UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

Dipartimento di Scienze Fisiche,

Informatiche e Matematiche

Corso di laurea triennale di Informatica

Progetto e sviluppo portale web per la tracciabilità della produzione di articoli di un'azienda e terzisti

Cristian Marsili

Tesi di laurea triennale

Relatore:

Prof. Riccardo Martoglia

Anno Accademico 2021/2022

Ringraziamenti

Un ringraziamento al professor Riccardo Martoglia per il suo supporto e disponibilità.
Ringrazio anche tutta la mia famiglia e gli amici per il loro supporto anche nei momenti più difficili durante questi anni.

INDICE

Introduzione

Il progetto ha un focus orientato verso le aziende in crescita dove la gestione della catena di produzione diventa sempre più complessa, lunga e costosa per via del numero di ordini sempre più crescente sia in quantità che in tipologia.

Le scelte che ha un'azienda in questi casi sono due:

1) Aumentare il numero di macchinari e personale.
2) Affidarsi a terzisti che possono svolgere lo stesso lavoro con un costo a volte più contenuto e fisso.

Questo progetto si propone di risolvere uno dei problemi presenti nel secondo caso. Ovvero un'azienda di medio-grandi dimensioni con tanti terzisti che producono diversi articoli e il cliente vuole controllare i propri ordini. Questo controllo gli permette di sapere se il terzista sta producendo, cosa e se ha qualche problema.

Come azienda di riferimento dove provare e avere un riscontro in tempo reale sullo sviluppo per valutarne l'utilità e l'efficacia di questo strumento è stata scelta la Exena Srl, un'azienda di grandi dimensioni che ha una produzione sia interna che esterna di scarpe di sicurezza per il lavoro.

L'idea iniziale è stata quella di realizzare un servizio web su un server Cloud che ci permettesse di creare dei file PDF stampabili. Questi file al

loro interno contengono delle etichette con alcune informazioni sul prodotto e fornitore. Informazioni che sono salvate all'interno di un codice a barre (Ean13 oppure Code128) leggibile da un qualsiasi palmare o smartphone con sistema operativo Android. Queste informazioni tramite una app installata nel dispositivo verranno inviate al web server e scritte nel database del servizio web. L'utente finale potrà visualizzare tramite il suo portale cosa ha inviato al web server, confermare le informazioni inviate e stamparle. Questo sistema ci permette di effettuare un controllo molto semplice sulla produzione di un articolo.

Il servizio è stato creato utilizzando come strumenti la Suite Infinity che ci permette di avere un sito base da dove iniziare senza quindi perder tempo a sviluppare l'infrastruttura grafica in html e css.

Da questo sito base si è poi sviluppato il progetto utilizzando Java per il back-end e Javascript per il front-end del sito.

Per la gestione delle informazioni si è utilizzato un database con PostgreSQL e per il web hosting Tomcat 9.

Il tempo di sviluppo di questo progetto si è limitato a solamente tre mesi presso l'azienda Infor-ma Srl. di Civitanova Marche attiva nel settore informatico da oltre 20 anni specializzandosi nella produzione di software gestionali. Il focus principale di questa azienda è concentrato verso le attività di medio-grandi dimensioni in crescita nel settore calzaturiero.

Grazie alla esperienza accumulata negli anni nel settore è riuscita a formare una partnership di lunga durata con l'azienda Zucchetti permettendogli di avere accesso a strumenti di sviluppo di alto livello.

Durante lo svolgimento del periodo di tirocinio sono stato affiancato dal tutor Giovanni Pompili che mi ha aiutato soprattutto durante le prime fasi di analisi e sviluppo per orientarmi meglio nell'ambiente di lavoro nel mio breve periodo di permanenza nell'azienda.

Contenuti

La tesi si dividerà in due parti, nella prima verrà discusso l'ambiente di sviluppo con una breve presentazione di tutti i software utilizzati e il loro scopo di utilizzo. In questa prima parte si inizierà con una breve presentazione della Suite Infinity che comprende i due programmi SitePainter e PortalStudio utilizzati per lo sviluppo, poi si passerà al database postgresql e concludere con Tomcat che è il software utilizzato per poter rendere disponibile il sito web dall'esterno.

Nella seconda parte verrà mostrato lo Sviluppo del progetto e le conclusioni finali tratte dopo un breve periodo di osservazione del suo utilizzo da parte di un'azienda. La fase di Sviluppo inizierà con l'analisi con alcuni diagrammi uml sulle relazioni di alcune tabelle per poi passare alle interfacce e concludere con la reportistica. Prima della conclusione verrà mostrata l'implementazione dell'interfaccia per la stampa delle etichette.

PARTE I

AMBIENTE DI SVILUPPO

Capitolo 1

Tecnologie utilizzate

In questo capitolo verranno presentati i vari strumenti utilizzati per la realizzazione di questo progetto.

1.1 Suite Infinity

Sviluppata da un'azienda italiana di software gestionali chiamata Zucchetti.

Questa suite di programmi ci permette di gestire il front-end e il back-end del nostro web-service.

Sono stati utilizzati per questo progetto i seguenti strumenti della suite Infinity:

→ SitePainter per il back-end.

→ PortalStudio per il front-end.

La versione utilizzata è la 6.3.30/31.

Dalla Figura 1.1 possiamo vedere tutti i moduli e software che compongono questa suite.

Figura 1.1 – Suite Infinity

1.1.1 SitePainter

È il back-end del nostro servizio web dove vengono messi a disposizione diversi strumenti per lo sviluppo delle varie routine / funzioni scritte in Java, maschere per l'inserimento o visualizzazione dei dati e la struttura dati del database. È quindi la base del nostro web service.

Andando in ordine di sviluppo si è iniziati con la gestione del database.

"Un database è un insieme di informazioni (o dati) strutturate in genere e archiviate elettronicamente in un sistema informatico.[1]"

Questa gestione è affidata al Plan Editor.

1.1.2 Plan Editor

Il database è stato gestito da uno strumento chiamato Plan Editor che ci permette di creare tabelle Master/Detail, tabelle temporanee, relazioni tra tabelle e variabili globali in modo facile ed intuitivo utilizzando un'interfaccia grafica.

Nella Figura 1.2 a fine di questa sezione è possibile notare la presentazione dell'interfaccia allo sviluppatore con tutti gli oggetti utilizzati.

Le tabelle Master/Detail sono due tipi di tabelle più comunemente chiamate in gergo informatico Padre e Figlio che sono legate da una relazione su dei campi comuni. Il come poi verranno gestite le tabelle e i tipi di dato dipenderanno dal dbms utilizzato che in questo caso è PostgreSQL.

Un altro tipo di tabella utilizzata sono quelle temporanee che hanno una struttura dinamica che può cambiare nel tempo e sono legate all'utente e alla sessione in corso. Vengono cancellate periodicamente

per evitare l'allocazione inutile di risorse dopo un certo periodo di inutilizzo. Sono state utilizzate soprattutto come contenitori di appoggio per le informazioni utili per la stampa delle etichette.

All'interno del Plan Editor è possibile anche dichiarare delle variabili globali sempre legate alla sessione dell'utente questo è possibile farlo in un menù a parte dove definiamo il nome, tipo, valore di default e una

breve descrizione per ricordarci dove è possibile utilizzarlo a cosa fa riferimento quella variabile. Il menu è possibile trovarlo nella *Figura 1.2* nella barra degli strumenti.

Grazie a questo strumento lo sviluppo dell'analisi del progetto è stato notevolmente resa più facile e veloce. Un altro punto a suo favore è che ci permette tramite una interfaccia grafica di avere sempre un'idea generale di come vengono gestite le informazioni.

Le tabelle create con questo strumento avranno poi una struttura particolare nel database effettivo. Tutte le tabelle non temporanee avranno come suffisso "xxx" questo serve per indicare dei prototipi ovvero la struttura base della nostra tabella. Il suffisso verrà poi sostituito dal nome dell'azienda registrata nella piattaforma in modo tale da garantire ad ogni attività un proprio spazio nel database.

Invece le tabelle con suffisso cptmp sono temporanee e derivano da due tabelle con suffisso _data e _proto.

La *Figura 1.6* alla fine della Sezione 1.3 PostgreSQL presenta l'esempio di presentazione di alcune tabelle.

È importante tener conto di come vengano gestite effettivamente le tabelle perché durante le query è possibile avere risultati inaspettati se si scelgono le tabelle sbagliate, soprattutto se si utilizza il profilo utente errato.

Questa fase è molto importante e ha richiesto molto tempo per essere completata perché poi ci permetterà di sviluppare delle interfacce di prototipo da dove iniziare lo sviluppo.

