,
          1.0
        ],
[
          6,
          1.0
        ]
      ],
      "cod_iva": [
        [
          2,
          " 22,0"
        ],
        [
          3,
          " 22,0"
        ],
        [
          6,
          " 22,0"
        ]
      ],
      "id_mag_riga": [
        [
          2,
          1
        ],
        [
          3,
          1
        ],
        [
          6,
          1
        ]
      ]
    }
}
```


Risposta


```
{
  "sigla": "OC",
  "serie": 1,
  "numero": 22,
  "id_riga": [
    [
      2,
      72
    ],
    [
      3,
      73
```

```
      ],
      [
         8,
         74
      ]
   ],
   "tp_riga": [
      [
         2,
         "R"
      ],
      [
         3,
         "R"
      ],
      [
         8,
         "R"
      ]
   ]
}
```

Le righe 2 e 3 sono state inserite in posizione corretta e quelle presenti nel documento di origine sono state “shiftate” per far posto ai nuovi inserimenti. La riga in posizione 6 si intende che deve essere inserita in ultima posizione, quindi nel documento risultante sarà inserita in posizione 8.

Esempio di inserimento di tre righe in un movimento di magazzino composto da testate differenti:

```
{
   "cmd": "ins_righe_movimento_magazzino",
   "dati": {
      "sigla": "BC",
      "serie": 3,
      "numero": 1,
      "cod_conto": "501.00072",
      "id_riga": [
         [
            3,
            0
         ],
         [
            6,
            0
         ],
         [
            7,
            0
         ]
      ],
      "tp_riga": [
         [
            3,
            "R"
         ],
         [
            6,
            "R"
         ],
         [
            7,
            "R"
         ]
      ],
      "codice_articolo": [
         [
            3,
            "ATT005"
         ],
         [
            6,
            "ATT005"
         ],
```

```
            [
                7,
                "ATT005"
            ]
        ],
        "quantita": [
            [
                3,
                1.0
            ],
            [
                6,
                1.0
            ],
            [
                7,
                1.0
            ]
        ],
        "cod_iva": [
            [
                3,
                " 22,0"
            ],
            [
                6,
                " 22,0"
            ],
            [
                7,
                " 22,0"
            ]
        ],
        "sigla_ordine": [],
        "serie_ordine": [],
        "numero_ordine": [],
        "sigla_doc_orig": [],
        "serie_doc_orig": [],
        "numero_doc_orig": [],
        "id_rif_testata": [],
        "id_magazzino": 1
    }
}
```


Risposta:


```
{
    "sigla": "BC",
    "serie": 3,
    "numero": 1,
    "cod_conto": "501.00072",
    "sigla_ordine": [],
    "serie_ordine": [],
    "numero_ordine": [],
    "sigla_doc_orig": [],
    "serie_doc_orig": [],
    "numero_doc_orig": [],
    "id_riga": [
        [
            3,
            3
        ],
        [
            7,
            4
        ],
        [
            8,
            5
        ]
    ],
    "tp_riga": [
```

```
        [
            3,
            "R"
        ],
        [
            7,
            "R"
        ],
        [
            8,
            "R"
        ]
    ],
    "id_rif_testata": []
}
```


Il documento di origine aveva 5 righe di cui le prime 2 righe appartenenti ad una testata e le rimanenti 3 ad un’altra.

La prima riga viene inserita in posizione 3 e prende come testata quella della riga precedente (quindi la prima testata), mentre le la 6 e la 7 vengono accodate in fondo e quindi vengono inserite nella seconda testata.

SERVIZO CON LA LISTA DI TUTTI GLI ALLEGATI DOCUVISION

Sevizio che permette di ottenere la lista di tutti gli allegati docuvision.

E' possibile filtrare i risultati tramite il l'elemento "filtri" e selezionare i soli campi che si vogliono leggere tramite l'elemento "campi" (come avviene per gli end-point REST).

Oltre ai filtri sui campi dell'archivio è possibile specificare nell'oggetto "*dati*" anche ulteriori parametri di filtro. Il parametro numerico "in_classe" deve essere sempre indicato ed in base alla classe si possono specificare opportuni filtri di selezione indicati in tabella1.

NOTA: i parametri di filtro sono gli stessi che, da gestionale in Docuvision->Gestione documento, compaiono nella maschera "Parametri su classe" (F2, scelta della classe e poi F7)

Il parametro "*in_allegati*" è opzionale (default T) e ha i seguenti significati:

- S = Sì: vengono selezionati solo i documenti allegati ad uno o più archivi del gestionale rappresentati dalle classi predefinite.
- N = No: vengono selezionati solo i documenti non allegati.
- T = Tutti: vengono selezionati i documenti indipendentemente se allegati o meno ad archivi rappresentati da classi predefinite.

Esempio: lista di tutti gli allegati degli articoli che nel codice contengono la parola TEST ed il risultato ha nr_protocollo>=3. Viene tornata solo la descrizione.

```
{
   "cmd": "lista_docdv",
  "dati":{
      "in_classe":1200,
      "in_arti_cod":,"TEST"
      "in_arti_op_cod":"CONTIENE"
   },

   "campi":[
      "descrizione"
   ],
   "filtri":[
      {
         "campo":"nr_protocollo",
         "condizione":">=",
         "valore":"3"
      }
   ]
}
```

Tabella 1: parametri di filtro su parametri classe

Categoria	Tipo	Nome Variabile
Generali	NUM	in_classe
	STR	in_allegati
1 - Stampa generica	STR	in_stpg_menu
	STR	in_stpg_op_menu
	STR	in_stpg_nota
	STR	in_stpg_op_nota
100 - Anagrafica nominativi	STR	in_nomi_tp_allega
	DAT	in_nomi_dt_da
	DAT	in_nomi_dt_a
200 - Bilanci	NUM	in_bila_anno
	STR	in_bila_tipo
	STR	in_bila_autore
	STR	in_bila_op_autore
400 - Anagrafica Cliente/Fornitore	STR	in_pico_tp_conto
	STR	in_pico_cod_conto

	STR	in_pico_cod_alt
	STR	in_pico_op_cod_alt
	STR	in_pico_nome_ric
	STR	in_pico_op_nome_ric
	STR	in_pico_prv
	NUM	in_pico_zona
	NUM	in_pico_catsta
600 - Operazione contabile	NUM	in_prn_sotto_az
	DAT	in_prn_dt_regis
	STR	in_prn_cau
	STR	in_prn_desc_testa
	STR	in_prn_op_desc_testa
	STR	in_prn_tp_rstro
	NUM	in_prn_serie_prot
	NUM	in_prn_nr_prot
	NUM	in_prn_nr_doc
	DAT	in_prn_dt_doc
	STR	in_prn_tp_invio
800 - Dichiarativi/Versamenti	NUM	in_dicv_anno
	STR	in_dicv_dich
	STR	in_dicv_tp_doc
	STR	in_dicv_sgl_prat
	STR	in_dicv_interno
	STR	in_dicv_dich_coniuge
	STR	in_dicv_tp_allega
900 - Stampe Fiscali	NUM	in_stpf_anno
	STR	in_stpf_doc
	STR	in_stpf_tp_doc
	STR	in_stpf_op_tp_doc
	STR	in_stpf_crit_sel
	STR	in_stpf_op_crit_sel
1000 - Agenda di lavoro	STR	in_agen_stato
	DAT	in_agen_dt_lavoro
	STR	in_agen_cod_conto
	STR	in_agen_prest
	STR	in_agen_op_prest
1100 - Parcella/Notula	NUM	in_notp_sotto_az
	STR	in_notp_tp_doc
	NUM	in_notp_serie
	NUM	in_notp_numero

	DAT	in_notp_dt_doc
	STR	in_notp_cod_conto
	STR	in_notp_note
	STR	in_notp_op_note
1200 - Anagrafica articolo	STR	in_arti_cod
	STR	in_arti_op_cod
	STR	in_arti_desc
	STR	in_arti_op_desc
	STR	in_arti_cod_alt
	STR	in_arti_op_cod_alt
	STR	in_arti_catsta
	STR	in_arti_tp_allega
1400 - Movimento di magazzino	NUM	in_mm_sotto_az
	STR	in_mm_sigla
	NUM	in_mm_serie
	NUM	in_mm_numero
	DAT	in_mm_dt_doc
	NUM	in_mm_anno_solare
	STR	in_mm_cod_conto
	NUM	in_mm_zona
	NUM	in_mm_catsta
	STR	in_mm_note
	STR	in_mm_op_note
1450 - Controllo di gestione	STR	in_cdg_tp_anag
Analitici	STR	in_cdg_a_cod
	STR	in_cdg_a_tp
	STR	in_cdg_a_natura
	STR	in_cdg_a_qualita
Commesse	STR	in_cdg_c_cod
	STR	in_cdg_c_padre
	STR	in_cdg_c_tp
	STR	in_cdg_c_respons
	STR	in_cdg_c_zona
	NUM	in_cdg_c_cli
	NUM	in_cdg_c_dest_merce
	NUM	in_cdg_c_age
Aree	STR	in_cdg_r_cod
	STR	in_cdg_r_padre
	STR	in_cdg_r_tp
	STR	in_cdg_r_respons
	STR	in_cdg_r_zona

1500 - Ordine/Preventivo/Matrice	NUM	in_ordi_sotto_az
	STR	in_ordi_sigla
	NUM	in_ordi_serie
	NUM	in_ordi_numero
	DAT	in_ordi_dt_doc
	STR	in_ordi_cod_conto
	STR	in_ordi_note
	STR	in_ordi_op_note
1600 - Bolla di lavorazione	NUM	in_blav_cod
	STR	in_blav_cod_sottobl
	DAT	in_blav_dt_bolla
	STR	in_blav_cod_conto
	STR	in_blav_nota
	STR	in_blav_op_nota

SERVIZIO PER LA LETTURA DEGLI ALLEGATI DOCUVISION

Tramite questo servizio è possibile, per ogni archivio del gestionale, leggere i documenti allegati.

Gli allegati verranno restituiti in un *multipart/form-data* in formato RAW ognuno evidenziato con il proprio Content-Type (es. application/pdf, image/jpg, etc.) come da RFC

