TSRR

IFAC-TSRR vol. 14 (2024) 1-11

Relazione tecnica su: “Procedura di controllo a campione sulle dichiarazioni sostitutive rilasciate dagli operatori economici nell’ambito degli affidamenti diretti di servizi e forniture” (art.50 del Codice dei contratti pubblici d.lgs 36/2023)

E. Marzi [1], A. Agostini [1]

([1]) "Nello Carrara" Institute of Applied Physics, CNR Florence Research Area, Via Madonna del Piano 10, 50019 Sesto Fiorentino (FI), Italy

 ISSN 2035-5831

1 - Introduzione

Il presente documento descrive le specifiche tecniche per lo sviluppo di un'applicazione software in PHP che implementi un sistema di controlli a campione sulle dichiarazioni sostitutive rilasciate dagli operatori economici nell'ambito di affidamenti diretti di servizi e forniture di importo inferiore a 40.000 Euro, ai sensi dell'art. 50, comma 1, lettera b) del Codice dei contratti pubblici di cui al d.lgs. 36/2023.

2- Requisiti funzionali richiesti

L'applicazione deve essere in grado di:

1. Importare un elenco di affidamenti diretti di servizi e forniture infra 40.000 Euro effettuati in un determinato periodo di riferimento (semestre o quadrimestre solare).
2. Suddividere gli affidamenti in tre fasce di valore, al netto dell'IVA: inferiore a 5.000 Euro, compresa tra 5.000 e 20.000 Euro, compresa tra 20.000 e 40.000 Euro.
3. Calcolare la numerosità del campione da sottoporre a verifica per ciascuna fascia di valore, applicando le percentuali indicate nella Tabella 1 della Circolare CNR 33/2023 (10%, 15% e 20% rispettivamente).
4. Estrarre in modo casuale e imparziale i campioni da sottoporre a verifica per ciascuna fascia di valore.
5. Generare un report contenente l'elenco degli affidamenti estratti per le verifiche.
6. Inviare una notifica via mail dell'estrazione

In aggiunta ai requisiti richiesti è stata implementata la possibilità di:

1. Inviare un messaggio nella chat di istituto a cui sono iscritti i RUP
2. Caricare il file con le estrazioni sull'applicativo NextCloud[1] di IFAC

3- Installazione

Il software, scritto in PHP, è accompagnato da una documentazione completa che descrive l'installazione, la configurazione e l'utilizzo dell'applicazione.

Assicurarsi che sulla macchina siano installati PHP (versione 7.3+) e MYSQL/MARIADB (Ver 15.1 Distrib 10.3.28-MariaDB).

Aprire il terminale e navigare nella directory del progetto. Inizializzare un nuovo progetto Composer[2] eseguendo il seguente comando:

```
composer init
```
(1)

Aprire il file composer.json in un editor di testo e aggiungere le seguenti dipendenze nell'oggetto require

```
"require": {
   "php": "^7.3",
   "phpoffice/phpword": "dev-master",
   "phpmailer/phpmailer": "^6.9.1",
   "voku/simple-mysqli": "8.*",
   "vlucas/phpdotenv": "^5.6",
   "dompdf/dompdf": "^2.0"
},
```
(2)

Salvare il file composer.json e tornare al terminale. Eseguire il seguente comando per installare le dipendenze specificate:

```
composer install
```
(3)

ATTENZIONE:
Per rendere la libreria voku/mysqli[3] compatibile con PHP 8.1 e successive, è necessario modificare la definizione della funzione "*execute()*" contenuta nel file "*vendor/voku/simple-mysqli/src/voku/db/Prepare.php*" alla riga 197

Da così:

```
public function execute()
```
(4)

A così:

```
public function execute(?array $params = null)
```
(5)

4 - Preparazione della base dati

La base dati su cui è stata implementata la procedura, il database "Ordini", contiene gli ordini di acquisto nella tabella "*Tabella_acquisti*". Per esigenze infrastrutturali, ogni anno il database viene archiviato e rinominato in "*Ordini[anno_precedente]*", così, per esempio, nel 2024 avremo un database "Ordini" e un database "Ordini23" contenente gli affidamenti dell'anno precedente.