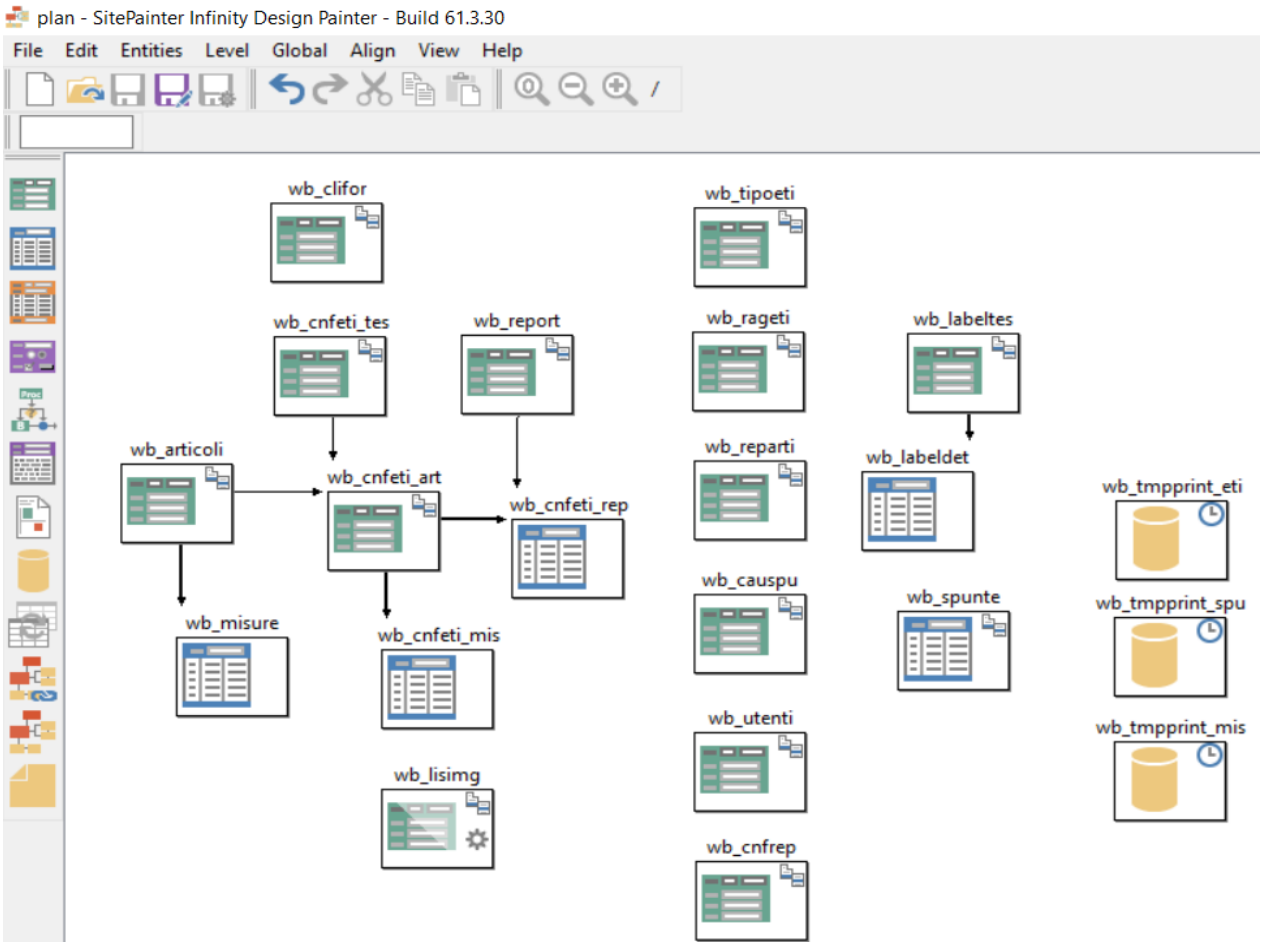


Figura 1.2 – Interfaccia PlanEditor

1.1.3 Maschere

Il secondo strumento utilizzato è quello delle maschere o interfacce utente, che seguono la struttura base di una o più tabelle. La maggior parte delle interfacce vengono generate automaticamente tramite un prototipo dopo il salvataggio e generazione della Plan.

Ogni maschera verrà generata con tutti i campi presenti nell'analisi e le sue relazioni tramite dei collegamenti che ci permettono o di visualizzare la seconda interfaccia tramite un popup dopo aver cliccato su un pulsante o di unirle con un oggetto chiamato "box" in modo da ottenere un'unica interfaccia. Un esempio di interfaccia composta è presente nella *Figura 1.3* a fine sezione.

È possibile, inoltre, porre delle condizioni di visualizzazione sulla seconda interfaccia in modo da ottenere in modo dinamico risultati differenti in base a delle situazioni o all'utente.

Nel progetto si è utilizzato molto la visualizzazione tramite box perché più semplice, meno fastidioso e intuitivo per l'utente.

Le funzionalità base offerte dal prototipo sono quelle di creazione, eliminazione, modifica e ricerca delle informazioni. Le prime tre interfacce condividono lo stesso layout modificabile mentre per l'ultima verrà rappresentata come una lista dentro una tabella.

È possibile, inoltre, creare interfacce vuote e collegare in seguito le tabelle che ci interessano sviluppando così un'interfaccia nuova da zero. Per il progetto è stata scelta quest'ultima via per le interfacce più importanti perché presentavano delle iterazioni molto particolari e più complesse che tramite la generazione di un semplice prototipo non è possibile gestire. E un'altra problematica dei prototipi è che se si modifica l'analisi è possibile che si debba ricreare anche l'interfaccia e di conseguenza perdere le modifiche fatte.

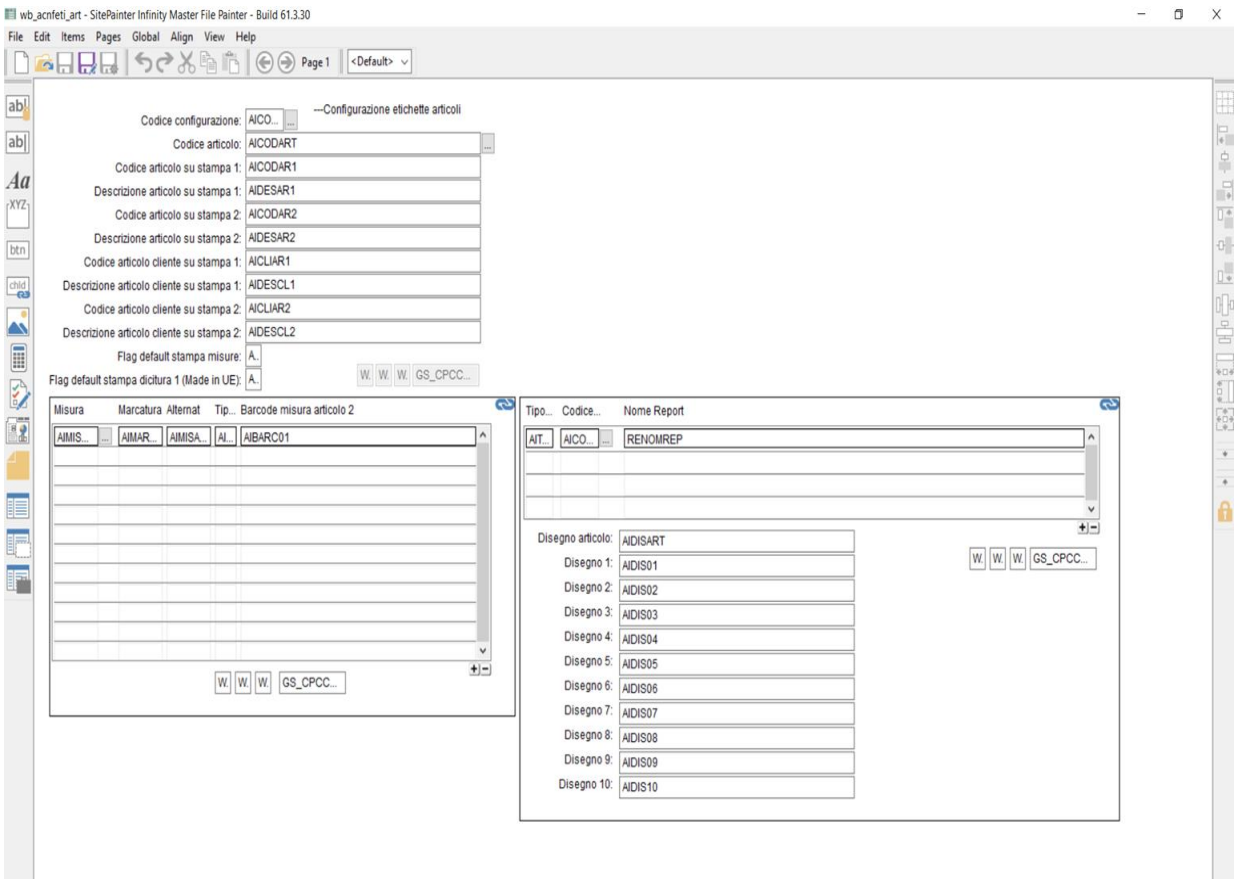


Figura 1.3 – Interfaccia utente Master

1.1.4 Routine

Il terzo strumento utilizzato sono le Routine chiamate anche funzioni o servlet. Vengono utilizzate all'interno del nostro web-service tramite una chiamata dal front end o dal back-end oppure da un evento come il login di un utente e sono scritte in linguaggio Java.

Queste routine vengono create attraverso un editor proprietario dello strumento che offre diversi vantaggi rispetto a un qualsiasi altro IDE.

Per notare le prime differenze un esempio è presente nella prossima Figura (*Figura 1.4*).

Questo perché ha un focus maggiore su determinati aspetti di sviluppo come, per esempio, le iterazioni con il database. Questo vantaggio si

traduce nella possibilità di utilizzare delle istruzioni "speciali" che raccolgono una serie di istruzioni Java utilizzate frequentemente per ridurre i tempi di sviluppo. Un esempio è la possibilità leggere delle informazioni ottenute da una query con un'unica istruzione come se fosse un ciclo "While…Do".