```
{
   "cmd": "get_allegato_archivio",
   "dati": {
      "tipo":"docuvision",
      "codice":"",
      "classe_dv":400,
      "id_dv":1,
      "numero_versione_dv":3,
   }
}
```

Dove:

- *tipo* è il tipo di allegato richiesto (al momento è possibile specificare solo docuvision)
- *codice* è il codice dell'entità di cui si vogliono leggere gli allegati (non deve essere valorizzato se id_dv>0)
- *classe_dv* è la classe del documento docuvision
- *id_dv indica l'id del particolare documento (0 oppure omesso se codice è valorizzato)*
- *numero_versione_dv (facoltativo) indica la versione del documento:*
  - *-1 - tutte le versioni*
  - *0 - ultima versione*
  - *>0 - versione specifica*

I campi *codice* e *id_dv* sono mutuamente esclusivi

Nel caso *numero_versione_dv>0* è obbligatorio specificare anche *id_dv*

Tabella classi docuvision

classe_dv	Descrizione
1	Stampa generica
100	Anagrafica nominativi
200	Bilanci
400	Anagrafica Cliente/Fornitore
600	Operazione contabile

800	Dichiarativi/Versamenti
900	Stampe Fiscali
1000	Agenda di lavoro
1100	Parcella/Notula
1200	Anagrafica articolo
1400	Movimento di magazzino
1450	Controllo di gestione
1500	Ordine/Preventivo/Matrice
1600	Bolla di lavorazione
1700	Anagrafiche MyDB

In caso che il particolare archivio contenga più documenti (es. un articolo con diversi datasheet) oppure lo stesso documento ha più revisioni, se vengono superati limiti di tempo (30s) o di spazio (50MB) il risultato verrà paginato restituendo come ultimo oggetto del *multipart* solo l'elemento *next*

```
--ac70wbp48zghcooa4lol2n2xzuqfohrt
Content-Type: text/plain
Content-Disposition: form-data; name="__next"

9;3
--ac70wbp48zghcooa4lol2n2xzuqfohrt—
```

In questo esempio si intende che il documento continua con altri file a partire da *next="9;3"*

La successiva chiamata va effettuata specificando *next* allo stesso livello di *cmd*

```
{
   "cmd": "get_allegato_archivio",
   "next": "9;3",
   "dati": {
      "tipo":"docuvision",
      "codice":"",
      "classe_dv":400,
      "id_dv":1,
      "numero_versione_dv":3,
   }
}
```

SERVIZIO CON LA LSITA DELLE STRUTTURE MYDB

Tramite questo servizio è possibile leggere e filtrare tutta la lista delle strutture MyDB presenti sui vari archivi.

```
{
   "cmd": "lista_strutture_mydb",
   "info_campi":true,
   "campi": [
      " lista_strutture_mydb:tipo_archivio"
   ],
   "filtri": [
      {
         "campo":" lista_strutture_mydb:codice_app",
         "condizione":"inizia_per",
         "valore":"924148"
      }

   ]
}
```

Nell'elemento *campi* possono essere indicati gli unici campi che si vogliono ottenere in risposta (nell'esempio *tipo_archivio*)

Nell'elemento *filtri* possono essere specificati dei filtri con le stesse condizioni usate per gli end-point REST

Se *info_campi=true* viene mostrato l'help in linea di tutti i campi che compongono la struttura

Il risultato è un json di questo tipo

```
{
   "lista_strutture_mydb": [
      {
         "tipo_archivio": "A",
         "codice_app": "010999fatturapa",
         "codice_archivio": "ANAPADOA",
         "desc_archivio": "Fattura XML (Codice Articolo)",
         "ordinamento": "I",
         "chiave_univoca": "N",
         "estensione_univoca": "S",
         "tipo_estensione": "R",
         "elenco_campi": [
            {
               "codice_campo": "zzz",
               "campo_estensione": "S",
               "etichetta_campo": "Tutti gli articoli",
               "tipo_campo": "*",
               "dimensione_campo": 32,
               "gest_campo_obblig": "S",
               "valore_dft_campo": "",
               "archivio_riportabile": "DMRT",
               "cod_produttore_rel": "",
               "codice_archivio_rel": "ARTT",
               "codice_campo_rel": ""
            }
…
}
```

Per indicare i campi negli elementi campi o filtri bisogna specificare il percorso completo del campo di interesse all'interno della struttura json usando i ":" come separatore. Ad esempio se volessimo usare il campo codice_campo all'interno della collection elenco_campi è necessario usare la seguente sintassi

lista_strutture_mydb:elenco_campi:codice_campo

SERVIZIO PER LA LETTURA DEI PARAMETRI DEI NUMERATORI

Tramite questo servizio è possibile ottenere, per ogni tipo documento, la propria gestione dei numeratori.

```
{
        "cmd": "get_proprieta_numeratori",
        "dati": {
                "in_sigla": "FF",
                "in_cod_fornitore":"601.00001",
                "in_tipo_documento_el":"TD01"
        }
}
```

dove:

- in_sigla: è la sigla del documento di cui si vogliono le informazioni
- in_cod_fornitore: codice del fornitore necessario esclusivamente nel caso di documenti di tipo FF
- in_tipo_documento_el: tipo del documento elettronico necessario esclusivamente nel caso di documenti di tipo FF

Il risultato riporta il numero massimo di serie gestite (max_serie), il valore massimo del numeratore (max_numero_documento) e se gestisce o meno la numerazione automatica (gest_numerazione_automatica)

```
{
        "max_serie": 1,
        "max_numero_documento": 999999,
        "gest_numerazione_automatica": "N"
}
```

SERVIZIO PER L'AUTENTICAZIONE DI CAMPI MYDB

Per i soli campi degli archivi MyDB etichettati come "Aut" è possibile gestire l'autenticazione tramite il servizio *verifica_autenticazione_mydb*.

Il servizio funziona esclusivamente per coppie di credenziali utente+password, quindi se un archivio mydb contiene più di 2 campi etichettati come "Aut" nel servizio è necessario specificare quali sono i campi sui quali si vuole verificare l'autenticazione.

E' possibile anche specificare il record specifico su cui fare il controllo: se viene omesso, il controllo viene fatto su tutti i record e restituisce esito positivo alla prima coppia di credenziali verificata.

```
{
	"cmd":"verifica_autenticazione_mydb",
	"dati":{
		"in_codice_archivio":"924148CQWEBAPI@mydb0001",
		"in_nomecampo_login":"user",
		"in_login_value":"utente1",
		"in_nomecampo_password":"pwd",
		"in_password_value":"pwd1",
		"in_id":3
	}
}
```

- **in_codice_archivio** : archivio MyDB contenente i campi di autenticazione
- **in_nomecampo_login** : campo dell'archivio MyDB designato come campo "username"
- **in_login_value** : username necessario per l'autenticazione
- **in_nomecampo_password** : campo dell'archivio MyDB designato come campo "password"
- **in_password_value** : password associata allo username necessaria per verificare l'autenticazione
- **in_id** : (facoltativo) id del record dell'archivio MyDB per il quale si deve verificare l'autenticazione

SERVIZIO PER LA LETTURA DEI CALENDARI DI PRODUZIONE

Il servizio, disponibile solo per aziende di livello produzione o superiore, restituisce la lista di tutti i calendari configurati su Mexal in Produzione > Tabelle > Calendari produzione (Alt+PKA)

```
{
	"cmd":"get_calendari_produzione"
}
```

SERVIZIO PER LA LETTURA MULTILOTTO

Questo servizio, affiancato agli end-point REST sui lotti, permette di leggere tutti i lotti specificando ulteriori parametri come filtro.

```
{
        "cmd": "get_multilotto",
        "ati": {
                "in_tipo_lotto":"<tipo>",
                "in_cod_articolo":"<codice>",
                "in_tp_cod_utente":0,
                "in_cod_utente":"<codice_utente>",
                "in_includi": 0,
                "in_nr_magazzino": 0
        }
}
```

- **in_tipo_lotto:** codice alfanumerico del tipo lotto (es. “aa”)
- **in_cod_articolo:** stringa codice articolo
- **in_tp_cod_utente:** modalità di ricerca per codice utente
  - 0 = completo
  - 1 = inizia per
  - 2 = contiene
- **in_codice_utente**: codice utente del lotto case sensitive (facoltativo)
- **in_includi:** tipo inclusione
  - 0 = include tutti i lotti del solo archivio corrente, tranne quelli con stato B/N, con data validità inferiore alla data terminale o con quantità 0
  - 1 = include tutti i lotti del solo archivio corrente
  - 2 = come 0 ma solo dei dati nello storico
  - 3 = come 1 ma solo dei dati nello storico;
  - 4 = come 0 ma dei dati sia nel corrente che nello storico;
  - 5 = come 1 ma dei dati sia nel corrente che nello storico;
- **in_numero_magazzino:** numero del magazzino da 0 a 255: 0=tutti i magazzini

NOTA: in_tipo_lotto e in_cod_articolo sono mutuamente esclusivi. Se indicati entrambi viene considerato solo in_cod_articolo

Riassunto dei comandi RESTful

	Metodo http	PATH	Query String	HEADER	BODY	Success http Code

Creazione	POST	https://base_address/webapi/<nome_risorsa>	SI	Authorization
				Content-Type
				Coordinate-Gestionali
					{
					<dati>
					}
						201 (Created)

Lettura	GET	https://base_address/webapi/<nome_risorsa> https://base_address/webapi/<nome_risorsa>/<codice>	fields max next info	Authorization Content-Type Coordinate-Gestionali	{}	200 (OK)

Aggiornamento	PUT	https://base_address/webapi/<nome_risorsa>/<codice>	SI	Authorization
				Content-Type
				Coordinate-Gestionali
					{
					<dati>
					}
						204 (No Content)

Eliminazione	DELETE	https://base_address/webapi/<nome_risorsa>/<codice>	NO	Authorization Content-Type Coordinate-Gestionali	{}	204 (No Content)
Ricerca	POST	https://base_address/webapi/<nome_risorsa>	fields max next	Authorization Content-Type Coordinate-Gestionali	filtri:[{…}, {…}]	200 (OK)

4. ACCESSO AI DATI DEL GESTIONALE

Per gli archivi ad un solo livello (articoli, clienti, fornitori etc.) per recuperare le informazioni di tutto l'archivio è sufficiente utilizzare qualsiasi comando REST con un'unica richiesta.

Invece per quanto riguarda archivi che presentano più livelli (tipicamente i documenti) può essere necessario effettuare più chiamate per avere i dati nella loro completezza.

Prendiamo come esempio i documenti: questi archivi sono composti da una tabella master e una tabella di dettaglio (testata + righe). Per avere quindi il dettaglio completo delle testate e di tutte le righe che compongono il documento sarà necessario utilizzare più chiamate REST.

Per i documenti c'è un contenitore dedicato in cui poi andranno specificate le varie entità a cui si vuole accedere.

Il path sarà di questo tipo

COMANDO https://base_address/webapi/risorse/documenti/<risorsa>

Prendiamo come esempio i documenti di tipo *ordini clienti*.

Per referenziare i dati di testata il path sarà di questo tipo

GET https://base_address/webapi/risorse/documenti/ordini-clienti

Con questo comando viene restituita una lista di tutte le testate dei documenti.

Se invece si vogliono avere i dettagli sulle righe dei documenti bisogna utilizzare un comando di questo tipo

GET https://base_address/webapi/risorse/documenti/ordini-clienti/righe

In questo caso verranno restituite tutte le righe di tutti i documenti di tipo *ordini-clienti*. Sarà poi compito del programmatore utilizzare i filtri in modo da leggere/scrivere solo le righe di determinati documenti.

Nel caso si debba specificare una chiave, bisognare comporla inserendo la sigla, la serie e il numero del documento in questo modo (per i movimenti di magazzino va aggiunto anche il codice del conto)

GET https://base_address/webapi/risorse/documenti/ordini-clienti/OC+1+55

Nota: i campi chiave necessari possono essere recuperati visionando la risposta dell'HELP. Per ogni entità sarà presente un elemento "chiavi" contenente tutte le chiavi necessarie per completare la richiesta.

Nota: i dati restituiti da una GET su una lista possono essere leggermente differenti da quelli restituiti dalla GET di una singola entità. Ad esempio nelle liste gli array sono tutti al massimo di 2 livelli, mentre nelle singole entità gli array possono essere restituiti fino a 3 livelli.

4.1 ESEMPIO GESTIONE ORDINI CLIENTI

Creazione di un ordine

Richiesta

POST https://base_address/webapi/risorse/documenti/ordini-clienti