Poichè la procedura di sorteggio a campione potrebbe prevedere il sorteggio di ordini di acquisto a cavallo di due anni distinti, è stato necessario creare una "*stored procedure*" che unisse la "*Tabella_acquisti*" di "Ordini" e quella di "*Ordini[anno_precedente]*" in un'unica tabella.

Di seguito il codice da eseguire sulla base dati per creare la "*stored procedure*":

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `unionOrdini`()
BEGIN
 DECLARE current_year INT;
 DECLARE previous_year INT;
 DECLARE previous_year_short INT;

 SET current_year = YEAR(SYSDATE());
 SET previous_year = current_year - 1;
 SET previous_year_short = MOD(previous_year, 100);
 SET @campi = "descrizione, richiedente, rup, COALESCE(consip,0) as consip, consipUso,
mepa, mepaUso, acquisto, aqgia, stato";
 SET @query = CONCAT(
  "CREATE TEMPORARY TABLE temp_table AS SELECT * FROM (",
  "SELECT CONCAT(",current_year,",id) as idU, ", @campi, " FROM Ordini.Tabella_acquisti
",
  "UNION ALL SELECT CONCAT(",previous_year,",id) as idU, ", @campi, " FROM Ordini",
previous_year_short, ".Tabella_acquisti) AS tt;"
 );