Un'altra caratteristica di questo editor è la divisione in "Pages" o pagine del servlet, questa suddivisione è utile solamente per organizzare meglio il sorgente perché effettivamente è come se si creasse una funzione.

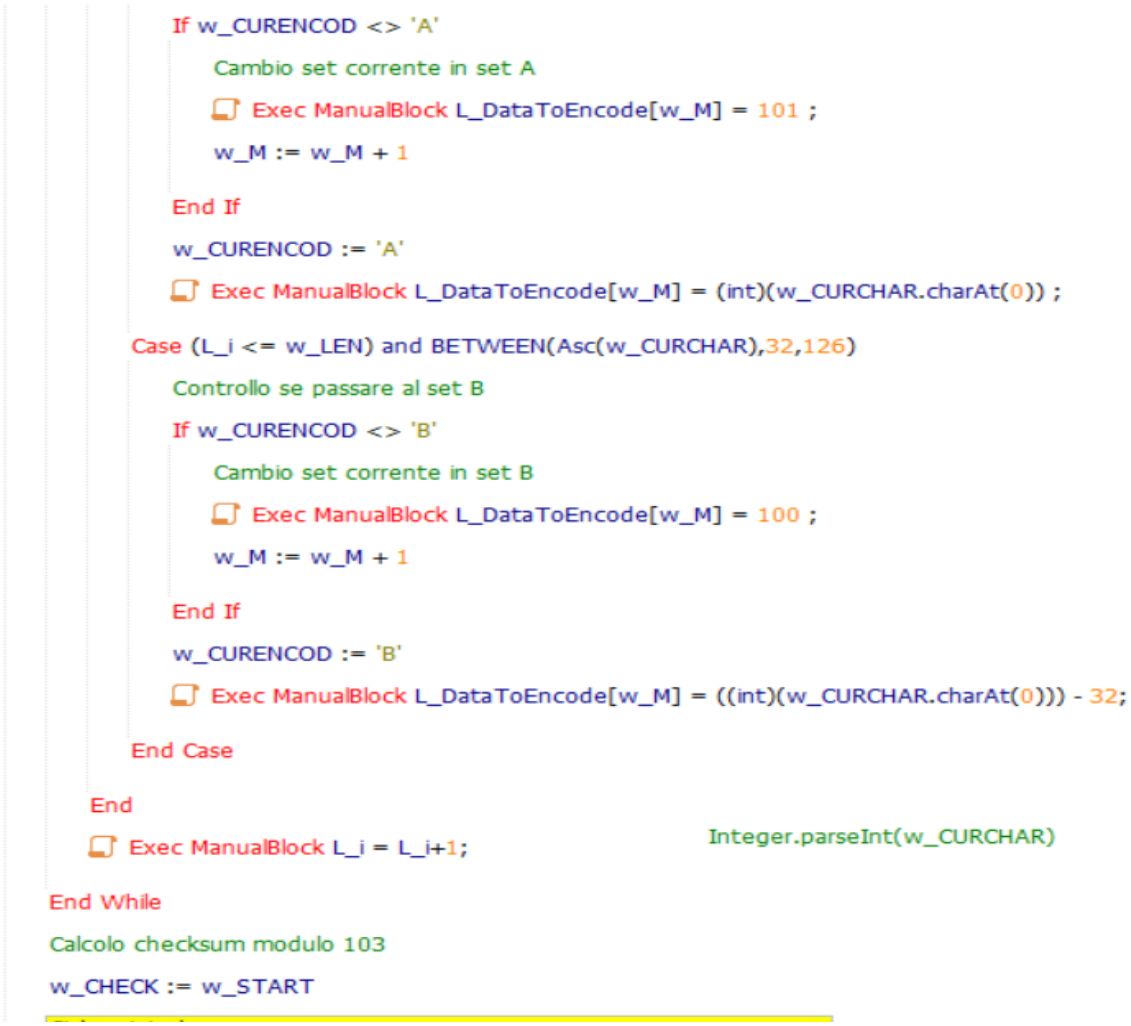


Figura 1.4 – Editor back-end

1.2.1 PortalStudio

PortalStudio è il secondo strumento utilizzato nella Suite Infinity il cui focus principale è il front-end con interazioni più dinamiche e uno sviluppo per il programmatore più semplice con l'utilizzo di linguaggi come Javascript lato Client.

"JavaScript è un linguaggio di programmazione multi paradigma orientato agli eventi, comunemente utilizzato nella programmazione Web lato client per la creazione, in siti web e applicazioni web. Nelle sue versioni più recenti può essere utilizzato anche per il lato server [2]"

È stato anche utilizzato del codice CSS per gestire la visualizzazione di alcuni oggetti a schermo.

" Il CSS, in informatica, è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML, ad esempio i siti web e relative pagine web. Le regole per comporre il CSS sono contenute in un insieme di direttive emanate dal W3C[5]"

Gli strumenti utilizzati al suo interno sono stati i Portlet, Report e Visual Query.

I Portlet del PortalStudio sono interfacce grafiche che saranno poi collegate a quelle del Back-End per poter essere utilizzate tramite una box richiamando il file del portlet. È possibile creare textbox, pulsanti, variabili, query, grid che sono simili a delle tabelle e altri strumenti per poter interagire con il back end o altri portlet.

Questi Portlet ci permettono di inserire codice in Javascript per gestire i vari eventi generati dall'utente, come per esempio al click di un pulsante si avvia una routine o una stampa di un Report o una query.

Un Report non è altro che un foglio di stampa dove è possibile decidere il layout. Sono stati molto utili per permettere la stampa delle etichette per il tracciamento delle fasi di produzione.

Per poter generare un report è necessario passare diversi parametri, tra cui il nome, il tipo di stampa (PDF,XML,HTML,SVG etc..), eventuali variabili e deve essere anche collegato a una Visual Query non vuota.

Le Visual Query sono delle semplicissime query con un'interfaccia grafica molto semplice. È stato molto utile perché ci ha permesso di creare in poco tempo un'unica query e di salvarla in un file .vqr e di metterla a disposizione di una o più funzioni o report.

Nella *Figura 1.5* è possibile notare quali strumenti ci offre e con cosa possono interagire, per esempio le query con i Portlet e i Report. Sono presenti anche oggetti del back-end ma non sono presenti effettivamente nel PortalStudio ma solo nel back-end SitePainter, qui sono presenti solo per capire come interagiscono con altri strumenti.

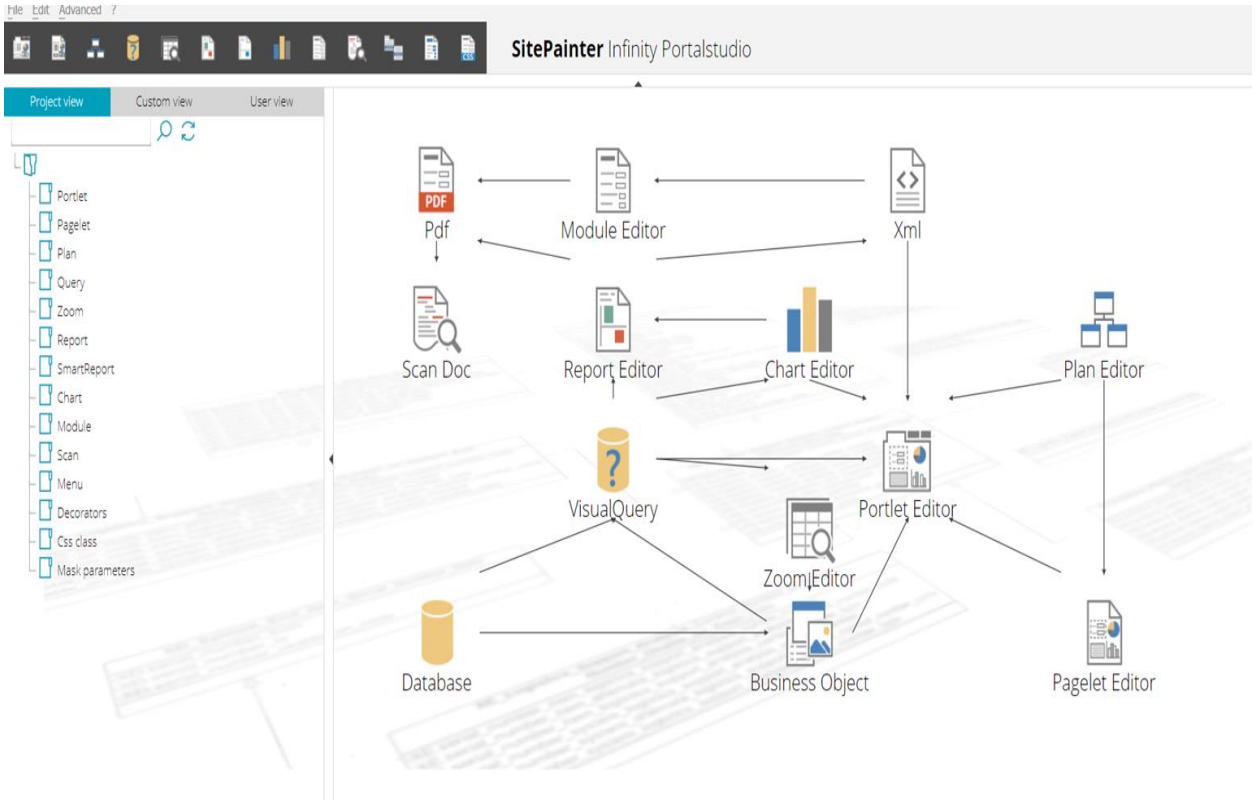


Figura 1.5 – Pagina principale PortalStudio

1.3 PostgreSQL

È stato scelto questo DBMS o Database Management System per il nostro database perché è affidabile, gratuito e l'azienda dove ho svolto il tirocinio lo aveva già utilizzato per altri progetti. Questo software ci permette di modificare, creare, prelevare e manipolare informazioni all'interno del nostro database nel modo più efficiente, sicuro e corretto possibile.