```
{
   "id_riga":[
      [
         1,
         1
```

```
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ]
   ],
   "mmart":[
      [
         1,
         "ATT002"
      ]
   ],
   "mmali":[
      [
         1,
         "22"
      ]
   ],
   "mmqta":[
      [
         1,
         3
      ]
   ],
   "codice_agente":"601.00001",
   "serie":1,
   "sigla":"OC",
   "numero":0,
   "mmdat":"20210825",
   "mmcli":"501.00022"
}
```

Modifica di un ordine

Richiesta

PUT https://base_address/webapi/risorse/documenti/ordini-clienti/OC+1+2

```
{
   "id_riga":[
      [
         1,
         1
      ]
   ],
   "tp_riga":[
      [
         1,
         "R"
      ]
   ],
   "codice_agente":"601.00002"
}
```

NOTA: i dati che referenziano le righe del documento devono essere sempre tutti specificati. Le righe non dichiarate verranno eliminate. In questo esempio è stato indicato che nel documento esiste solo una riga e di tipo "R". Se il documento avesse avuto altre righe sarebbero state eliminate. ATTENZIONE!

In questo caso si vuole modificare solo il codice agente lasciando tutto il resto inalteraro. Gli indici di riga e il tipo di riga devono essere sempre specificati.

Lettura di tutti gli ordini

Richiesta

GET https://base_address/webapi/risorse/documenti/ordini-clienti

Lettura di un ordine

Richiesta

GET https://base_address/webapi/risorse/documenti/ordini-clienti/OC+1+2

Cancellazione di un ordine

Richiesta

DELETE https://base_address/webapi/risorse/documenti/ordini-clienti/OC+1+2

Lettura di tutte le righe di un particolare ordine

Richiesta

POST https://base_address/webapi/risorse/documenti/ordini-clienti/righe/ricerca

```
{
   "filtri":[
      {
         "campo":"sigla",
         "condizione":"=",
         "valore":"OC"
      },
      {
         "campo":"serie",
         "condizione":"=",
         "valore":1
      },
      {
         "campo":"numero",
         "condizione":"=",
         "valore":3
      }
   ]
}
```

Ritorna tutte le righe documento dell'OC1/3

Nota: le righe del documento, essendo una lista, possono essere modificate solo quando si esegue la PUT sul documento che le contiene specificando riga per riga i campi da modificare.

Trasformazione di un ordine in d.d.t

PUT https://base_address/webapi/risorse/documenti/ordini-clienti/OC+1+2?sigla_trasformazione=BC

```
{
   "id_riga":[
      [
         1,
```

```
            1
        ]
    ],
    "tp_riga":[
        [
            1,
            "R"
        ]
    ]
}
```

Elimina il documento OC1/2 e crea la BC corrispondente

5. CHANGELOG

Di seguito saranno indicati tutti i cambiamenti di questo manuale rispetto alle versioni precedenti.

V2.26: DALLA VERSIONE GESTIONALE 2025F - V877 - 87700

Nuovi end-point

È stato aggiunto l'*end-point* con la lista delle classi Docuvision solo GET lista: *dati-generali/classi-docuvision*

Nuovi servizi

È stato aggiunto il servizio per la ricerca dei lotti specificando ulteriori parametri come codice articolo o codice utente. Per tutti i dettali fare riferimento alla sezione dedicata.

Nuove funzionalità e miglioramenti

- Nell'end-point *documenti/movimenti-magazzino* in fase di trasformazione da OC/OX il campo *tipo_stato_riga* accetta come parametro anche la "e" come avviene in emissione/revisione documenti. In questo modo, in caso di evasione parziale di una riga, il residuo rimane sul documento originale (OC/OX) in stato "E".
- Ottimizzata la lettura degli archivi MyDB tramite end-point di ricerca passando da ordini di grandezza di secondi a millisecondi

Correzione anomalie

- Nel caso di POST o PUT su documenti di tipo ordine-cliente o ordine-fornitore in presenza di righe che gestivano più lotti poteva erroneamente essere scritto un lotto appartenente ad un altro articolo senza segnalare alcun errore.
- Nell'end-point articoli (lista) veniva mostrato, facendo *info=true*, il campo *descr_lingua* anche se non effettivamente presente tra i campi disponibili.

Correzioni al manuale

E' stato specificato meglio che il case_insensitive si applica per tutti gli operatori che coinvolgono stringhe

V2.25: DALLA VERSIOEN GESTIONALE 2025E1 - V876A - 87601

Nuovo campi

Nel servizio leggi_pratica_dr sono stati aggiunti, oltre ai già presenti campi sul conteggio immobili, anche quelli sul conteggio dei terreni: *nr_terreni*, *nr_terreni_prop*, *nr_terreni_no_pro*.

Prima era tutto conteggiato come immobile.

Correzione refusi

Nel servizio *get_allegato_archivio* i campi *id_dv* e *codice* sono mutuamente esclusivi

V2.24: DALLA VERSIONE GESTIONALE 2025E - V876 - 87600

Nuovi campi

Nell'end-point *dati-generali/parametri-aziendali* nella sezione relativa ai parametri di produzione sono stati aggiunti i seguenti campi:

"fissa_prezzi_bl": "Anagrafica",

"gest_avanzamento_parziale_bl_collegate_oc": "N",

"gest_bl_visualizza_pf_per_cliente": "N",

"gest_escludi_da_odp": "N"

Correzione anomalie

Se un cliente aveva più di un referente e, tramite webapi, veniva aggiornato uno dei campi, venivano cancellate tutte le altre anagrafiche dei referenti associate al cliente (Sol. 78572)

La funzione 'gest_quadro_LM' tornava sempre la "N" di quadro non compilato anche nel caso di sua avvenuta compilazione (Sol. 78578)

V2.23: DALLA VERSIONE GESTIONALE 2025D - V875 - 87500

Nuovi campi

Negli end-point dei documenti è stato aggiunto il campo di riga *cod_rif_orig_pf*


```
{
        "nome": "cod_rif_orig_pf",
        "descrizione": "Riferimento PF origine della BL; solo produzione",
        "tipo": "Numerico",
        "dimensione_array": "dinamico"
}
```


Nell' end-point *tipi-lotti-matricole* sono stati aggiunti i seguenti campi:


```
{
        "nome": "data_obbl",
        "descrizione": "Tipo lotto : data validita' obbligatoria",
        "tipo": "Alfanumerico"
},
{
        "nome": "giorni_autocod",
        "descrizione": "Tipo lotto : giorni per autocodifica",
        "tipo": "Numerico"
},
{
        "nome": "giorni_manuale",
        "descrizione": "Tipo lotto : giorni per codifica manuale",
        "tipo": "Numerico"
},
{
        "nome": "giorni_giacenza",
        "descrizione": "Tipo lotto : giorni minimi giacenza magazzino",
        "tipo": "Numerico"
}
```


Nuove funzionalità

Nel servizio *get_lsita_deleghe* è stata aggiunta la possibilità di filtrare a partire da una data di ultima modifica in poi (vedi sezione dedicata)

L'end-point *dati-generali/tipi-lotti-matricole* è stato completato con tutti i metodi CRUD aggiungendo POST, PUT e DELETE (vedi sezione dedicata)

Migliorie

- Migliorata la gestione in caso di presenza di sotto aziende anche se non viene specificata nell'header della richiesta evitando la riapertura dell'azienda
- Velocizzata dell'86% la lettura di liste MyDB con molti record (centinaia di migliata)

Correzione anomalie

Negli end-point dei *referenti* facendo una PUT per cancellare dei campi inserendo stringa vuota ("") i campi non venivano cancellati (Sol. 77921)

V2.22: DALLA VERSIONE GESTIONALE 2025C - V874 - 87400

Nuovi end-point

Produzione: tabella con la correlazione delle unità di misura *dati-generali/correlazione-unita-misura*

Nuovi servizi

Per le aziende di livello produzione o superiore è stato aggiunto il servizio per leggere i calendari di produzione: get_calendari_produzione.

Nuovi campi

- Nel servizio per ottenere le strutture MyDB (*lista_strutture_mydb*) viene restituito anche il campo *ordinamento*
- Negli end-point dei documenti di magazzino è stato aggiunto il campo *cod_rif_pf_bl:*
  {
  "nome": "cod_rif_pf_bl",
  "descrizione": "riferimento PF della BL; solo produzione",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"
  }

Modifiche al protocollo

- Per l'end-point dei progressivi degli articoli (*articoli/<codice>/progressivi*), in caso l'articolo non abbia mai avuto nessuna movimentazione viene tornato un json vuoto invece che errore 400
- Per gestire meglio la concorrenza è stata modificata la procedura di paginazione: la logica della gestione dei NEXT rimante identica ma non possono essere salvati per essere riusati per chiamate successive. Per tutti dettagli si veda la sezione dedicata.

Correzione anomalie

In alcuni casi particolari, quando da WebAPI si eseguiva una trasformazione di un documento inserendo nuove righe, si perdevano le provvigioni su tutte le righe del documento (Sol. 77778)

V2.21: DALLA VERSIONE GESTIONALE 2025B - V873 – 87300

Nuovi servizi

Aggiunto il nuovo servizio per autenticare i campi di tipo AUT sugli archivi MyDB. Vedi sezione dedicata.

Nuovi end-point

Esposta la tabella con la gestione delle lingue straniere: *dati-generali/lingue-straniere.* Sono disponibili tutti i metodi CRUD tranne POST per la ricerca.

Nuove funzioni

Nell'*end-point dati-generali/categorie-statistiche-cli-for* sono stati esposti tutti i metodi CRUD oltre la lista (tranne la ricerca)

Nuovi campi

- Nell'end-point documenti/lavorazione/bolle è stato aggiunto il campo *codice_art_prg*

  {

  "nome": "cod_art_dbp",

  "descrizione": "Codice articolo con DBP",

  "tipo": "Alfanumerico"

  }
- Nell'end-point *anagrafica-contatti* è stato aggiunto il campo *fino_val_spese_sped:*
  {

  "nome": "fino_val_spese_sped",

  "descrizione": "Fino a valore spese spedizione",

"tipo": "Numerico con virgola"

}

- Nell'end-point clienti è stato aggiungo il campo *fino_spese_sped*

{

"nome": "fino_spese_sped",

"descrizione": "Vettore - Fino a valore spese di spedizione",

"tipo": "Numerico con virgola"

},

- Nel servizio leggi_prtica_dr nel response sono stati aggiunti i campi relativi all'IBAN:
  - elenco_iban
  - iban_abituale
  - modalita_invio
- Nel servizio spezzariga_bl è stato aggiunto il nuovo parametro *solo_bl*="S/N" che, se attivo il parametro di produzione "Avanzamento parziale bl collegate a oc", spezza la riga solo sulla BL lasciando invariato l'OC

Correzione anomalie

- Nel servizio *leggi_comunicazioni_lipe* non era gestito correttamente il trimestre a cavallo d'anno (trimestre anno precedente ma versato in anno successivo)
- Nel servizio lista delle deleghe non era gestito correttamente il modello pratica RNC

Correzione refusi

- Nella sezione relativa alla ricerca nei filtri era stato indicato erroneamente l'indice array come stringa invece che come intero
- Nella sezione relativa alla lettura dei documenti docuvision non era stata documentata la paginazione in caso di superamento dei limiti
- Nella sezione relativa alla lista delle deleghe sono stati specificati i valori disponibili per il campo *in_mod_pratica*

V2.20: DALLA VERSIONE GESTIONALE 2025A - V872 - 87200

Nuove funzionalità

- Nel servizio *lista_docdv* è stata aggiunta la possibilità di filtrare, oltre che sui campi presenti nella risorsa, anche su altri parametri. Si veda la sezione dedicata.
- Nel servizio con la lsita delle strutture MyDB sono visibili anche eventuali campi di autenticazione, che saranno però non presenti in chiaro sul particolare archivio MyDB che ne fa uso per motivi di sicurezza. Per la gestione di questi campi verrà implementanto un servizio ad hoc.

Nuovi campi

- Nell'*end-point articoli* è stato aggiunto il campo *peso_medio_unit*
- Nel servizio *lista_strutture_mydb* sono stati aggiunti i metadati *data_ult_mod* e *checksum*
- Nell' *end-point anagrafica-contatti* sono stati aggiunti i seguenti campi:

  valuta_estera, cat_statistica, zona, addebito_bollo, gest_cig_cup, rg_bolle, rg_effetti, proposto_e_ord, sconto_icz, trasporto_mezzo, cod_vett_abituale, tipo_porto, descr_porto, codice_incoterms, gest_calc_colli, gest_calc_peso, tipo_spese_sped, val_spese_sped, asp_est_beni

  *(*per i dettagli su descrizioni e tipi fare *GET anagrafica-contatti?info=true)*

V2.19: DALLA VERSIONE GESTIONALE 2024I3 - V871C - 87103

Correzione anomalie

- nell'*end-point* dell'anagrafica dei contatti l'autocodifca non era gestita inserendo come codice 0
- nell'*end-point* dell'anagrafica dei contatti manca il campo *codice_pdc* nella lista
- nell'*end-point* delle righe della prima nota mancava il campo *nr_documento*

V2.18: DALLA VERSIONE GESTIONALE 2024I - V871 – 87100

Nuovi end-point

- Aggiunto l'end-point dell'anagrafica dei contatti completo di tutti i metodi CRUD: *anagrafica-contatti.* Per i dettagli sui campi usare l'help in linea GET <addr>/risorse/*anagrafica-contatti?info=true*

Nuovi campi

- Nell'end-point *dati-generali/taglie* è stato aggiunto il campo *descr_tg_ext* con il quale è possibile gestire descrizioni di taglie più lunghe di 3 caratteri.
- Negli end-point degli ordini sono stati esposti anche i campi con i riferimenti alle bolle di produzione (solo per singola entità):

{

"nome": "**nr_bl_colleg**",

"descrizione": "num. bolla.lav.collegata*g",

"tipo": "Numerico",

"dimensione_array": "dinamico"

},

{

"nome": "**nr_rif_doc_colleg**",

"descrizione": "num. rif.doc.collegato*g",

"tipo": "Numerico",

"dimensione_array": "dinamico"

}

- Nel servizio della lista delle deleghe ora vengono ritornati anche i riferimenti all'azienda collegata alla pratica o viceversa alla pratica collegata all'azienda:
  - *mod_pratica_collegata*
  - *pratica_collegata*
  - *interno_collegato*
  - *sigla_azienda_collegata*

Nuovi servizi

Servizio per la lettura della configurazione dei numeratori per i vari tipi di documento. Si veda la sezione dedicata.

Nuove funzionalità

- Nel servizio della lettura dei cancellati è stato aggiunto un parametro per indicare se la risorsa cancellata è presente o meno nello storico
- Nel servizio per la lettura degli allegati docuvision sono stati aggiungi anche i parametri per identificare univocamente l'allegato e la relativa versione

V2.17: DALLA VERSIONE GESTIONALE 2024H - V870 – 87000

Nuove Funzionalità

- Negli end-point dei documenti di magazzino sono gestiti anche i documenti ivati FX/BX/NX/RX
- Negli end-point dei documenti di magazzino sono stati inclusi anche i documenti Deposito Ordine: DO/DX

Nuovi end-point

L'*end-point* degli impegni è stato spostato in *documenti/lavorazione* ed ora gestisce obbligatoriamente anche il parametro *progressivo:*

"nome": "**progressivo**",
"descrizione": "Progressivo inserimento",
"tipo": "Numerico"

In caso di POST con gestione degli impegni multi-riga, se è valorizzato con -1, viene inserito un nuovo impegno assegnando automaticamente il progressivo. In tutti gli altri casi se non viene specificato nulla viene sempre valorizzato a 0.

Nel caso che la gestione degli impegni multi-riga sia disabilitato il parametro deve sempre essere passato con 0 e nella POST per la creazione di un nuovo impegno deve essere omesso.

Esempio
GET documenti/lavorazione/impegni/52+0+1+1+1+FUELEX+0+**0**

Il vecchio *end-point* impegni continuerà a funzionare come prima, ma a lungo termine potrebbe essere deprecato. È pertanto consigliato aggiornare le procedure usando il nuovo end-point.

Nuovi servizi

Sono stati implementati i servizi per la lettura degli allegati docuvision. Per tutti i dettagli si veda la sezione dedicata.

Servizio per ottenere la lista di tutte le strutture MyDB utilizzate. Per tutti i dettagli si veda la sezione dedicata.

Nuovi campi

- Negli end-point dei documenti sono stati aggiunti i dati di riga per i centri di costo/ricavo:
  "nome": "**id_causale_riga**",
  "descrizione": "Causale -v._MMCMO-/Contropartita -max 32 x Doc.",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"

  "nome": "**id_ncau_riga**",
  "descrizione": "Causale di riga",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"

  "nome": "**id_ccr_riga**",
  "descrizione": "Centro costo ricavo",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"
- Negli end-point dei movimenti di magazzino è stato aggiunto il campo sul tracking number:
  "nome": "**nr_tracking**",
  "descrizione": "Tracking number",
  "tipo": "Alfanumerico",
  "dimensione_array": "dinamico"
- Negli end-point dei movimenti di magazzino sono stati aggiungi i riferimenti ai documenti Deposito Ordine:
  "nome": "**rif_sgl_ddep**",
  "descrizione": "Riferimento sigla documento deposito",
  "tipo": "Alfanumerico",
  "dimensione_array": "dinamico"

  "nome": "**rif_serie_ddep**",
  "descrizione": "Riferimento serie documento deposito",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"

  "nome": "**rif_nr_ddep**",
  "descrizione": "Riferimento numero documento deposito",
  "tipo": "Numerico",
  "dimensione_array": "dinamico"

  "nome": "**rif_dt_ddep**",
  "descrizione": "Riferimento data documento deposito",
  "tipo": "Alfanumerico",
  "dimensione_array": "dinamico"

Correzione anomalie

- Per gli end-point dei clienti e dei fornitori il campo nome_ricerca era gestito in modo errato e non era possibile fare POST o PUT valorizzando questo campo. È stato inserito un nuovo campo dedicato per i clienti fornitori: *nome_ricerca_cf*