 PREPARE stmt FROM @query;
 EXECUTE stmt;
 DEALLOCATE PREPARE stmt;
END$$
DELIMITER ;
```
(6)

La procedura crea una tabella temporanea *temp_table* che combina i dati delle tabelle degli acquisti dell'anno corrente e dell'anno precedente, aggiungendo una colonna idU che identifica in modo univoco

ogni riga con l'anno e l'ID originale. Questa tabella temporanea verrà utilizzata per ulteriori elaborazioni e query sui dati combinati.

4.1 - La procedura nel dettaglio

1. *DELIMITER $$* cambia il delimitatore delle istruzioni SQL da "*;*" a "*$$*" per consentire l'utilizzo del *punto e virgola* all'interno della procedura.
2. *CREATE DEFINER=\root@localhost PROCEDURE unionOrdini()* crea una nuova procedura SQL chiamata *unionOrdini* con la definizione di utente root sull'host localhost.
3. *BEGIN* inizia il blocco della procedura.
4. *DECLARE* dichiara tre variabili intere: *current_year*, *previous_year* e *previous_year_short*.
5. *SET current_year = YEAR(SYSDATE());* assegna l'anno corrente alla variabile *current_year*.
6. *SET previous_year = current_year - 1;* calcola l'anno precedente sottraendo 1 da *current_year* e lo assegna a *previous_year*.
7. *SET previous_year_short = MOD(previous_year, 100);* calcola gli ultimi due digit dell'anno precedente utilizzando la funzione MOD e li assegna a *previous_year_short*.
8. *SET @campi = "descrizione, richiedente, rup, COALESCE(consip,0) as consip, consipUso, mepa, mepaUso, acquisto, aqgia, stato";* crea una variabile @campi che contiene un elenco di colonne da selezionare dalle tabelle di acquisti.
9. *SET @query = CONCAT(...)* costruisce una stringa SQL dinamica @*query* che creerà la tabella temporanea *temp_table* unendo i dati delle tabelle di acquisti dell'anno corrente (Ordini.Tabella_acquisti) e dell'anno precedente (Ordini<previous_year_short>.Tabella_acquisti).
   La clausola *SELECT ** seleziona tutte le colonne specificate in @*campi*.
   La clausola *FROM ()* crea una subquery che unisce i dati delle due tabelle di acquisti utilizzando *UNION ALL*.
   La funzione *CONCAT(<current_year>,id)* concatena l'anno corrente con l'ID originale di ogni riga per creare una nuova colonna *idU* che identificherà in modo univoco le righe.
10. *PREPARE stmt FROM @query;* prepara un'istruzione SQL a partire dalla stringa @*query*.
11. *EXECUTE stmt;* esegue l'istruzione SQL preparata, creando effettivamente la tabella temporanea *temp_table*.
12. *DEALLOCATE PREPARE stmt;* dealloca le risorse utilizzate dall'istruzione SQL preparata.
13. *END$$* termina il blocco della procedura.
14. *DELIMITER ;* ripristina il delimitatore delle istruzioni SQL al valore predefinito ;

5 - Descrizione delle funzionalità

L'applicazione, sviluppata in PHP, si occupa di estrarre un campione casuale di affidamenti diretti per fornitori basandosi su fasce di importo prestabilite:

5.1 -Classe CampioneDichiarazioniSostitutive:

Questa è la classe principale dell'applicazione.

1. Nella costruzione dell'oggetto, vengono caricate le variabili d'ambiente dal file *.env*[4] e stabilita la connessione al database.

2. La classe definisce le fasce di importo per gli affidamenti (*_affidamentiFasce*) e i campi da visualizzare nel report (*_fieldsFilter*).
3. Il metodo *creaTabellaTempData()* esegue la *stored procedure* descritta in precedenza che crea una tabella temporanea *temp_table* combinando i dati degli acquisti dell'anno corrente e dell'anno precedente.