"In informatica, un database management system (abbreviato in DBMS o sistema di gestione di basi di dati) è un sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione efficiente di database[3]"

Nel progetto è servito solamente come base per la gestione del database perché ci sono bastati il PlanEditor per avere la struttura del database e le VisualQuery per il loro contenuto.

Le tabelle vengono salvate effettivamente come in *Figura 1.6*.

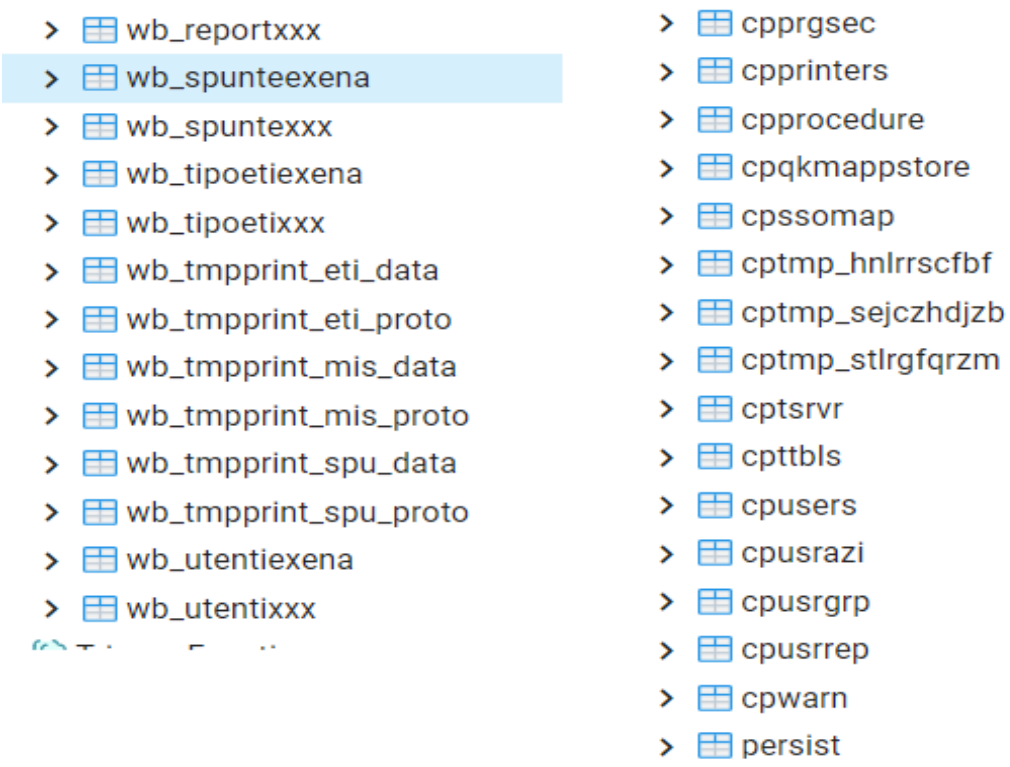


Figura 1.6 – Esempio tabelle presenti nel database

1.4 Apache Tomcat

È un web server open source gratuito sviluppato dalla Apache Software Foundation. Ci permette di rendere disponibile localmente e sia via web la nostra applicazione sviluppata in Java e Javascript.

"Apache Tomcat è un server web open source sviluppato dalla Apache Software Foundation. Implementa le specifiche JavaServer Pages (JSP) e servlet, fornendo quindi un'ambiente per l'esecuzione di applicazioni web sviluppate in linguaggio Java.[4]"

La versione utilizzata per questo progetto è la 9 perché la più stabile e affidabile.

Per rendere l'applicazione disponibile via web è stato molto facile perché l'azienda era già provvista di un web server configurato con Tomcat, quindi, è bastato inserire i sorgenti del sito web all'interno della cartella Webapps, configurare il file "dbconfig" nella cartella dei sorgenti del sito con i parametri del database e avviare il servizio.

Qualsiasi modifica back-end necessita un riavvio di questo servizio dopo la compilazione e deploy della nuova versione del sito. Per il front-end visto che utilizza Javascript e CSS qualsiasi modifica è in tempo reale.

Per l'utente per accedere al sito è basta utilizzare un link e avere un profilo utente registrato dall'amministratore per motivi di sicurezza.

PARTE II

Sviluppo e risultati

Capitolo 2

PROGETTAZIONE

In questo capitolo verranno illustrate le varie fasi di progettazione e sviluppo che si sono susseguite per raggiungere lo scopo del progetto.

Si inizierà con l'analisi per poi passare alle interfacce con una breve spiegazione delle funzioni utilizzate e concludere con la reportistica.

2.1 ANALISI

In questo capitolo verranno spiegate le varie tabelle e relazioni create ed utilizzate nel progetto e come vengono utilizzate.

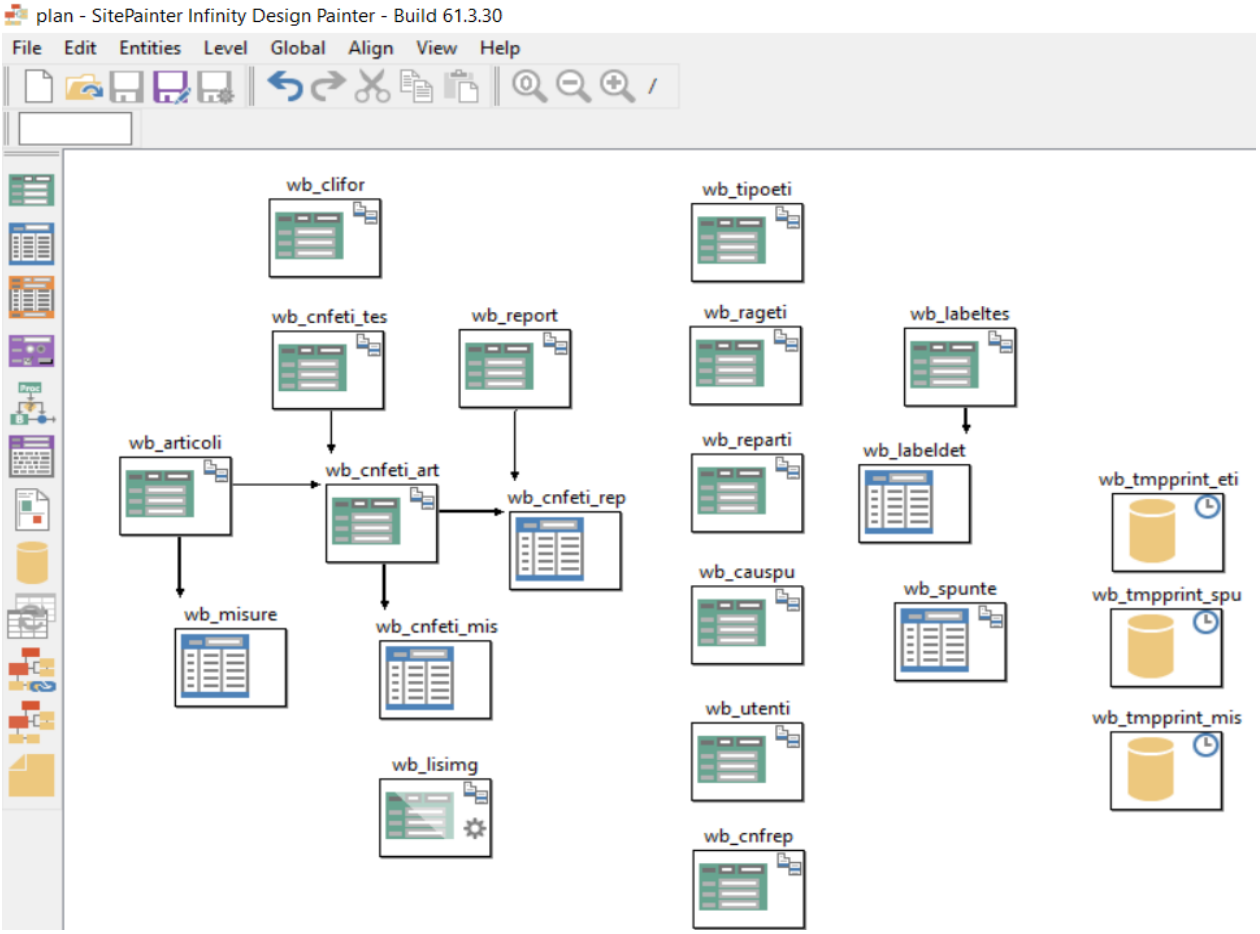


Figura 2.1 Plan Editor

La *Figura 2.1* ci mostra in modo molto semplice e grafico una panoramica della analisi con le varie relazioni tra le tabelle tramite un plan editor. Questa plan non include le tabelle utilizzate internamente dal SitePainter per i suoi processi come la tabella utenti di login al portale cpusers.

Come primo sviluppo sono state identificate e create le seguenti tabelle che formano base della nostra analisi: utenti, clienti/fornitori, articoli, report utilizzati e tipo etichette.