- Nell'end-point con la lista delle strutture articolo la descrizione non era presente: è stato aggiunto il campo *descr_struttura*
- In presenza di un documento trasformato il servizio di lettura righe documenti non valorizzava correttamente le variabili dei lotti.
- La ricerca con filtro su un archivio con numero di campi maggiore di quello della successiva ricerca sempre con filtro, ad esempio prima clienti e fornitori e poi alias articolo, portava all'errore di violazione di protezione di memoria.

Modifiche a questo manuale

È stato rimosso il capitolo Allegati in quanto obsoleto. Per tutti i dettagli sui campi dei vari end-point è necessario consultare l'help in linea (info=true)

V2.16: DALLA VERSIONE GESTIOANLE 2024G - V866 – 86600

Correzione anomalie
In alcuni contesti le chiavi che identificano una particolarità potevano contenere caratteri speciali non utf8

V2.15: DALLA VERSIONE GESTIONALE 2024F - V865 – 86500

Nuove funzionalità

Nei servizi *get_lista_pratiche_dr* è stata aggiunta la possibilità di usare dei filtri (vedi sezione dedicata)

Nuovi campi

- Nell'end-point *documenti/lavorazione/bolle/<codice_bolla>* è stato aggiunto il campo *tp_bl_storico: Bolla storicizzata (S/N solo lettura)*
- Nell'end-point *indirizzi-spedizio*ne sono stati aggiunti i seguenti campi:
  - *rif_esterno_tp*: Riferimenti esterni tipo
  - *rif_esterno_sgl*: Riferimenti esterni sigla
  - *rif_esterno_nr*: Riferimenti esterni numero
  - *rif_esterno_cod*: Riferimenti esterni codice
  - *rif_esterno_dt*: Riferimenti esterni data
- Nel servizio sulla lista delle deleghe (get_lista_deleghe) sono stati esposti i seguenti campi:
  - "nr_rata": intero – numero rata a cui fa riferimento la delega
  - "nr_rate": intero – numero di rate in cui è stato disposto il pagamento
  - "id_delega_origine": stringa – id della delega di origine
  - "maggiorazione": flag S/N per sapere se nella delega è stata applicata o meno una maggiorazione.

Nuovi end-point

- *risorse/documenti/moduli-stampa*: GET lista e singola risorsa dei moduli di stampa disponibili (es. GET risorse/documenti/moduli-stampa/OC)
- *dati-generali/centri-costo-ricavo:* GET lista e POST ricerca per le tabelle dei centri costo/ricavo

V2.14: DALLA VERSIONE GESTIONALE 2024E - V864 - 86400

Nuovi campi

Negli end-point delle righe dei documenti (es. ordini-clienti/righe) è stato esposto il campo *sc_merce_doc*: sconto merce del documento.

Nuove funzionalità

Nei nuovi servizi di stampa dei moduli pdf è stato aggiunto il parametro per specificare il modulo di stampa: *cod_modulo. (vedi sezione dedicata)*

Correzione anomalie

Il filtro sul campo *data_ult_mod* sull'end-point documenti/movimenti-magazzino/righe/ricerca tornava l'errore "Nome campo filtro 'data_ult_mod' non valido"

V2.13.1: DALLA VERSIONE GESTIONALE 2024D1 - V863A - 86301

Negli end-point dei documenti è stato aggiunto il campo *cod_art_anag*: Codice articolo proprietario dell'anagrafica

V2.13: DALLA VERSIONE GESTIONALE 2024D -V863 - 86300

Modifiche al protocollo

Nei filtri di ricerca è stata aggiunta la possibilità di controllare una condizione in OR su più valori di un campo. Non è ancora possibile usare la logica OR tra diverse condizioni. Vedi sezione dedicata.

Nuove funzionalità

- Servizio per ottenere la lista dei record cancellati nei vari end-point: *get_registro_operazioni_utente.* Vedi sezione dedicata.
- Servizi per ottenere la stampa PDF di documenti. Vedi sezione dedicata.
- Servizio per l'inserimento di righe in un documento con la restituzione dei nuovi id riga. Vedi sezione dedicata.
- Nel servizio *get_lista_deleghe* sono stati aggiunti nuovi filtri:
  - **in_tipo** : filtra le deleghe per tipo
  - **in_definitiva** : filtra le deleghe "definitive"
  - **in_accise** : filtra le deleghe con "accise"
  - **in_ravvedimento_operoso** : filtra le deleghe di "ravvedimento"
  - **in_avviso_bonario** : filtra le deleghe relative ad avvisi bonari
  - **in_codice_invio** : filtra le deleghe in base alla modalità di invio della delega

Nuovi campi

Negli end-point *ordini-clienti*, *ordini-fornitori e preventivi* sono stati aggiunti i seguenti campi:

- tipo_imballo: Imballo variabile in emissione
- dt_inizio_rateo: Data inizio Rateo Risconto
- dt_fine_rateo: Data fine Rateo Risconto

Nel end-poitn *liste-prelievo* sono stati esposti i seguenti campi:

- descr_articolo: Descrizione articolo (array)
- nota: Nota documento
- note_documento: Annotazioni documento (array)
- note_articolo: Annotazioni articolo (array)

Correzione refusi

Nelle liste, il campo *data_rich_elab è errato:* il nome corretto è *data_ric_elab*

V2.12: DALLA VERSIONE GESTIONALE 2024C - V862 - 86200

Nuovi campi

- Negli end-point *ordini-clienti* e *ordini-fornitori* è stato aggiunto il campo *id_ncau_riga*: Causale di riga
- Per tutti gli end-point che gestiscono metadati, nel caso di GET su lista, viene sempre riportato anche il campo *data_rich_elab* in cui viene riportata la data in cui è stata fatta la richiesta (sarà sempre la stessa in caso di next, in quanto si riferisce alla prima richiesta). Può essere utile nel caso si voglia tenere traccia delle modifiche ed usarla come filtro maggiore o uguale sul campo di data ultima modifica ("data_ult_mod") nella richiesta successiva in modo da avere solo i modificati dalla richiesta precedente alla data della nuova richiesta.
- Nell'end-point *movimenti-magazzino* sono stati aggiunti i riferimenti alle dichiarazioni d'intento:
  - anno_int: Anno ricevimento lettera d'intento (testata)
  - prog_int: Progressivo lettera d'intento (testata)
  - anno_int_riga: Anno ricevimento lettera d'intento (riga)
  - prog_int_riga: Progressivo lettera d'intento (riga)
- Nel servizio *get_lista_deleghe* viene ritornato anche il campo di un carattere *avviso_bonario*.

Nuove funzionalità

- Servizio per la lettura delle dichiarazioni iva annuali. Vedi sezione dedicata
- Servizio per la lettura delle comunicazioni LIPE. Vedi sezione dedicata.

V2.11: DALLA VERSIONE GESTIONALE 2024B - V861 - 86100

Modifiche al protocollo

Aggiunta condizione *inizia_per* nei filtri (vedi sezione dedicata)

Nuovi campi

- Nell'end-point *articoli*, solo per la lista, è stata aggiunta la descrizione completa come somma dei campi descrizione+descrizione_agg: il campo si chiama *descr_completa*
- Nell'end-point *dati-generali/pagamenti* è stato aggiunto il campo *conto_pagamento*
- Nell'end-point *documenti/movimenti-magazzino sono stati aggiunti i seguenti campi accessibili solo con GET su singola risorsa (non in lista):*
    - *cod_ctrpar_acc: codice contropartita acconto*
    - *cod_ctrpar_ab: codice contropartita abbuono*
- Negli end-point sui progressivi degli articoli (*risorse/progressivi-articoli* e *risorse/<articolo>/progressivi*) sono stati aggiunti anche i dettagli sugli impegni per picking:
    - qta_picking
    - qta_picking_tg
    - colli_picking
    - tara_picking
- Nell'end-point *parametri-aziendali* è stata aggiunta la sezione *parametri_contabili*
- Nell'end-point *aziende* è stato aggiunto, in creazione (POST), il parametro sul livello aziendale: *in_livello_azienda*
    - "NULL" - Azienda di livello NULLO
    - "1" - Azienda di livello BASE
    - "1F" - Azienda di livello BASE CON FATTURAZIONE
    - "2" - Azienda di livello ESTESO
    - "3" - Azienda di livello PRODUZIONE
- Nel servizio *avanzamento_produzione* è stato aggiunto il nuovo parametro in ingresso cod_magazzino: solo per operazione "C" indica il magazzino su cui fare il CL

V2.10: DALLA VERSIONE GESTIONALE 2024A3 - V860C - 86003

Correzioni e anomalie:

- nel servizio *get_lista_deleghe* è stato inserito un codice di errore per ogni azienda che per qualche motivo non è stato possibile processare
- negli end-point *anagrafica-unica* e *anagrafica-unica/storico* a volte la lista andava in errore a causa di una apertura anomala dell'archivio delle divise.

Nuovi campi:

Nell'end-point *dati-generali/parametri-aziendali* è stato aggiunto anche il parametro per la gestione dello scadenzario a partite

V2.9.1

Correzione refusi

- Il precedente Changelog (v2.9) è errato, solo l'ultimo punto è stato rilasciato. I rimanenti punti usciranno in versione 2024B

V2.9: DALLA VERSIONE GESTIONALE 2024A1 - V860A - 86001

Modifiche al protocollo

~~Aggiunta condizione *inizia_per* nei filtri (vedi sezione dedicata)~~

Nuovi campi

- ~~Nell'end-point *articoli*, solo per la lista, è stata aggiunta la descrizione completa come somma dei campi descrizione+descrizione_agg: il campo si chiama *descr_completa*~~
- ~~Nell'end-point *dati-generali/pagamenti* è stato aggiunto il campo *conto_pagamento*~~

- ~~Nell'end-point *documenti/movimenti-magazzino sono stati aggiunti i seguenti campi accessibili solo con GET su singola risorsa (non in lista):*~~
  - ~~*cod_ctrpar_acc: codice contropartita acconto*~~
  - ~~*cod_ctrpar_ab: codice contropartita abbuono*~~
- Nel servizio sullo sblocco delle deleghe (cambia_stato_delega) è stato aggiunto il codice di errore oltre il valore

V2.8: DALLA VERSIONE GESTIONALE 2024A - V860 - 86000

Nuovi campi

Negli end-point dei movimenti di magazzino sono stati aggiunti i seguenti campi:

- "gest_bollo_virt" : "Gestione bollo virtuale nel documento S/N"
- "dt_inizio_rateo" : "Data inizio Rateo Risconto"
- "dt_fine_rateo" : "Data fine Rateo Risconto"

Nell'end-point *articoli* è stata aggiunta la variabile *gest_peso_wms* alfanumerico "S/N": *Pesata WMS*

Nell'end-point *lotti* è stata aggiunta la variaile *gest_stor_lotto* alfanumerico "S/N": *Lotto storicizzato S/N*

Nuovi end-point

dati-generali/parametri-aziendali: solo GET singola risorsa. Verranno di volta in volta aggiunti nuovi parametri, al momento sono presenti solo alcuni parametri di magazzino e produzione.

Nuove funzionalità

- Nell'end-point *aziende* è stato aggiunto anche il metodo POST per la creazione di una nuova azienda e il collegamento all'anagrafica unica: vedi sezione dedicata
- Nel servizio *get_prog_ubicazioni* è stata inserita la possibilità di filtrare anche per *id* oltre che per *data_ult_mod:* *vedi sezione dedicata*

Nuovi servizi

- Servizio per la lettura di un modello dichiarativo: vedi sezione dedicata
- Servizio per ottenere la lista delle pratiche dichiarativi: vedi sezione dedicata
- Servizio per ottenere la lista delle deleghe dichiarativi: vedi sezione dedicata

Modifiche al manuale

Aggiunta una nota sulla gestione dei documenti in caso di azienda senza gestione taglie (vedi capitolo dedicato)

Modifiche al protocollo

Come indicato nel *changelog* della versione 2.6 i seguenti end-point non saranno più raggiungibili. Qualora siano ancora utilizzati è necessario usare il nome corretto sostituendo l'underscore con il trattino ('-' invece che '_')

- categorie_statistiche_cli_for
- categorie_provvigioni_articoli
- condizioni_generali_pagamenti
- categorie_sconti_articoli

V2.7: DALLA VERSIONE GESTIONALE 2023H - V851 - 85100

Nuovi campi

