Diario di lavoro

|  |  |
| --- | --- |
| Luogo | Scuola d’arti mestieri Trevano |
| Data | 15.10.2020 |

|  |
| --- |
| Lavori svolti |
| Oggi ho aggiornato il metodo getDbLinkBackupFile models backup\_model perché non funzionava. In effetti se creavo nella cartella sources del progetto la cartella gestione\_backup (nome di un collegamento), salvavo nell’array files delle informazioni sbagliate (file che non non corrispondono ai file di backup). Quindi ho realizzato il seguente codice.  Immagine che contiene testo  Descrizione generata automaticamente  Nella prima condizione verifico che riesco ad aprire la directory del corrispettivo collegamento nel quale ci sono i file di backup. Poi creo un ciclo che permette di leggere il contenuto della directory. Con il metodo strpos verifico che il nome del file contiene la data corrente e l’estensione SQL. Se corrisponde lo salvo nell’array files alla prima poszione e termino il ciclo, altrimenti lo riempio con una stringa vuota.  Immagine che contiene testo  Descrizione generata automaticamente  Creo un cosice simile anche per i file log. In effetti nella prima condizione verifico che riesco ad aprire la directory log del corrispettivo collegamento. All’interno creao sempre un ciclo che legge il contenuto della directory e se trova il nome dell’elemento (file) contiene la data corrente e l’estensione txt, salvo nell’array files alla seconda posizione e esco dal ciclo. In più verifico lo stato del backup salvandolo nella terza posizione del array files. In effetti utilizzo il metodo statusBackup del models backup\_model.  Per evrificare che la classe funzionasse ho cacellato la cartella di backup del collegamento accesso in modo che nella views visualizza\_backup il segnale dello stato sia rosso.  Immagine che contiene testo  Descrizione generata automaticamente  Mi sono anche il sistema di backup che ho implementato runBackup non funzionava, perché non mi ha creato i backup i giornalieri. Quindi ho cercato su internet e ho trovato un sistema che permette di creare dei cronjob all’interno del proggetto. In effetti è un framework può essere facilmente integrato nel progetto o eseguito come uno scheduler di comandi autonomi.  Per istallare il framework bisogna utilizzare composer. Composer è un gestore di pacchetti a livello di applicazione per il linguaggio di programmazione PHP che fornisce un formato standard per la gestione delle dipendenze del software PHP e delle librerie richieste.  Per installare rapidamente Composer nella directory corrente bisogna eseguire il seguente script nel terminale. Per vedere come installare composer vedere il seguente link <https://getcomposer.org/download/>  Dopo che si abbia installato composer nella directory del proggetto (ho installato composer nella directory sources) posso eseguire il segunte comando php composer.phar require peppeocchi/php-cron-scheduler.  Quest’ultimo crea una cartella vendor nel quale ci sono tutte le librerie che permettono di fare funzionare il framework. Nel models backup\_models ho importato il file autoload.php che è un file nel quale viene gestista l’inclusione delle classi. Utilizzando l’operatore use posso fornire un alias alla classe nella quale viene implementato i cronjob.  Immagine che contiene testo  Descrizione generata automaticamente  La varriabile $scheduler è un oggetto di tipo Scheduler che permette di creare un cronjob. Con il metodo raw posso eseguire dei comandi di terminale come ad esempio mysqldump per eseguire i miei backup giornalieri. Il metodo at permette di definire lintervallo nel quale i backup devono essere eseguiti (simile a crontab). Per finire eseguo in un ciclo infinito (in modo che viene eseguito ogni volta) un cronjob. Quest’ultimo viene eseguito con il metodo run. |

|  |
| --- |
| Problemi riscontrati e soluzioni adottate |
| Oggi ho riscontrato dei problemi che mi venivano generati dal models backup\_model. In particolare dal metodo getDbLinkBackupFile che salvava anche dei file che non centravano con i backup. In effetti il primo codice che ho implementato mi teneva in conto dell’ultimo file che è presente nella directory. Ma non controllava che il file fosse un file SQL o txt. Ad esempio per il collegamento gestione\_directory, nella sua directory c’era un file .DS\_Store (.DS\_Store è un file nascosto con un formato proprietario creato da macOS per memorizzare attributi personalizzati di una cartella come la posizione delle icone o la scelta di un'immagine di sfondo. Cioè contiene metadati) che veniva mostrato nella views visualizza\_backup. Questo file non deve essere mostrato e per risolvere questo problema ho aggiunto un controllo che verifica il nome del file che deve corrispondere dump-<nome\_collegamento>-<data\_attuale>.sql. |

|  |
| --- |
| Punto della situazione rispetto alla pianificazione |
| Oggi volevo terminare il metodo che crea il pdf, ma mi sono reso conto del problema dei backup. In effetti il sistema non era efficace e non controllavo che il file fosse corretto. Quindi ho perso del tempo, perché oggi dovevo anche eseguire le email con PHPMailer. Per installare PHPMailer posso già utilizzare composer che ho installato nella cartella sources. |

|  |
| --- |
| Programma di massima per la prossima giornata di lavoro |
| Per la prossima giornata devo terminare il metodo che permette di crere un pdf. Cioè al interno del file devo salvare il contenuto dei vari collegamenti. Finito questa attività posso implementare con PHPMailer un sistema che invia le email quando c’è stato un problema con il backup. Sono le stesse attività che ho privisto oggi di realizzare ma a causa del sistema di backup che non funzionava, ho perso del tempo di lavoro. |