Nella seconda fase di analisi sono state create le tabelle che contengono le informazioni utili che servono per la stampa dell'etichetta.

Nella terza fase di analisi sono state create le tabelle per le spunte inviate dal palmare.

Nell'ultima fase di analisi sono state create delle tabelle che gestiscono la cronologia di creazione delle etichette in caso di utenti malevoli o errori di creazione.

2.1.1 ANALISI TABELLE FONDAMENTALI

Le prime tabelle create sono quelle per i reparti (wb_reparti), utenti (wb_utenti), clienti/fornitori (wb_clifor), articoli (wb_articoli) e misure (wb_misure) che contengono le informazioni più importanti e basilari del progetto da cui costruire il resto dell'analisi.

I reparti (wb_reparti) servono a identificare l'azienda o settore che sta utilizzando il palmare o il sito web es.: R1 = Azienda1, R2 = Azienda2, R3 = Sotto Reparto Azienda2 e come deve apparire nell'etichetta. Nel caso di questo progetto il codice del reparto è sempre uguale a quello che appare sull'etichetta ma è possibile che siano discordi.

La tabella utenti (wb_utenti) è stata creata per decidere chi può utilizzare i palmari tra gli operatori web, perché il SitePainter ha già una sua gestione interna con i relativi permessi ma in modo del tutto differente dal nostro scopo di utilizzo e legato solo al sito web. Inoltre, è contenuto anche il codice del reparto di riferimento ma non è stata creata una relazione con la tabella wb_reparti perché è possibile creare utenti con

reparti fittizi per controlli amministrativi o test. Questa tabella tramite una richiesta web service viene scaricata all'interno del palmare e si utilizza il codice utente per l'accesso al reparto.

I Clienti e Fornitori ci serviranno poi per la stampa delle etichette per identificare gli articoli e il layout corretto dell'etichetta da stampare, perché può succedere che due clienti o fornitori differenti utilizzino lo stesso codice articolo, ma per prodotti finali differenti. Quindi in questo modo abbiamo più configurazioni per il singolo articolo in base al cliente/fornitore. È possibile anche che un fornitore abbia lo stesso codice del cliente ma si differenziano con un campo di tipo contratto C= cliente F= fornitore.

La tabella degli articoli (wb_articoli) contiene una lista dei prodotti utilizzati ed è in relazione con la tabella wb_misure per gestire tutte le varie misure possibili di un articolo fino a un massimo di 22. Questo è un limite che ci siamo posti perché realmente ogni cliente ne utilizza massimo 18 e ci serve un numero massimo fisso per la stampa delle etichette e delle spunte per non avere problemi di spazio a video. Questo numero massimo di misure utilizzate è stato ricavato con una semplice estrazione da diversi database di alcuni clienti dell'azienda dove ho svolto il tirocinio e dal l'utilizzo futuro di questo strumento. Nel caso saranno necessarie più di 22 misure è possibile dividere l'articolo in due.

Le ultime due tabelle create per concludere l'analisi base sono wb_report e wb_tipoeti. La prima contiene una lista dei nomi dei file dei layout di stampa base dell'etichetta che in base all'articolo scelto dall'utente viene modificato e wb_tipoeti per la tipologia di etichetta es. scatola, scatolone etc.…

2.1.2 ANALISI STRUTTURA ETICHETTA STAMPA

Per stampare un'etichetta ci servono ulteriori informazioni come le immagini da allegare sull'etichetta, le descrizioni da far apparire, eventuali codici a barre aggiuntivi e quali tipi di etichette è possibile stampare.

Queste informazioni sono presenti in un gruppo di tabelle con il prefisso: wb_cnfeti_*.

Iniziando con la testata o parte iniziale dell'etichetta c'è la tabella wb_cnfeti_tes con le informazioni riguardanti i clienti e fornitori con delle etichette stampabili, perché non è detto che ogni cliente sia abilitato a stampare una sua etichetta.

Il corpo dell'etichetta è composto dalla tabella wb_cnfeti_art che contiene la lista degli articoli collegati alla testata e contiene le descrizioni ed è in relazione con la tabella wb_articoli perché non è possibile creare una scheda di stampa di un prodotto senza già averlo caricato nel sistema e definito le sue misure di default.

Le misure abilitate e se esiste un codice a barre e il suo tipo per l'articolo sono contenute nella tabella wb_cnfeti_mis.

Le immagini e i layout di stampa per tipo di etichetta sono salvati nella tabella wb_cnfeti_rep ed è in relazione con tutti i report disponibili nella tabella wb_report.

Nella *Figura 2.1.1* alla fine di questo paragrafo è possibile vedere tramite un diagramma uml come le tabelle wb_cnfeti_* siano collegate tra loro e con le tabelle articoli, misure e report viste nel paragrafo precedente.

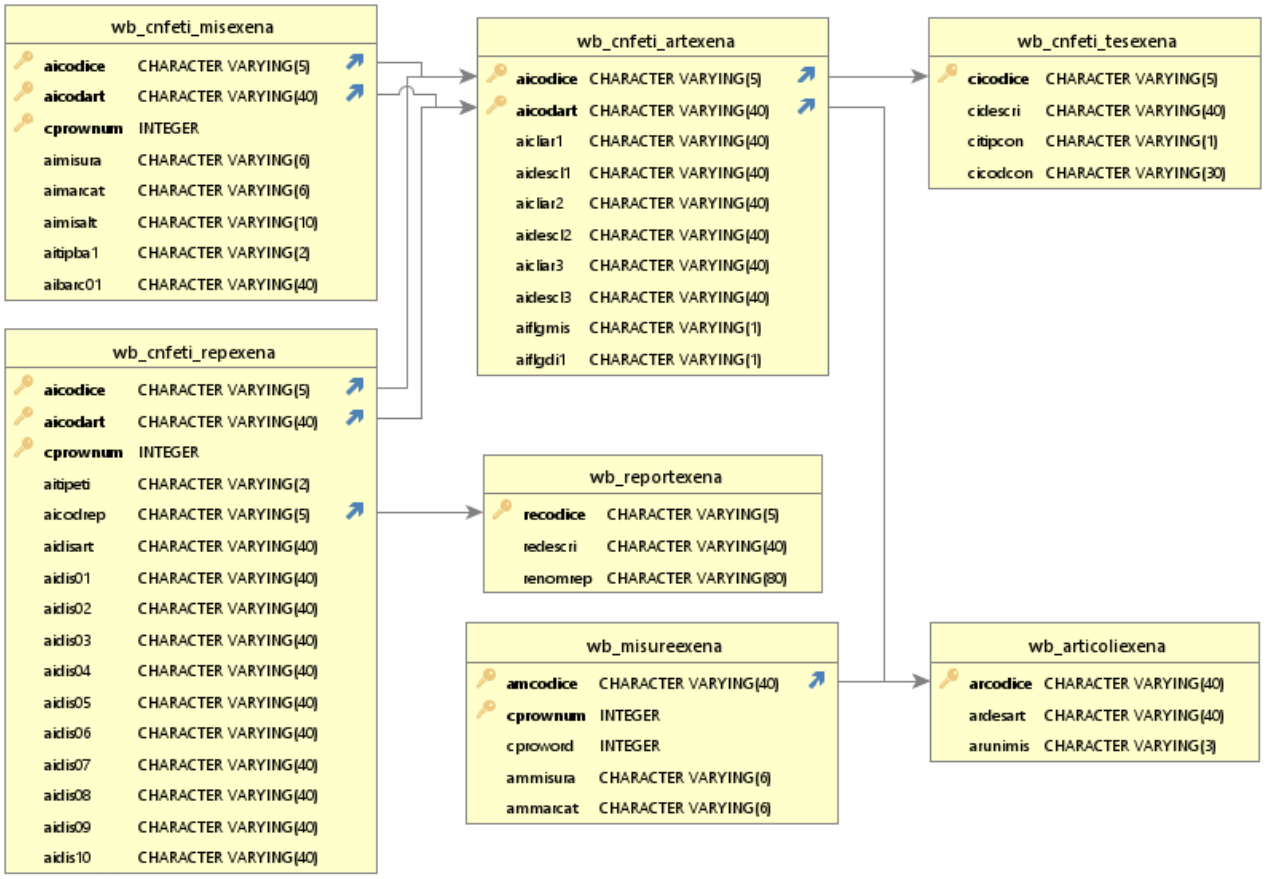


Figura 2.1.1 Diagramma UML Analisi stampa etichette

2.1.3 ANALISI SPUNTE PALMARE

Le tabelle utilizzate per le spunte sono wb_spunte e wb_causpu.

wb_spunte contiene le spunte inviate dall'operatore del palmare che devono subire un trattamento di elaborazione prima di essere pronte.

Sono composte da un codice univoco generato dal palmare che non è affidabile e uno univoco dal sistema con un autonumber inserito dopo l'elaborazione, codice operatore, data spunta, codice a barre in chiaro, reparto, causale, stato spunta, codice palmare.

Il seriale spunta del palmare è stato definito come non affidabile perché generato esternamente e non è possibile determinare se è assolutamente giusto o sbagliato quindi ci si è affidati a una seconda chiave univoca b3numero. La struttura delle spunte è visibile nella *Figura 2.1.2*.