- Nell'end-point *prima-nota/righe* è stata aggiunta la data di ultima modifica come per le testate: *data_ult_mod*
- Nell'end-point *articoli* è stata aggiunta la data di ultima modifica delle sole immagini: *dt_variaz_img*
- Nell'end-point *anagrafica-unica* in caso di GET della singola risorsa è disponibile anche la data di inizio del particolare blocco di informazioni xxx_stor_iniz
- Nell'end-point *anagrafica-unica* in caso di GET della singola risorsa è disponibile anche la data di inizio validità e fine validità alla particolare data chiesta (*data_iniz_valid*, *data_fine_valid*). Le date devono essere intese come validità per l'intera anagrafica e non solo dei singoli blocchi di informazioni: *data_iniz_valid* sarà la più grande di tutte le date di inizio validità (xxx_stor_iniz) dei singoli blocchi mentre *data_fine_valid* sarà la più piccola di tutte le date di fine validità (xxx_stor_data) dei singoli blocchi di informazioni.
- Nell'end-point *anagrafica-unica/storico* inserito il campo *tp_dati* con l'informazione sul tipo di dato che è stato modificato (vedi sezione dedicata)
- Nell'end-point *documenti/ordini-fornitori/righe* è stato esposto il campo *id_riga_orig* come negli ordini cliente.
- Nell'end-point *lotti* sono stati aggiunti i metadati di utente e data ultima modifica: *utente_ult_mod, data_ult_mod*

Nuovi end-point

E' stato aggiunto l'end-point *anagrafica-unica/storico* solo GET e POST per la ricerca con l'elenco di tutte le storicizzazioni

Nuove funzionalità

Nell'end-point *anagrafica-unica* è stato aggiunto un nuovo tipo di storicizzazione nel parametro *storicizza* in query string. Storicizza=R aggiorna l'anagrafica e storicizza alla data indicata (vedi dettagli nella sezione dedicata)

Nuovi Servizi

- Servizio per leggere le informazioni sulle località italiane (CAP, Provincia, Comune, Località, Zone). Vedi sezione dedicata.
- Sevizio per il conteggio del numero delle operazioni contabili. Vedi sezione dedicata.

Modifiche al protocollo

End-point *anagrafica-unica,* corretti i nomi delle seguenti variabili per uniformità con la convezione dei nomi web-api:

Vecchio nome	Correzione
data_validita	data_fine_valid
anag_pers_fisica	compresa in anag_tipo_sogg
~~anag_tipo_piva~~	**eliminata**
anag_partita_iva	anag_piva
anag_rag_sociale	anag_rag_soc
anag_denominazione	anag_denom
res_stesso_sede_leg	res_stesso_dom
leg_soc_cod_fiscale	leg_soc_cod_fis
leg_soc_rag_sociale	leg_soc_rag_soc
docum_rilas	docum_aut_ril

docum_data_rilas	docum_data_ril

Correzione errori

Nell'end-point *alias-articoli* il campo *quantita* gestiva erroneamente solo numeri interi (definito come Numerico). Corretto in "Numerico con virgola".

V2.6.2: DALLA VERSIONE GESTIONALE 2023G3 - V850C - 85003

Correzione refuso

Nell'end-point *anagrafica-unica* era richiesto l'accesso con azienda anche se non necessario. È stato corretto ed è possibile usare *l'end-point* senza dover specificare le Coordinate-Gestionale

V2.6.1: DALLA VERSIONE GESTIONALE 2023G2 -V850B – 85002

Nell'end-point *anagrafica-unica* è stata aggiunta la possibilità di storicizzare documenti anche da WebAPI (vedi sezione dedicata).

V2.6: DALLA VERSIONE GESTIOANLE 2023G -V850 – 85000

Nuovi campi

Aggiunti i campi per i riferimenti esterni agli ordini di origine:

- rif_sgl_origine: Rif. Documenti Esterni - Sigla documento ordine di origine
- rif_nr_origine: Rif. Documenti Esterni - Num documento ordine di origine
- rif_dt_origine: Rif. Documenti Esterni - Data documento ordine di origine

Aggiunto campo *qta_tg_plus_prod* per la gestione delle taglie sul servizio avanzamento_produzione

qta_tg_plus_prod: array per la gestione delle taglie in caso di aggiunta di carico. La somma delle quantità delle taglie deve corrispondere con la quantità indicata nella variabile *qta_agg_carico*

Nuove risorse

- Aggiunto end-point alenco aziende: solo GET della lista delle aziende webapi/risorse/aziende
- Aggiunto end-point scadenzario: solo GET della collection e POST per la ricerca:
  - GET webapi/risorse/scadenzario?tipo=<tipo>
  - POST webapi/risorse/scadenzario/ricerca

  Dove <tipo> indica quale tipo di scadenzario si vuole leggere :
  - D: a Documento
  - P: a Partite
  - X: eXtracontabile.
- Aggiunto end-point su prima nota: implementati tutti i metodi di lettura e scrittura. Vedi capitolo dedicato.
- Aggiunto end-point anagrafica-unica: implementati tutti i metodi di letture e scritture. Vedi capitolo dedicato.

Nuovi Servizi

- calcolo dei totali documento
- calcolo dei totali di riga
- calcolo dell'esposizione del cliente

Correzione anomalie

- Nei progressivi degli articoli, se era presente un articolo con un lotto presente su più magazzini la procedura del calcolo dei progressivi scartava il lotto.
- Nell'end-point *articoli* in campi *scorta_minima* e *scorta_massima* erano invertiti.

- Nell'end-point *movimenti-magazzino* non erano stati esposti i campi realitivi agli articoli di tipo testo
- Nell'end-point *liste-prelievo* è stato aggiunto un controllo più restrittivo sui campi chiave (sigla, serie, numero, id_riga, prog_riga) i quali dovranno sempre essere indicati in caso di revisione (PUT)

Correzione refusi

Gli end-point sulle categorie statistiche dei clienti/fornitori, categorie provvigioni, condizioni pagamenti e categoria sconti erano stati nominati con il separatore *underscore "_"* invece che con il trattino "-". Per uniformità sono stati corretti:

- categorie-statistiche-cli-for
- categorie-provvigioni-articoli
- condizioni-generali-pagamenti
- categorie-sconti-articoli

E' necessario aggiornare tutte le procedure che utilizzano la nomenclatura errata in quanto da Gennaio 2024 non potranno più essere raggiungibili

Modifiche al protocollo

Per l'end-point *liste-prelievo*, in caso di modifica (PUT) della sola testata è ora necessario specicifare in query string il parametro *solo_testata=true* come avviene già per i documenti. Vedi sezione dedicata.

V2.5.1: DALLA VERSIONE GESTIONALE 2023F1 - V845A – 84501

Nuovi campi

In tutti gli end-point del contenitore documenti sono stati aggiunti i seguenti campi:

- *codice_cig: Codice CIG per appalti pubblici*
- codice_cup: Codice CUP per appalti pubblici

V2.5: DALLA VERSIONE GESTIONALE 2023F - V845 – 84500

Nuove risorse

Aggiunto end-point sulla cartella abbinamenti: (GET lista e POST ricerca) *risorse/dati-generali/cartella-abbinamenti*

Nuovi campi

In tutti gli end-point del contenitore documenti sono stati aggiunti i seguenti campi:

- *sc_merce_doc*: sconto merce del documento (solo nell'accesso alla singola singola risorsa)
- *abbuono*

V2.4: DALLA VERSIONE GESTIONALE 2023E2 - V844B - 84402

Nuove risorse

- Aggiunto end-point sulle categorie sconti degli articoli (solo GET lista): *risorse/dati-generali/categorie_sconti_articoli*
- Aggiunto end-point sulle categorie statistiche dei clienti fornitori (solo GET lista): *risorse/dati-generali/categorie_statistiche_cli_for*
- Aggiunto end-point sulle categorie delle provvigioni degli articoli (solo GET lista): *risorse/dati-generali/categorie_provvigioni_articoli*
- Aggiunto end-point sulle condizioni generali dei pagamenti (solo GET lista): *risorse/dati-generali/condizioni_generali_pagamenti*

Nuovi campi

Come per i documenti di tipo ordine anche per i movimenti di magazzino sono stati esposti, nelle liste, i campi sui riferimenti esterni:

- rif_sgl_attuale
- rif_nr_attuale
- rif_dt_attuale

Nuovi servizi

Servizio per la lettura di tutti gli articoli di tutte le ubicazioni. Per i dettagli si rimanda alla sezione dedicata.

V2.3: DALLA VERSIONE GESTIONALE 2023E - V844 - 84400

Nuovi campi

- Il campo *id_rif_testata* è stato riportato anche negli end-point delle righe dei movimenti di magazzino (movimenti-magazzino/righe)
- In tutti gli end-point dei Documenti è stato esposto anche il campo *cod_mndto_rid:* Codice mandatao RID SEPA

V2.2: DALLA VERSIONE GESTIONALE 2023D - V843 - 84300

Nuove risorse

- Lista posizione referenti: *risorse/dati-generali/posizione-referenti* solo metodo GET
- DBA articolo: *risorse/dba* sia GET che POST per la ricerca

Nuovi campi

Aggiunti i campi per la gestione del multi-agente nei documenti (solo per il metodo GET sul singolo documento):

- *cod_agente: array alfanumerico con codici multiagente*
- *cond_agente: array numerico condizione multiagente*
- *tipo_provv: array alfanumerico con tipo provvigione multiagente*
- *formula_pr: array alfanumerico con la formula della provvigione per il multiagente (valida solo per il 1° elemento)*
- *calc_formula_pr: array numerico con il calcolo della formula provvigione (solo lettura)*
- *quota_ripart_pr: array numerico con la quota di ripartizione delle provvigioni*
- *modalita_pr: array alfanumerico con la modalità delle provvigioni per multiagente*
- *imp_calc_pr: array alfanumerico con l'importo del calcolo delle provvigioni: 1 lordo, 2 netto*

Nuovi servizi

Servizio per il calcolo dei dati relativi all'esposizione di un cliente/fornitore. Vedi capitolo dedicato.

V2.1.1: DALLA VERSIONE GESTIONALE 2023C2 - V842B - 84202

Correzione errori
Facendo una richiesta di ricerca con il metodo POST sugli end-point dei documenti (es. documenti/ordini-clienti/ricerca) filtrando per data_ult_mod ritornava il seguente errore
6001 - errore gestionale [Nome campo filtro data_ult_mod non valido]
(Sol. 70130)

V2.1: DALLA VERSIONE GESTIONALE 2023C - V842 - 84200

Nuove risorse

- referenti clienti e fornitori:
  - */webapi/risorse/referenti/clienti*
  - */webapi/risorse/referenti/fornitori*
- tabella zone clienti/fornitori: */webapi/risorse/dati-generali/zone-clienti-fornitori* solo GET della collection

- tabella banche clienti: */webapi/risorse/banche* GET e POST per la ricerca
- tabella categorie provvigioni clienti/fornitori: */webapi/risorse/dati-generali/categorie-provvigioni* solo GET della collection
- tabella formazione prezzi articoli: */webapi/risorse/dati-generali/categorie-prezzi* solo GET della collection
- tabella categorie sconti clienti/fornitori: */webapi/risorse/dati-generali/categorie-sconti* solo GET della collection
- tabella aspetto esteriore dei beni: */webapi/risorse/dati-generali/aspetto-esteriore-beni* solo GET della collection

Nuove funzionalità

- per le risorse sotto l'end-point *documenti* è possibile indicare in trasformazione (quindi solo per metodo POST) l'eventuale scorporo dell'iva (es. quando si trasforma da OX a BC). In questo caso è necessario specificare in query string il parametro booleano *gest_scorporo_iva* che può assumere valore *true/false*

Nuovi campi

Nell'end-point dei *documenti* (per qualsiasi tipo di documento) nelle righe è ora disponibile il metadato sulla data di ultima modifica come avviene per tutti gli altri end-point: *data_ult_mod*.

Modifiche sistemistiche

Modificata la gestione dei servizi WebAPI in caso di accesso esclusivo da parte del gestionale. Aggiornato il capitolo sulle modalità di accesso.

V2.0.1

Correzione refusi

- Nel manuale non è mai stato indicato che è possibile richiedere i dati anche in formato compresso *gzip*.