4. Il metodo *suddividiAffidamentiFasce()* esegue una query per suddividere gli affidamenti nelle fasce di importo definite.
5. Il metodo *estraiCampioniDaVerificare()* estrae un campione casuale di affidamenti da verificare per ogni fascia di importo, utilizzando percentuali diverse a seconda della fascia.
6. Il metodo *estraiCampione()* è il metodo principale che avvia l'intero processo di estrazione del campione, generazione del report e notifiche.
7. Il metodo *generaReport()* crea un'istanza della classe *WordReportGenerator* e genera il report in formato DOCX.
8. Il metodo *notificaEstrazione()* invia una notifica tramite email con il report allegato.
9. Il metodo *copiaFileInCloud()* carica il report generato su NextCloud e invia una notifica nella chat di NextCloud di IFAC.

5.2 Classe WordReportGenerator:

Questa classe si occupa della generazione del report in formato DOCX.

1. Il metodo *generaReportWord()* utilizza la libreria PHPWord[54] per generare il report a partire da un template DOCX. Il report contiene tabelle suddivise per fasce di importo, con gli affidamenti evidenziati se fanno parte del campione estratto.
2. Il metodo *salvaFileDoc()* salva il report generato come file DOCX sul file system.
3. Il metodo *downloadFile()* consente di scaricare direttamente il report generato.
4. Il metodo *esportaPdf()* converte il report DOCX in formato PDF utilizzando la libreria dompdf[6].

5.3 Classe NotificaEstrazione

Questa classe si occupa dell'invio della notifica di estrazione tramite email con il report in allegato.

1. Utilizza la libreria PHPMailer[7] per inviare le email.
2. Le impostazioni di connessione al server di posta vengono caricate dal file *.env*.

5.4 Classe NextcloudFileUploader

Questa classe si occupa del caricamento del report generato su NextCloud.

1. Utilizza la libreria cURL per caricare il file sul server WebDAV di NextCloud.
2. Dopo il caricamento, invia una notifica nella chat di NextCloud utilizzando l'endpoint fornito.

6 - Flusso operativo

Viene creata un'istanza della classe *CampioneDichiarazioniSostitutive* a cui vengono passate le date di riferimento del periodo di riferimento da cui estrarre il campione.

1. Il metodo estraiCampione() viene chiamato per avviare il processo di estrazione del campione.
2. Viene creata la tabella temporanea necessaria per l'elaborazione dei dati.
3. Gli affidamenti vengono suddivisi in fasce di importo.
4. Viene estratto un campione casuale di affidamenti da verificare per ogni fascia.
5. Viene generato un report in formato DOCX utilizzando la classe *WordReportGenerator*.
6. Viene inviata una notifica tramite email con il report allegato utilizzando la classe *NotificaEstrazione*.

Fig. 1 – La mail di notifica di avvenuta estrazione

7. Il report viene caricato sull'istanza NextCloud di istituto e viene inviata una notifica nella chat di NextCloud utilizzando la classe *NextcloudFileUploader*.

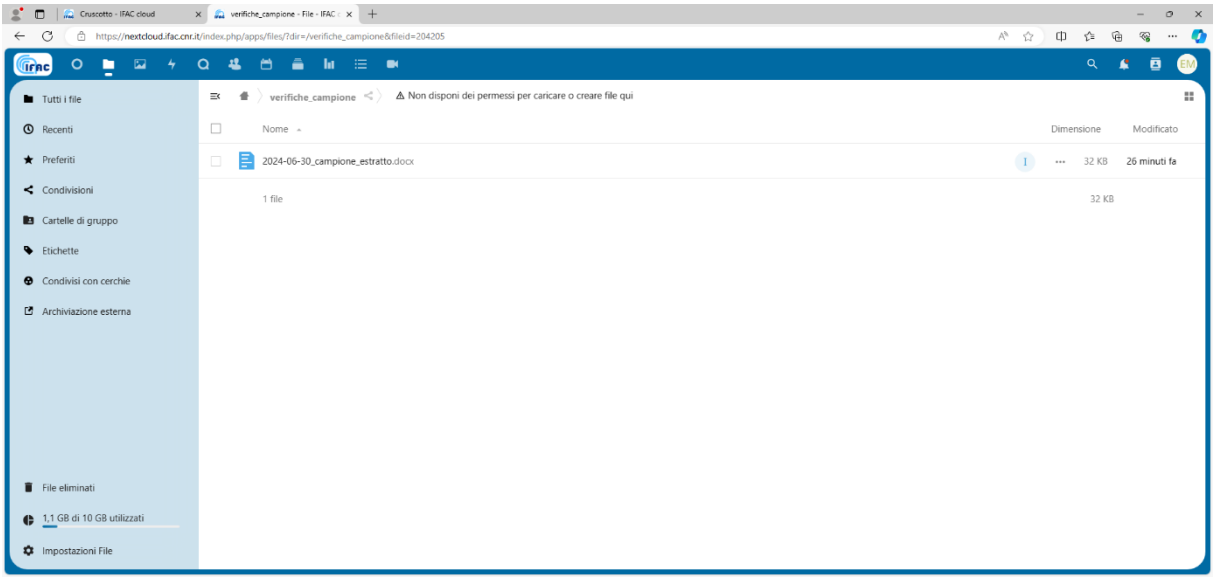


Fig. 2 – La directory condivisa su NextCloud contenente il file dell'estrazione

7 - Estrazione del campione casuale

Il metodo *suddividiAffidamentiFasce()* è responsabile della suddivisione degli affidamenti nelle rispettive fasce di importo. Di seguito la descrizione dettagliata del suo funzionamento:

```
private function suddividiAffidamentiFasce()                                    (7)
{
   $query = "SELECT TA.idU, AD.cig, AD.oggetto, AD.rup, AD.dittaVincitrice, AD.importo
         FROM Pubblicazioni.AvvisiAffidamentiDiretti AS AD
         JOIN $this->tableName AS TA ON TA.idU=CONCAT(AD.annoAcquisto, AD.idAcquisto)
          AND TA.consip <> 1
         WHERE AD.data BETWEEN ? AND ?
         AND AD.CUP NOT IN (
          SELECT CUP
          FROM Contratti.Anagrafica
          WHERE Numero_Contratto LIKE '%PNRR%'
         )";

   $stmt = $this->_connection->prepare($query);
   $stmt->bind_param('ss', $this->_periodoDiRiferimento['start'], $this-
>_periodoDiRiferimento['end']);
   $result = $stmt->execute();

   if($result !=false){
      while ($row = $result->fetchArray()) {
         $importo = $row['importo'];
         if ($importo < 5000) {
            $this->_affidamentiFasce['inferiore_5000'][] = $this->filtraCampi($row);
         } elseif ($importo >= 5000 && $importo < 20000) {
            $this->_affidamentiFasce['tra_5000_20000'][] = $this->filtraCampi($row);
         } elseif ($importo >= 20000 && $importo < 40000) {
            $this->_affidamentiFasce['tra_20000_40000'][] = $this->filtraCampi($row);
         }
      }
   }
}
```

1. Il metodo inizia costruendo una query SQL per recuperare gli affidamenti dalla tabella *Pubblicazioni.AvvisiAffidamentiDiretti* all'interno di un determinato periodo di riferimento ($this->_periodoDiRiferimento).
2. La query effettua una JOIN con la tabella temporanea ($this->tableName) creata in precedenza dal metodo *creaTabellaTempData()*. Questa tabella combina i dati degli acquisti dell'anno corrente e dell'anno precedente.
3. La condizione *TA.consip <> 1* esclude gli affidamenti gestiti attraverso Consip (Concessionaria Servizi Informatici Pubblici).
4. Inoltre, la query esclude gli affidamenti relativi ai contratti *PNRR* (Piano Nazionale di Ripresa e Resilienza) controllando il numero di contratto nella tabella Contratti.Anagrafica.
5. Per ogni riga di risultato, il metodo controlla l'importo dell'affidamento ($row['importo']) e lo inserisce nell'array corrispondente dell'array associativo $this->_affidamentiFasce in base alla fascia di importo:
   - Se l'importo è inferiore a 5000€, viene aggiunto all'array $this->_affidamentiFasce['inferiore_5000']
   - Se l'importo è compreso tra 5000€ e 20000€ viene aggiunto all'array $this->_affidamentiFasce['tra_5000_20000']
   - Se l'importo è compreso tra 20000€ e 40000€, viene aggiunto all'array $this->_affidamentiFasce['tra_20000_40000']

Alla fine dell'esecuzione del metodo *suddividiAffidamentiFasce()*, l'array associativo $this->_affidamentiFasce conterrà tutti gli affidamenti suddivisi nelle rispettive fasce di importo, pronto per essere utilizzato dai metodi successivi per l'estrazione del campione e la generazione del report.

Il metodo *estraiCampioniDaVerificare()* è responsabile dell'estrazione del campione casuale di affidamenti da verificare per ogni fascia di importo. Di seguito una descrizione dettagliata del funzionamento di questo metodo:

```
private function estraiCampioniDaVerificare()                                    (8)
{
   $this->_campioneEstratto = array();

   // Fascia inferiore a 5000 Euro
   $numerositaCampione = ceil(count($this->_affidamentiFasce['inferiore_5000']) * 0.1);
   $this->_campioneEstratto['inferiore_5000'] = $this->estraiCampioneCasuale(
         $this->_affidamentiFasce['inferiore_5000'], $numerositaCampione);

   // Fascia tra 5000 e 20000 Euro
   $numerositaCampione = ceil(count($this->_affidamentiFasce['tra_5000_20000']) * 0.15);
   $this->_campioneEstratto['tra_5000_20000'] = $this->estraiCampioneCasuale(
         $this->_affidamentiFasce['tra_5000_20000'], $numerositaCampione);