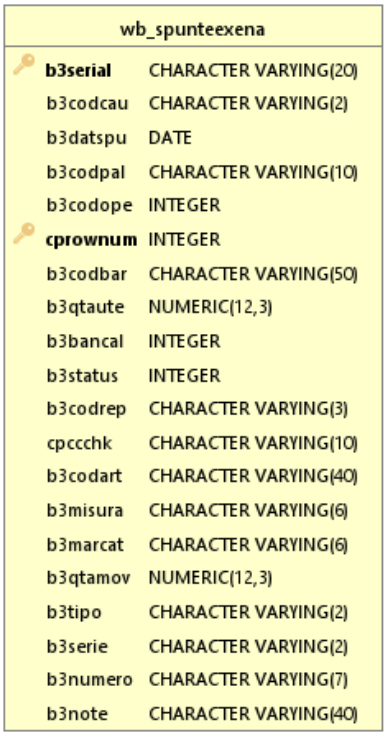


Figura 2.1.2 Diagramma UML Analisi spunta

wb_causpu è una tabella molto più semplice contenente una lista di codici e descrizioni delle causali disponibili per le spunte, questa tabella viene scaricata dal palmare tramite una richiesta al web service e verranno visualizzate nel palmare.

2.1.4 ANALISI CRONOLOGIA CREAZIONE ETICHETTA

Nell'ultima fase di analisi c'è stato un focus maggiore sulla sicurezza della piattaforma creando una cronologia di creazione di tutte le etichetta da parte dei vari utenti.

Le tabelle create a questo scopo sono tre: wb_labeltes, wb_labeldet, wb_rageti.

La prima contiene la testata di creazione di ogni etichetta con il cliente/fornitore, seriale etichetta, data, seriale raggruppo, tipo etichetta e reparto mentre l'articolo, misure e quantità sono state inserite in una tabella figlia wb_labeldet.

Con queste due tabelle possiamo leggere la cronologia di creazione di ogni singola etichetta ma per poter capire quante ne sono state create e stampate in una singola sessione ci serve il seriale di raggruppo.

Questo seriale viene inserito nella tabella wb_rageti come chiave e in ogni riga della testata di creazione di ogni etichetta nel campo etserrag.

Nella *Figura 2.1.3* a fine paragrafo è possibile vedere il diagramma uml con i campi e le relazioni tra le due tabelle. Non esiste una relazione diretta tra wb_labeltes e wb_rageti perché una cancellazione di un seriale di raggruppo porterebbe alla eliminazione di un numero considerevole di etichette nello storico. Per esempio, l'azienda usata come riferimento per questo progetto genera circa 800-2000 etichette per sessione, se per errore l'utente amministrativo cancellasse un seriale di raggruppo tutte le etichette collegate a esso verrebbero perse. Questo creerebbe un buco nello storico di creazione delle etichette che è da evitare.

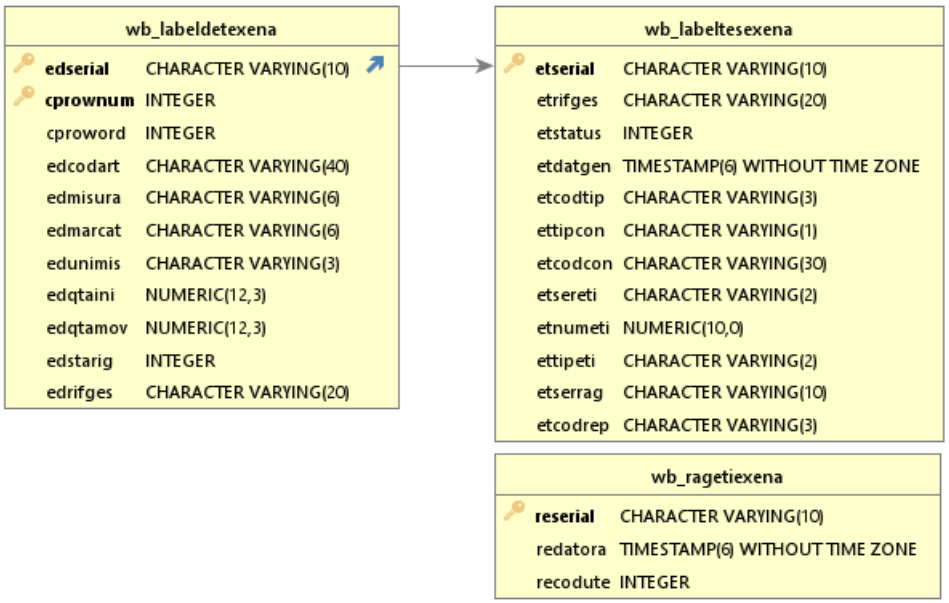


Figura 2.1.3 Diagramma UML Cronologia creazione etichette

2.2 INTERFACCE

Dopo aver definito l'analisi si è passati alla creazione e prova delle interfacce generate automaticamente dal SitePainter per verificare che l'analisi fatta fosse corretta.

Dopo alcune prove e aver verificato che l'analisi era corretta si è passati alla creazione delle due principali interfacce che verranno utilizzate da tutti gli utenti non amministrativi.

Le due interfacce sono:

- ➔ Generazione etichette
- ➔ Spunte

Ci sono anche altre interfacce per l'inserimento degli articoli, clienti/fornitori, tipo etichette e struttura etichetta ma per motivi di sicurezza non sono visibili ai semplici utenti ma solo agli amministratori, questo perché contengono dati sensibili che possono modificare anche il comportamento del web service.

2.3 GENERAZIONE ETICHETTE

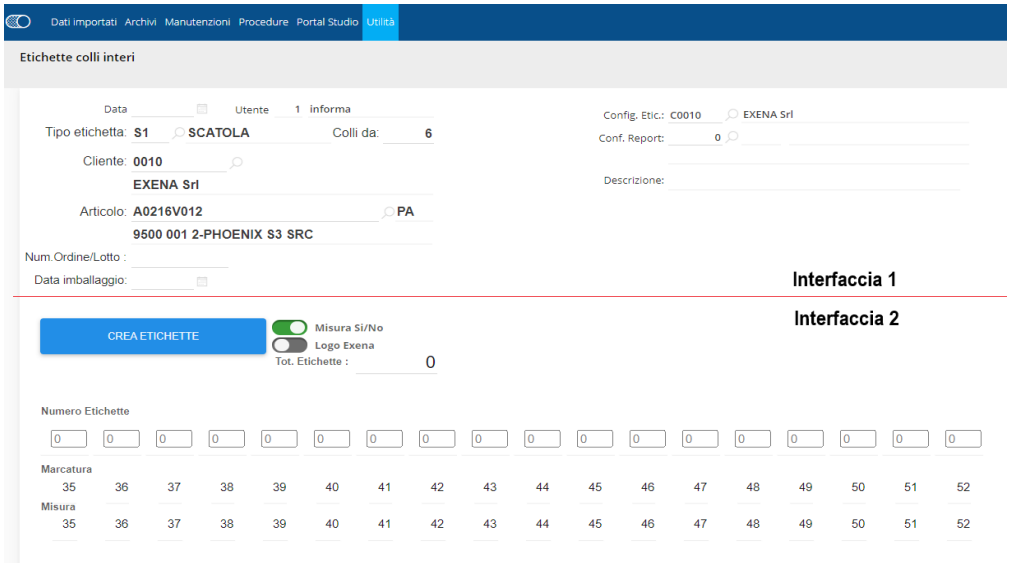


Figura 2.2 Interfaccia utente generazione etichette

Guardando l'immagine *Figura 2.2* possiamo notare il numero elevato di campi utilizzati per generare delle etichette per un articolo di un cliente.

L'interfaccia è stata pensata per permettere all'utente di inserire in ordine il tipo di etichetta che si vuole stampare, il numero di colli, per quale cliente e quale articolo. Il numero ordine/lotto, Data imballaggio e Descrizione sono dei campi liberi che l'utente può valorizzare per apparire nell'etichetta.

Ad ogni campo con una lente a lato se viene valorizzato dall'utente si eseguono dei controlli se nel database esiste quell'informazione. Si utilizza sempre una chiave univoca per quei campi per evitare ambiguità. In caso positivo vengono riempiti anche i campi descrittivi vicini altrimenti apparirà un messaggio di errore e verrà svuotato il campo.

Alla valorizzazione dei campi presenti sull'interfaccia back-end da parte dell'utente vengono eseguite le seguenti funzioni:

wb_fprint_eti1

wb_fprint_eti2

wb_fprint_eti3

La prima funzione wb_fprint_eti1 viene lanciata quando si devono passare informazioni dalla prima interfaccia alla seconda tramite dei eventi perché quest'ultima genererà l'etichetta con le informazioni inserite.

La seconda funzione wb_fprint_eti2 verrà lanciata quando l'utente avrà inserito tipo, cliente e articolo. La sua prima esecuzione che avverrà nel back-end si limita a creare e preparare una tabella temporanea che conterrà un'etichetta per ogni riga.

Questa tabella temporanea è presente nella *Figura 2.1* a destra con il nome di wb_tmpprint_eti, la sua struttura è dinamica e viene decisa in questa fase.

L'ultima funzione wb_fprint_eti3 viene eseguita subito dopo wb_fprint_eti2 e serve a trovare il report corretto da proporre all'utente. Per farlo legge nell'anagrafica degli articoli nella tabella wb_ cnfeti_rep cercando in base al cliente, all'articolo e il tipo di etichetta e ricavare il codice report che poi va cercato se esiste nella tabella wb_report. In caso positivo viene estratto il nome del file della struttura del report, ma in caso contrario viene caricato uno di default per quel tipo di etichetta.