  Per ogni *http-request* è sufficiente specificare nell'header il campo *Accept-Encoding:gzip* come previsto dallo standard http. WebAPI in questo caso risponderà con dati compressi in formato gzip, sarà poi compito del client decomprimere i dati ricevuti. Vedi sezione dedicata.
- Aggiornato il capitolo dei documenti indicando come viene gestito l'ordinamento delle righe.

V2.0: DALLA VERSIONE GESTIONALE 2023B4 - V841D - 84104

Modifiche al protocollo

- Nel caso di trasformazione di documenti col metodo POST è ora sufficiente indicare i soli parametri necessari, tutti gli altri dati di riga vengono presi automaticamente dalla riga del documento di origine (come avveniva con la PUT). La trasformazione con PUT non è più ammessa. Vedi sezione dedicata
- Nel caso di revisione di qualsiasi end-point è possibile abilitare un controllo lato server (servizi – configurazioni - configurazione moduli - WebAPI) sulla data di ultima modifica in modo da mantenere coerenza tra i dati (vedi capitolo dedicato)
- È stato eliminato il limite di 100 record massimo sulle GET di liste. Ora WebAPI restituisce sempre tutto quello che è in grado di dare entro 30 secondi oppure entro i 50MB di payload: se una delle 2 condizioni non è soddisfatta inizierà a paginare indicando con *next* il prossimo record da cui partire. La presenza di *max* in query string è prioritario a patto che non si violino una delle 2 condizioni sopra citate (vedi sezione dedicata).

Nuove funzionalità

- Aggiunta la possibilità di copiare una riga di un documento semplicemente indicando l'indice della riga da cui copiare i dati mediante il nuovo campo array *idx_riga_copia* (vedi sezione dedicata)
- Nell'end-point *documenti/ordini-clienti* è stata aggiunta la possibilità di gestire anche i documenti di tipo OX
- Per tutte le risorse dell'end-point *documenti* è stato aggiunto un parametro in query-string valido per il solo metodo PUT che indica se modificare o meno solo i dati di testata: *solo_testata=true/false* (vedi capitolo dedicato).

Nuovi campi

- Nell'end-point *liste-prelievo* è stato aggiunto un nuovo campo di tipo array *gest_tipo_lav* in sola lettura. Può essere scritto esclusivamente da client Zerodo
- Nell'end-point *documenti* è stato aggiunto un nuovo campo *bloccato_in_mag* in sola lettura. Può essere scritto esclusivamente da client Zerodo

Modifiche al manuale

Creato un capitolo dedicato per l'entità documenti

V1.10: DALLA VERSIONE GESTIONALE 2023A2 - V840B - 84002

Correzione errori

Nella risorsa articoli era stato inserito due volte il campo *cod_struttura* e poteva assumere valori errati. È stato corretto uno dei due nomi in *rif_struttura* che identifica il riferimento alla struttura nel caso di articoli di tipo "B".

Nuove risorse

È stata inserita una nuova risorsa per ottenere i progressivi di tutti gli articoli: *progressivi-articoli*

La risorsa è accessibile solo tramite metodo GET e restituisce tutti i progressivi per tutti gli articoli del magazzino specificato in header, oppure di tutti i magazzini se in header è impostato *Magazzino=0.*

Per i dettagli sull'end-point si veda l'HELP in linea (GET /risorse/help?extended=true)

Nuovi campi

In tutti i documenti di magazzino (ordini e movimenti) è stato esposto il campo con il riferimento alla tabella dell'aspetto esteriore dei beni. Il campo si chiama *asp_est_beni:* formato scalare per le collection e formato array per le singole risorse. Per i dettagli fare riferimento all'help in linea (es. GET documenti/ordini-clienti?info=true)

V1.9.2: DALLA VERSIONE GESTIONALE 2023A1 - V840A - 84001

Correzione refusi

Aggiunti maggiori dettagli nella sezione relativa alla gestione del parametro encoding

Correzione errori

- Se il parametro encoding era utilizzato in un path in cui il codice da decodificare era in mezzo al path e non alla fine (es. immagini articoli) la decodifica andava in errore.
- Nella risorsa *documenti/lavorazione/bolle* per la creazione/modifica di un nuovo documento è richiesto come parametro nel body *nr_rif_impegni* invece di *nr_rif_crea.*

V1.9.1: DALLA VERSIONE GESTIONALE 2023A - V840 - 84000

Nuovi campi

Nella risorsa *documenti/lavorazione/bolle/<codice_bolla>* è stato aggiunto il campo *descr_um*: Descrizione unità di misura (sola lettura)

Modifiche al protocollo

Al fine di gestire nel path della risorsa anche caratteri speciali come SLASH "/" e BACKSLASH "\", è stata inserita una nuova chiave in query string (*encoding*) per gestire l'encoding di questi caratteri. Per i dettagli si veda la sezione dedicata.

V1.9: DALLA VERSIONE GESTIONALE 2022J3 - V831C - 83103

Correzione refusi

- Negli esempi di trasformazione dei documenti sono stati corretti i valori sugli *id_riga*
- Nell'esempio dello sviluppo distinta base il campo con il codice articolo padre era errato (codice_**articolo**_padre): è stato corretto in codice_**art**_padre
- Inserito nei titoli del changelog anche la versione_prodotto reperibile da WebAPI facendo una request sull'end-point *dati-generali/installazione* (es.83103)

Correzione errori

Per le installazioni locali su macchine Windows la Java Virtual Machine su cui gira il Server Tomcat per la gestione del servizio WebAPI, aveva il charset di default impostato come *Windows-1252.* Questo comportava che le letture (GET) con WebAPI venivano correttamente codificate in *UTF-8*, ma le scritture (PUT-POST) venivano gestite in *Windows-1252* e quindi in fase di parsing del JSON in ingresso, nel caso di presenza di caratteri speciali, Mexal restituiva un errore sulla codifica.

Nuovi campi

- Nella risorsa *articoli* è stato aggiunto il campo *gest_macchinario* che, solo per gli articoli di tipo L, può assumere i valori S/N.
- Nella risorsa *movimenti_magazzino* sono stati esposti anche i seguenti ulteriori campi:
  - id_catsta_conto: Categoria statistica cliente/fornitore (cod_cat_sta in anagrafica cliente)
  - id_zona_conto: Zona cliente/fornitore (cod_zona in anagrafica cliente)
- Nella risorsa *movimenti_magazzino/righe* sono stati esposti anche i seguenti ulteriori campi:
  - costo_ult: costo ultimo allo scarico (-1 prende il costo in anagrafica articolo)
  - costo_std: costo standard allo scarico (-1 prende il costo in anagrafica articolo)
  - costo_med_pond: costo medio ponderato allo scarico (-1 prende il costo in anagrafica articolo)

V1.8: DALLA VERSIONE GESTIONALE 2022J - V831 - 83100

Nuovi servizi

Condizioni documento: vedi sezione dedicata

Nuovi campi

Nella risorsa *fornitori* è stato aggiunto il campo *assog_ivaxml: Condizioni generali - Import. op.passive con nat.esenz.IVA N2.1*

Nella risorsa *articoli* sono stati aggiunti due campi di sola lettura:

1. *dt_cos_ult_ues:* Data costo ultimo op. passive anno (sola lettura)
2. *dt_cos_ult_pes: Data costo ultimo anno precedente (sola lettura)*

Nelle risorse *liste-prelievo* ed *ubicazioni* sono state aggiunte informazioni su utente e data di ultima modifica:

1. *utente_ult_mod:* utente ultima modifica
2. *data_ult_mod:* data di ultima modifica

Correzioni

Risorse liste prelievo: aggiunti riferimento ai lotti sulla riga d'ordine (in precedenza id_lotto era sempre vuoto)

V1.7: DALLA VERSIONE GESTIONALE 2022I - V830 - 83000

Nuovi servizi

- Avanzamento di prodizione: vedi sezione dedicata.
- Spezza riga bolle di lavorazione: vedi sezione dedicata

Nuove risorse

- *mappa-articoli*: è stata esposta la lista con il dettaglio logistico di ogni articolo (es. ubicazione di default, magazzino, etc.). Sono disponibili i metodi GET per lista completa e POST per la ricerca. La risorsa è raggiungibile al seguente end-point: https://base_address/webapi/risorse/mappa-articoli
- *tipi-lotti-matricole:* è stata esposta in sola lettura la lista dei tipi di lotto/matricola. Sono disponibili i metodi GET per lista e singolo elemento e POST per la ricerca. La risorsa è raggiungibile al seguente end-point: https://base_address/webapi/risorse/ dati-generali/tipi-lotti-matricole
- *fasi-lavorazione*: è stata esposta la tabella con la descrizione delle fasi di lavorazione. Sono disponibili i metodi GET per lista completa e POST per la ricerca. La risorsa è raggiungibile al seguente end-point: https://base_address/webapi/risorse/dati-generali/mappa-articoli

Nuovi campi

- Aggiunti nella risorsa *liste-prelievo* i campi di tipo array *priorita* e *gest_residuo* modificabili per le sole righe d'ordine (non picking).
- Per ragioni di coerenza semantica sono stati cambiati i nomi dei campi per la gestione dei residui. Dato che la quantità da indicare non indica il residuo ma quella che andrà nel documento trasformato è stato sostituito res con trs:

  "nr_colli_trs"
  "quantita_trs "
  "qta_taglia_trs "
  "qta_lotto_trs "
  "nr_colli_lo_trs "
  "qta_tg_lo_trs"
- Nella risorsa ubicazioni, quando si richiede il dettaglio dei progressivi con *cod_art_progr,* è stata inserita la possibilità di escludere o meno i lotti (per motivi prestazionali e di compatiblità col pregresso) utilizzando in query string la variabile dettlotti=true/false
- Aggiunte le variabili di sola lettura per i totali del documento in tutte le risorse sotto il contenitore *documenti:*
  - tot_doc_validi: indica se i valori specificati nei totali sono validi
  - tot_iva
  - tot_documento
  - tot_doc_pagare

Correzioni

Aggiunti esempi nella trasformazione dei documenti: casi con lotti

V1.6: DALLA VERSIONE GESTIONALE 2022G - V824 - 82400

Correzione anomalie

Esportato il campo "*descr_art_ext*" anche per la risorsa *movimenti-magazzino/righe (Sol.66278)*

Nuove funzionalità

Trasformazione documenti (vedi sezione dedicata)

Nuove variabili

"nr_colli_res": "Numero colli residuo"
"quantita_res": "Quantita' / Peso Lordo residuo"
"qta_taglia_res": "Quantita' per taglia residuo"
"qta_lotto_res": "Lotti - Quantita' residuo"
"nr_colli_lo_res": "Lotti - Colli residuo"
"qta_tg_lo_res":"Lotti - Quantita' per taglia residuo"

Nuove risorse

- Liste-prelievo (vedi sezione dedicata)
- Giacenze articoli: nella risorsa articoli è stata aggiunta l'entità progressivi. Facendo una GET su questa entità verranno restituite tutte le giacenze dell'articolo selezionato. *Es. GET base_address/articoli/COD_ART/progressivi*
  Verranno restituiti array di n record, e per ogni record è indicata la quantità di giacenza (inventario, carico, scarico) il magazzino, l'eventuale ubicazione.
- *Alias-articoli: https://base_address/webapi/risorse/**alias-articoli** (solo lista e ricerca)*
- *Imballi: https://base_address/webapi/risorse/dati-generali/**imballi***

V1.5.2: DALLA VERSIONE GESTIONALE 2022D - V821 - 82100

Nuovi campi

- Nella risorsa *dati-generali/particolarita* è stato aggiunto il campo ***descr_art_cli**: Descrizione articolo del Cli/For se "tipo_part" = 'A'*
- Nella risorsa *ubicazioni* è stato aggiunto il campo ***percorso***: *Codice percorso utilizzato per creare un percorso all'interno del magazzino*
- Nella risorsa *documenti/movimenti-magazzino* sono stati aggiunti i seguenti campi :

  Campi di testata