   // Fascia tra 20000 e 40000 Euro
   $numerositaCampione = ceil(count($this->_affidamentiFasce['tra_20000_40000']) * 0.2);
   $this->_campioneEstratto['tra_20000_40000'] = $this->estraiCampioneCasuale(
         $this->_affidamentiFasce['tra_20000_40000'], $numerositaCampione);
}
```

1. Il metodo inizializza un array vuoto *$this->_campioneEstratto* che conterrà i campioni estratti per ogni fascia di importo.
2. Per ogni fascia di importo (inferiore_5000, tra_5000_20000, tra_20000_40000), viene calcolata la numerosità del campione moltiplicando il numero totale di affidamenti in quella fascia per una percentuale specifica:
   - Per la fascia inferiore_5000, la percentuale è del 10% (* 0.1)
   - Per la fascia tra_5000_20000, la percentuale è del 15% (* 0.15)
   - Per la fascia tra_20000_40000, la percentuale è del 20% (* 0.2). Il risultato viene arrotondato all'intero superiore utilizzando la funzione ceil()
3. Per ogni fascia, viene chiamato il metodo *estraiCampioneCasuale()* passando come argomenti l'array degli affidamenti per quella fascia ($this->_affidamentiFasce[fascia]) e la numerosità del campione calcolata precedentemente ($numerositaCampione).

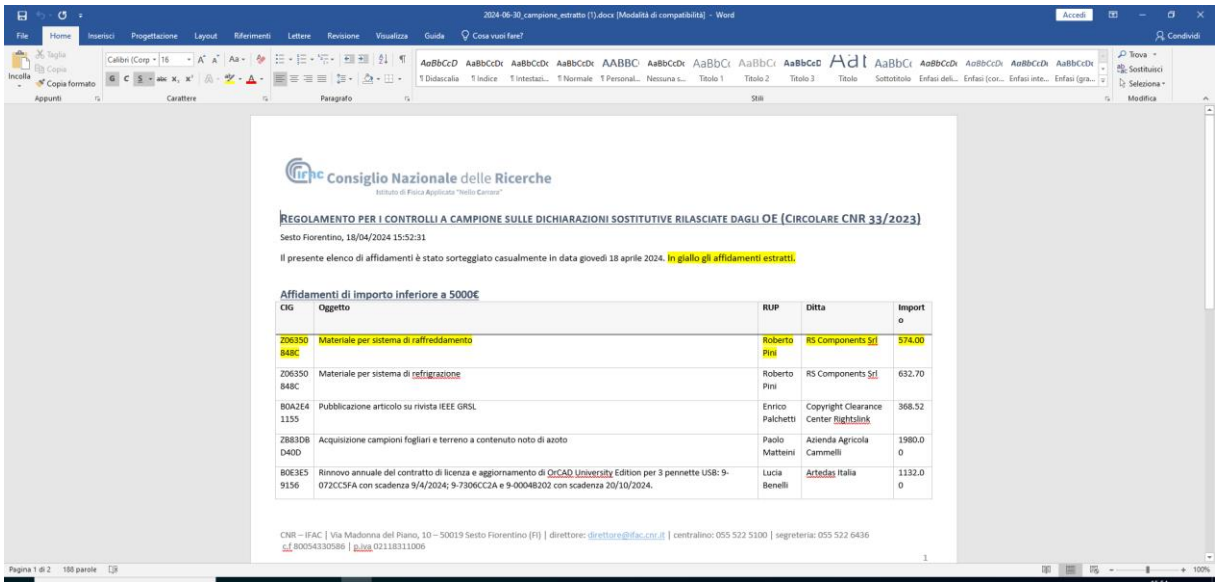
Il metodo *estraiCampioneCasuale()* richiamato da *estraiCampioniDaVerificare* è responsabile dell'estrazione effettiva del campione casuale

```
private function estraiCampioneCasuale(array $affidamenti, int $numerositaCampione)   (9)
{
   $campione = array();
   if(!empty($affidamenti)){
      $chiavi = array_rand($affidamenti, $numerositaCampione);
      if (!is_array($chiavi)) {
         $chiavi = array($chiavi);
      }
      foreach ($chiavi as $chiave) {
         $campione[] = $affidamenti[$chiave];
      }
   }
```