Nel caso l'utente voglia cambiare etichetta può farlo. Nella *Figura 2.2* a destra c'è la dicitura Conf.Rep che contiene il report selezionato e con la lente d'ingrandimento è possibile selezionare uno tra i report disponibili nella tabella wb_report.

2.3.1 STAMPA ETICHETTE

Per generare un'etichetta l'utente dopo aver valorizzato tutti i campi obbligatori (tipo etichetta, cliente, articolo, numero colli, numero etichette) può premere il pulsante blu chiamato "Crea Etichette" presente nella *Figura 2.2*.

Questo pulsante avvierà nuovamente la funzione wb_fprint_eti2, passando come parametri i valori inseriti dall'utente.

La funzione durante la sua seconda esecuzione non farà altro che prendere le informazioni dell'articolo dalla sua anagrafica prelevando i dati che a noi servono che sono le sue descrizioni, codici a barre per ogni misura e le immagini e salvarli nella tabella temporanea wb_tmpprint_eti. All'interno di questa tabella avremo per ogni riga una etichetta quindi nel caso si voglia stampare 100 etichette troveremo all'interno di questa tabella 100 righe. Questo perché il sistema che genera i report considera una riga di una query fatta su una tabella come una pagina di un report da stampare e nel nostro caso per ogni pagina avremo un'etichetta.

In simultanea alla esecuzione di questa fase la funzione salverà nelle tabelle wp_labeltes, wp_labeldet e wp_rageti le informazioni di generazione dell'etichetta per motivi di sicurezza.

2.3.2 REPORT

I report che sono stati creati per questo progetto sono due, uno per la tracciabilità dell'articolo e un altro senza. Questi report contengono la struttura dell'etichetta che andremo a stampare.

Sono stati realizzati con il tool di creazione di report di PortalStudio inserendo in modo molto facile e intuitivo i vari campi che ci servono e utilizzando come fonte dei dati una query che preleva i dati da una tabella temporanea che abbiamo riempito precedentemente.

Non è detto poi che ogni campo verrà valorizzato ed è possibile anche inserire condizioni per nascondere vari componenti in mancanza di altre informazioni.

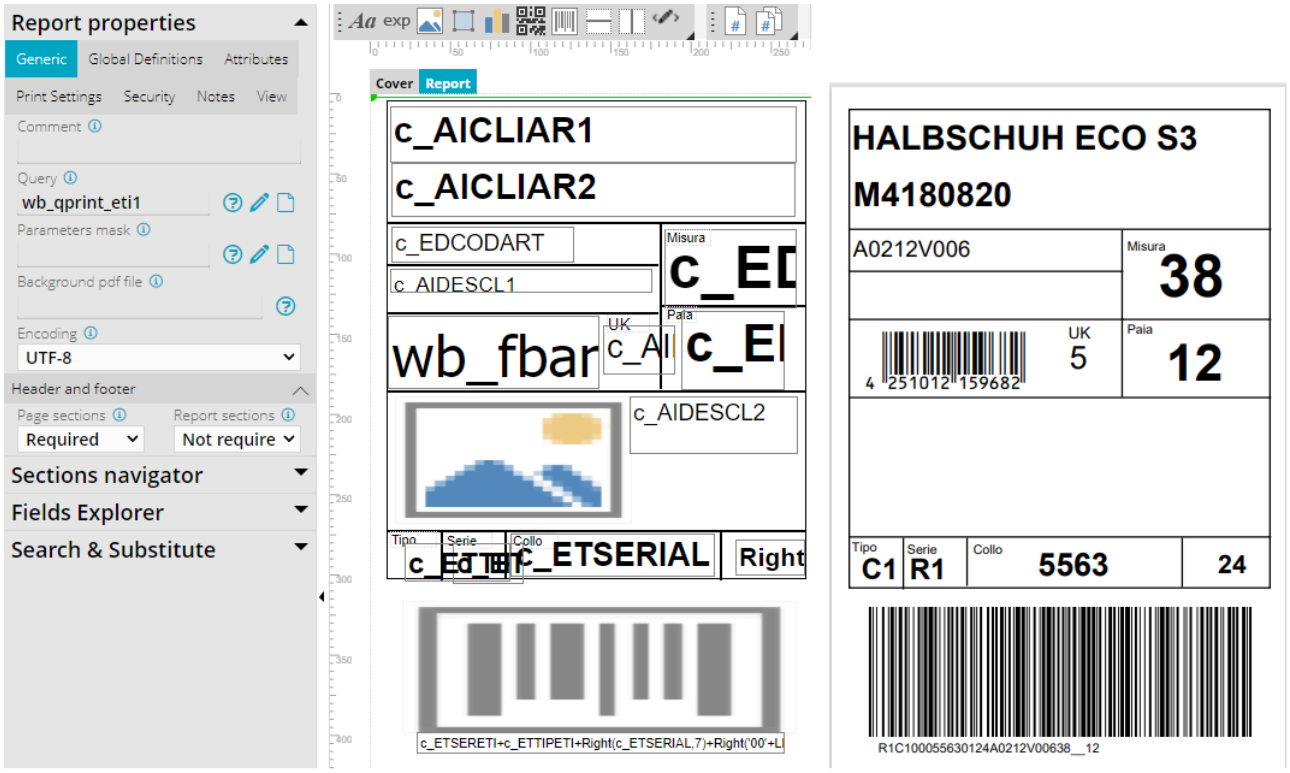


Figura 2.7 Struttura Etichetta

Nella *Figura 2.7* possiamo vedere a sinistra la struttura di un report con tracciamento mentre a destra il suo risultato con una stampa. Nel caso di un'etichetta senza tracciamento i codici a barre non sono presenti oppure è presente solamente l'ean13.

Nelle etichette con codice a barre si è utilizzata una funzione per convertire una stringa in codice Code128 e un font speciale per poter interpretare il risultato e visualizzare a schermo un codice a barre. Si è utilizzato un carattere di dimensione più elevata possibile per ottenere una buona lettura da parte del palmare.

In alcuni articoli è presente anche un codice a barre Ean13, che è molto comune ed utilizzato per esempio dalla maggior parte dei supermercati in Italia e all'estero. È stata creata una funzione specifica anche per generare questo specifico codice a barre e usato un font per poter interpretare il risultato.

Nella *Figura 2.7* è possibile notare come i riquadri contenti i due codici a barre siano differenti. Questo è causato dal fatto che nel primo è stato utilizzato un font esterno allo strumento mentre nel secondo è interno.

In caso di errori non viene generato il report e verrà segnalato un errore generico e se invece mancano delle informazioni necessarie per la stampa comparirà un messaggio di avviso a riguardo.

2.3.3 GENERAZIONE BARCODE

Sono state utilizzate due funzioni per la creazione dei barcode code 128 ed ean13 e un font speciale per ognuno per interpretarne il risultato.

La prima funzione è più complessa e prende in input i dati necessari per identificare l'articolo per la tracciatura.

Queste informazioni sono calcolate con la seguente funzione:

c_ETSERETI+c_ETTIPETI+Right(c_ETSE-
RIAL,7)+Right('00'+LRTrim(Str(c_CPROWNUM,2,0)),2)+Anno-
Mese+Left(c_EDCODART,9)+Left(Left(c_EDMI-
SURA,4)+"____",4)+Right('00'+LRTrim(Str(c_EDQTAMOV,2,0)),2)

E possiamo vedere il risultato in basso a destra della *Figura 2.7*.

L'elaborazione consiste di creare due campi che indicano l'inizio e la fine del barcode per la lettura e non verranno salvati dallo strumento.

Dopo il carattere di inizio ci sono i dati veri e propri che sono divisi in tre categorie:

A → Maiuscole e caratteri speciali

B → Lettere maiuscole e minuscole

C → Numeri

Dopo i dati ci sono i numeri di controllo d'integrità per verificare se ci sono eventuali errori e in fine il carattere di chiusura del barcode.

Oltre il Code128 c'è la possibilità di inserire nell'etichetta con barcode di tipo Ean13 (European Article Number) il quale è a discrezione dell'azienda che lo utilizza di inserirlo o farlo inserire nel sistema. Si limita a un massimo di 13 caratteri con 3 metodi di codifica A,B e C e 3 caratteri extra di controllo.

Il primo carattere è quello di inizio (101).

Poi ci sono 6 caratteri di cui 1 riservato per identificare la codifica utilizzata se di tipo A o B.

Un carattere di controllo a metà (01010) e poi altri 6 caratteri con codifica di tipo C.

Alla fine, c'è l'ultimo carattere di controllo (101) per indicare la fine del barcode.

2.4 SPUNTE

Le spunte sono la conferma di lettura di una etichetta con tracciabilità da parte di un palmare.

Si dividono in due stati:

1) Non Elaborata

2) Elaborata

Nel primo caso è la spunta grezza che viene ricevuta dal web service da parte del palmare con solo le informazioni relative al codice a barre in chiaro, palmare, operatore, data, bancale, causale, quantità totale e reparto

Le informazioni mancanti sono codice univoco spunta affidabile, l'articolo, misura e quantità misura contenute nel codice a barre in chiaro ricevuto che deve esser elaborato.

L'elaborazione è molto semplice, l'utente seleziona le spunte da elaborare e preme il pulsante "Elabora" che esegue una funzione che prende in input il Seriale Spunta e se presente il "Numero registrazione" se già elaborata.

Esegue una query che preleva i codici barre di quella spunta e tramite delle substring sezionare i vari campi che ci interessano.