```
{
        "nome": "rif_nr_bl",
        "descrizione": "Numero di riferimento bolla di lavoro",
        "tipo": "Numerico"
},
{
        "nome": "rif_nr_sbl",
        "descrizione": "Numero di riferimento sottobolla di lavoro",
        "tipo": "Numerico"
},
{
        "nome": "rif_nr_fs_lav",
        "descrizione": "Numero di riferimento fase di lavorazione",
        "tipo": "Numerico"
}
```

  Campi di riga (array)

```
{
        "nome": "cod_rif_pf_bl",
        "descrizione": "riferimento PF della BL; solo produzione",
        "tipo": "Numerico",
        "dimensione_array": "dinamico"
},
{
        "nome": "dt_doc_bl",
        "descrizione": "data documento della BL; solo produzione",
        "tipo": "Alfanumerico",
        "dimensione_array": "dinamico"
},
```

```
{
        "nome": "nr_bl",
        "descrizione": "numero BL; solo produzione",
        "tipo": "Numerico",
        "dimensione_array": "dinamico"
},
{
        "nome": "nr_sbl",
        "descrizione": "sotto BL; solo produzione",
        "tipo": "Numerico",
        "dimensione_array": "dinamico"
},
{
        "nome": "prog_fase_bl",
        "descrizione": "fase BL; solo produzione",
        "tipo": "Numerico",
        "dimensione_array": "dinamico"
}
```


Correzione refusi

- Nelle risorse *documenti/ordini documenti/preventivi documenti/matrici* il campo *id_righe_dba* è stato cambiato in *id_rec_dba_dbvo* ma non documentato
- Il Content-type è opzionale esclusivamente per in metodi GET e DELETE per i metodi POST e PUT è obbligatorio

V1.5.1: DALLA VERSIONE GESTIONALE 2022C2 - V820B - 82002

Correzione refusi

Nel servizio *sviluppo_distinta_base* sono stati corretti i nomi di 2 variabili:

escludi_da_deposito (mancava la **l**)
escludi_da_scarico (mancava la **l**)

Nell'entità *dati-generali/particolarita* è stato corretto il nome di una variabile:

erano stati erroneamente dichiarati due campi con lo stesso nome (cod_articolo), è stato lasciato *cod_articolo* per indicare il codice dell'articolo mentre è stato aggiunto *cod_art_cli* per indicare il codice articolo associato al cliente/fornitore

V1.5: DALLA VERSIONE GESTIONALE 2022C1 - V820A - 82001

Nuove risorse

Archivio Lotti: aggiunti relativi riferimenti all'id del lotto anche nelle risorse che lo referenziano

Archivio Ubicazioni (vedi sezione dedicata): aggiunti relativi riferimenti all'id dell'ubicazione anche nelle risorse che la referenziano

Nuovi campi

Aggiunto campo **tp_articolo** (tipologia articolo) nella risorsa degli impegni e nel servizio per lo sviluppo della distinta base *sviluppo_distinta_base*

V1.4: DALLA VERSIONE GESTIONALE 2022B4 - V819D - 81904

Nuove variabili

Nella risorsa *articoli* (sia lista che singola entità) sono state aggiunte le seguenti variabili che indicano la presenza o meno di immagini (S/N):

"img_art_disp": indica se l'immagine articolo è presente

"img_cat_disp": indica se l'immagine di catalogo è presente

"img_icona_disp": indica se l'icona è presente

V1.3: DALLA VERSIONE GESTIONALE 2022B - V819 - 81900

Correzione refusi di tipi di dato tra liste e singole risorse:

A=alfanumerico

F=float

N=numerico

Clienti / Fornitori

"gest_rit_acc" Impostato tipo 'A' S/N in get/put

"alq_regio_sport" Impostato tipo 'F' in lista

"alq_comun_sport" Impostato tipo 'F' in lista

"fido_migliaia" In lista rimane solo "fido" mentre in singola entità "fido" e "fido_migliaia"

"anag_dt_mod" Impostato tipo 'A' in lista

Articoli

"larghezza_ubic" Impostato tipo 'F' in get/put

"altezza_ubic" Impostato tipo 'F' in get/put

"profondita_ubic" Impostato tipo 'F' in get/put

"peso_ubicazione" Impostato tipo 'F' in get/put

"cod_srv_intra" Impostato tipo 'A' numero formattato con '0' a sx in get/put come in lista

Ordini / Preventivi / Matrici

"tp_cambio_vlt" Impostato tipo 'A' S/N in get/put

"tp_cambio_euro" Impostato tipo 'A' S/N in get/put

"gest_mydb_testa" Impostato tipo 'A' S/N in get/put

"gest_mydb_riga" Impostato tipo 'A' S/N in get/put

Movimenti di magazzino

"tp_cambio_vlt" Impostato tipo 'A' S/N in get/put

"tp_cambio_euro" Impostato tipo 'A' S/N in get/put

"gest_mydb_testa" Impostato tipo 'A' S/N in get/put

“tipo_stato_riga” Impostato tipo 'A' in lista

"gest_mydb_riga" Impostato tipo 'A' S/N in get/put

V1.2: DALLA VERSIONE GESTIONALE 2022A - V818 - 81800

Correzione refusi:

- Sviluppo distinta base: quantità obbligatoria

Modifiche al protocollo:

- Nuove risorse:
  - Fasi distinte base
  - Archivi MyDB
  - Aggiunta la risorsa “conti” oltre ai clienti e fornitori
- Nuova nomenclatura delle variabili: per tutti i dettagli usare il comando HELP e info=true. Per la conversione tra vecchia e nuova nomenclatura vedere Allegato A
- Corretta sintassi filtri per gestire array a due livelli
- Aggiunti i campi *id_riga_orig* e *cat_annul_riga* per i documenti di magazzino e degli ordini

V1.1 : DALLA VERSIONE GESTIONALE 2021I - V816 - 81600

Modifiche al protocollo:

- Aggiunto il parametro opzionale Magazzino nell’header della richiesta per permettere di specificare il magazzino di default
- Le liste sono state arricchite di ulteriori dati e gli array possono arrivare fino a 2 livelli invece che 1 solo livello
- Cambiato path del comando HELP e aggiunto filtro *extended* in query string
- Aggiunte nuove risorse: usare il comando HELP per tutti i dettagli
- Cambiato il path di alcune risorse: usare il comando HELP per tutti i dettagli
- Aggiunto limite al numero massimo di utenti appartenenti al gruppo WebAPI: 3
- Aggiunto servizio per lo sviluppo della distinta base
- Eliminata documentazione sull’accesso alle risorse tramite il canale *servizi*: si consiglia di utilizzare sempre l’accesso con logica REST sul path “risorse”
- Inserita entità allegati: in questa versione solo lettura immagini articoli
- Non è più necessario gestire il cookie per installazioni Live
- Nell’entità di tipo ricerca aggiunta la condizione “contiene”

Nota:

- nei prossimi rilasci di WebAPI i nomi delle chiavi dei JSON verranno cambiati gestendo un massimo di 15 caratteri. In questo modo i nomi degli elementi risulteranno più parlanti e più aderenti ad una logica RESTful. Nel caso siano già

state fatte applicazioni con la nomenclatura attuale, di volta in volta verranno rilasciate delle tabelle di correlazione tra i vecchi ed i nuovi nomi dei vari archivi aggiornati. In ogni caso per i dettagli sui nuovi nomi dei campi si può usare il metodo GET con il parametro *info=true* in query string.

Correzione refusi:

- esec_collage_server_remoto: era stato erroneamente indicato esegui_collage_server_remoto