```
    return $campione;
}
```

1. Viene creato l'array vuoto *$campione* per contenere il campione estratto.
2. La funzione *array_rand()* viene utilizzata per estrarre in modo casuale un numero di chiavi "*$numerositaCampione*" dall'array $affidamenti. Queste chiavi rappresentano gli indici degli elementi da includere nel campione.
3. Se array_rand() restituisce un singolo valore anziché un array (nel caso in cui $numerositaCampione sia 1), il valore viene convertito in un array. Quindi, per ogni chiave estratta, l'elemento corrispondente dall'array *$affidamenti* viene aggiunto all'array *$campione*.
4. Il campione estratto per ogni fascia viene quindi restituito e memorizzato nell'array *$this->_campioneEstratto*, utilizzando la chiave corrispondente alla fascia di importo.

In questo modo, il metodo *estraiCampioniDaVerificare()* garantisce che per ogni fascia di importo venga estratto un campione casuale di affidamenti, con una percentuale di campionamento diversa a seconda della fascia. L'utilizzo della funzione *array_rand()* assicura che l'estrazione del campione avvenga in modo completamente casuale.



Consiglio Nazionale delle Ricerche

REGOLAMENTO PER I CONTROLLI A CAMPIONE SULLE DICHIARAZIONI SOSTITUTIVE RILASCIATE DAGLI OE (CIRCOLARE CNR 33/2023)

Sesto Fiorentino, 18/04/2024 15:52:31

Il presente elenco di affidamenti è stato sorteggiato casualmente in data giovedì 18 aprile 2024. In giallo gli affidamenti estratti.

Affidamenti di importo inferiore a 5000€

CIG	Oggetto	RUP	Ditta	Importo
Z06350848C	Materiale per sistema di raffreddamento	Roberto Pini	RS Components Srl	574.00
Z06350848C	Materiale per sistema di refrigrazione	Roberto Pini	RS Components Srl	632.70
B0A2E41155	Pubblicazione articolo su rivista IEEE GRSL	Enrico Palchetti	Copyright Clearance Center Rightslink	368.52
ZB83DBD40D	Acquisizione campioni fogliari e terreno a contenuto noto di azoto	Paolo Matteini	Azienda Agricola Cammelli	1980.00
B0E3E59156	Rinnovo annuale del contratto di licenza e aggiornamento di OrCAD University Edition per 3 pennette USB: 9-072CC5FA con scadenza 9/4/2024; 9-7306CC2A e 9-0004B202 con scadenza 20/10/2024.	Lucia Benelli	Artedas Italia	1132.00

CNR – IFAC | Via Madonna del Piano, 10 – 50019 Sesto Fiorentino (FI) | direttore: direttore@ifac.cnr.it | centralino: 055 522 5100 | segreteria: 055 522 6436
c.f. 80054330586 | p.iva 02118311006

Fig. 3 – Il file .docx contenente gli affidamenti. In giallo gli affidamenti estratti

8 - Attivazione

La procedura può essere lanciata manualmente da terminale eseguendo il comando

```
php estrai.php
```
(10)

oppure inserendola nel servizio di scheduling Cron

```
0 10 30 6,12 * /usr/bin/php [...path...]/estrai.php >> /root/verifiche/ReportVerifiche.log
```
(11)

Nell'esempio, l'estrazione viene eseguita alle 10 di mattina del 30 giugno e del 30 dicembre dell'anno in corso e l'esito inviato al file di log "ReportVerifiche".

Riferimenti

[1] Nextcloud - strumento di collaborazione sicuro per la sincronizzazione di file - https://nextcloud.com/it/
[2] Composer - gestore delle dipendenze per PHP - https://getcomposer.org/
[3] Koku Simple MySQLi Class - MySQL Abstraction Layer compatibile con PHP 7+ e PHP 8.0 che fornisce un'interazione semplice e sicura con il database - https://github.com/voku/simple-mysqli
[4] Dotenv - carica automaticamente le variabili di ambiente da file .env a getenv(), $_ENV e $_SERVER. - https://github.com/vlucas/phpdotenv
[5] PHPWord - una libreria scritta in puro PHP che fornisce un insieme di classi per scrivere e leggere da diversi formati di file - https://github.com/PHPOffice/PHPWord
[6] Dompdf - un convertitore da HTML a PDF - https://github.com/dompdf/dompdf
[7] PHPMailer – una libreria completa per la creazione e il trasferimento di e-mail per PHP - https://github.com/PHPMailer/PHPMailer

PHP è un linguaggio di scripting interpretato, originariamente concepito per la programmazione di pagine web dinamiche. L'interprete PHP è un software libero distribuito sotto la licenza PHP - https://www.php.net/

MySQL o Oracle MySQL è un relational database management system composto da un client a riga di comando e un server - https://www.mysql.com/it/
MariaDB è un RDBMS nato da un fork di MySQL creato dal programmatore originale di tale programma. Aperto ai contributi della comunità, l'area di sviluppo principale è lo storage engine Aria, precedentemente chiamato Maria da cui deriva MariaDB - https://mariadb.org/

Il codice sorgente dell'applicazione è disponibile sulla piattaforma condivisa BALTIG del CNR Sede Centrale accessibile con le credenziali SIPER. https://baltig.cnr.it

Indice