Un esempio utilizzando il codice a barre della *Figura 2.7* può essere il seguente.

Codice a barre: R1C100055630124A0212V00638_12

Articolo: R1C100055630124A0212V00638_12

Numero: R1C100055630124A0212V00638_12

Quantità: R1C100055630124A0212V00638_12

Gli altri campi servono per identificare:

Reparto generazione: R1C100055630124A0212V00638_12

Mese creazione in due cifre: R1C100055630124A0212V00638_12

Numero etichetta: R1C100055630124A0212V00638_12

Che non ci interessano ma servono nel caso si voglia risalire alla sua generazione.

Questa fase non è stata inserita nel palmare per evitare di complicare e specializzare troppo l'applicazione verso un determinato codice a barre ma di tenerla il più generica possibile in caso vengano utilizzati codici a barre nuovi.

La fase di elaborazione deve esser svolta manualmente dall'utente sul portale web per poter essere valutate dall'operatore che deciderà se accettarle o rifiutarle in caso di problemi e di aggiungere/cambiare con una seconda elaborazione una nota.

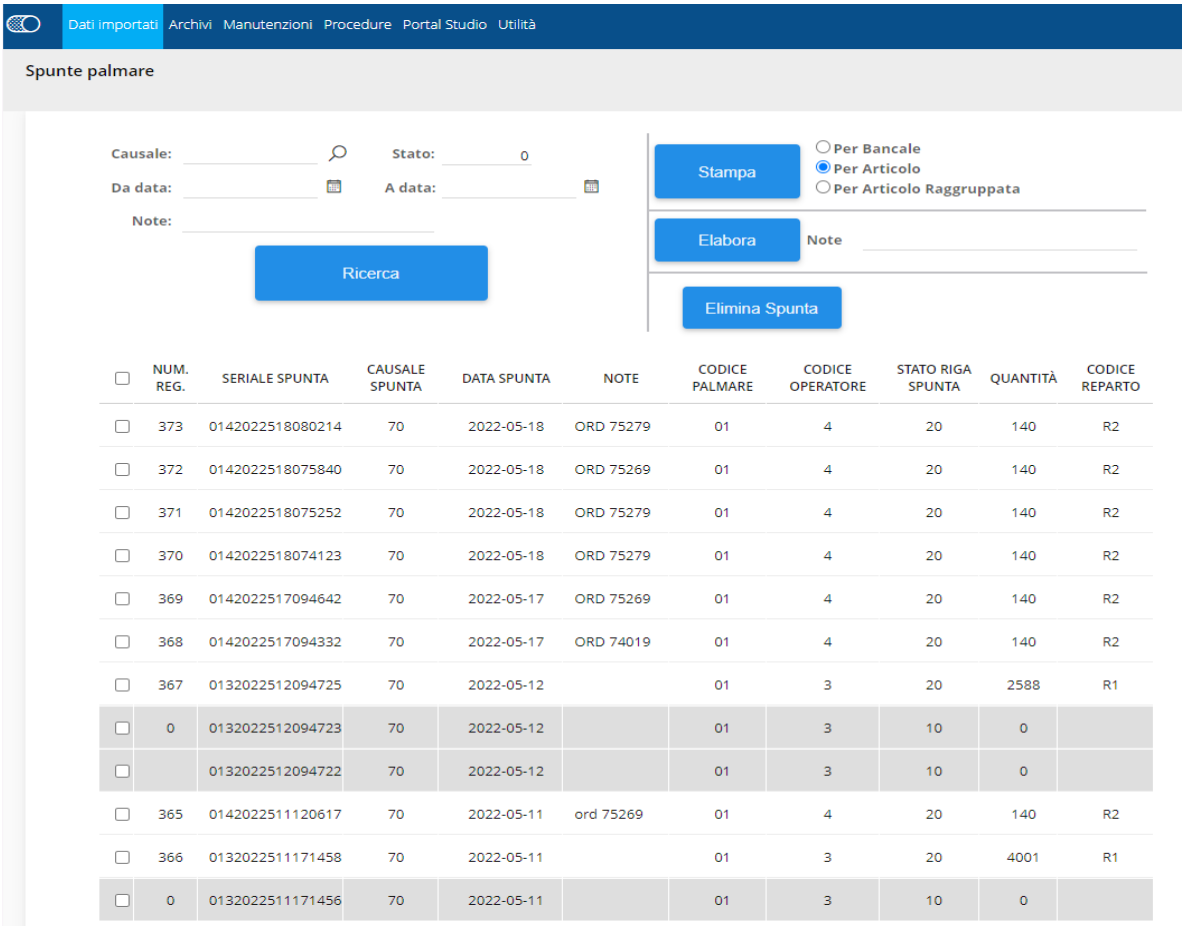
La funzione inoltre controlla eventuali errori o codici a barre non gestiti e segnalare il problema all'operatore.

È da notare che la spunta ha due seriali, uno fornito dal palmare e uno invece assegnato dopo l'elaborazione della spunta. Sono stati utilizzati due perché è possibile che all'interno dello stesso reparto due palmari che non possono comunicare tra di loro, perché non è presente un'infrastruttura che gli possa permetter ciò, generino lo stesso seriale che è creato secondo questa logica:

Codice Palmare + Codice Operatore + Reparto + Giorno/Mese/Anno + Ora fino ai millisecondi

Il codice palmare non dovrebbe crear problemi di univocità ma l'app utilizza file di configurazione in chiaro modificabili da un utente medio/esperto quindi in fase di analisi per evitare problemi da parte di utenti malevoli è stata presa in considerazione l'eventuale presenza di due palmari con lo stesso codice per non bloccare il sistema e l'azienda creando un danno.

Nella *Figura 2.8* a fine capitolo è presente l'interfaccia utente delle spunte, e righe in grigio sono le spunte non elaborate. Le spunte visualizzate nell'interfaccia sono raggruppate per SerialeSpunta, Num. Reg., Causale, Data, Note, Codice Palmare, Codice Operatore. Questo perché una spunta può avere anche cento letture al suo interno e non è pratico per l'utente da visualizzare ed elaborare.



	NUM. REG.	SERIALE SPUNTA	CAUSALE SPUNTA	DATA SPUNTA	NOTE	CODICE PALMARE	CODICE OPERATORE	STATO RIGA SPUNTA	QUANTITÀ	CODICE REPARTO
☐	373	0142022518080214	70	2022-05-18	ORD 75279	01	4	20	140	R2
☐	372	0142022518075840	70	2022-05-18	ORD 75269	01	4	20	140	R2
☐	371	0142022518075252	70	2022-05-18	ORD 75279	01	4	20	140	R2
☐	370	0142022518074123	70	2022-05-18	ORD 75279	01	4	20	140	R2
☐	369	0142022517094642	70	2022-05-17	ORD 75269	01	4	20	140	R2
☐	368	0142022517094332	70	2022-05-17	ORD 74019	01	4	20	140	R2
☐	367	0132022512094725	70	2022-05-12		01	3	20	2588	R1
☐	0	0132022512094723	70	2022-05-12		01	3	10	0	
☐		0132022512094722	70	2022-05-12		01	3	10	0	
☐	365	0142022511120617	70	2022-05-11	ord 75269	01	4	20	140	R2
☐	366	0132022511171458	70	2022-05-11		01	3	20	4001	R1
☐	0	0132022511171456	70	2022-05-11		01	3	10	0	

Figura 2.8 Interfaccia utente Spunte

2.4.1 STAMPA SPUNTE

L'utente oltre a elaborare e vedere le spunte ordinate in una tabella può stamparle per poter visionare cosa contiene ogni riga di una spunta o avere un resoconto del lavoro fatto.

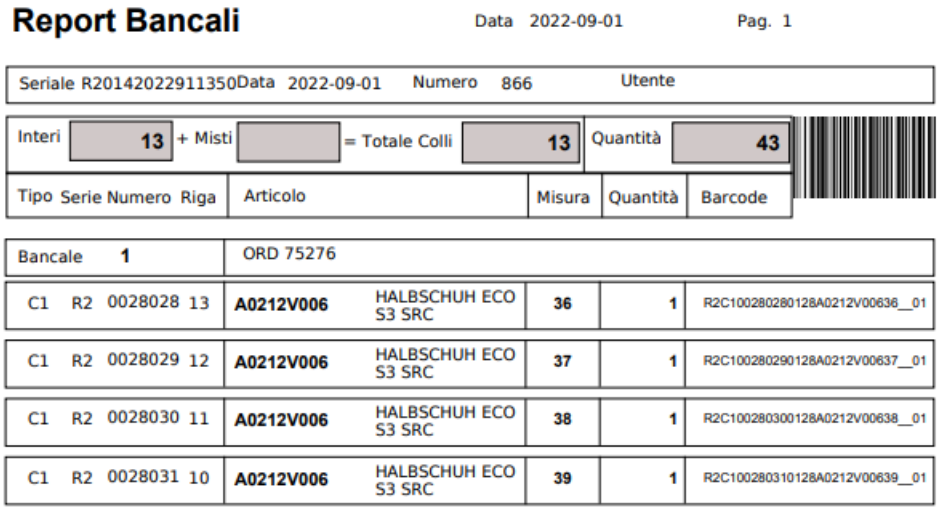
Le stampe disponibili ,presenti nella *Figura 2.8* in alto a destra, sono tre e in base alle spunte selezionate vengono presentate in modo simile a come avviene per la generazione delle etichette.

Nella prima stampa "per bancale" si ha una lista di ogni spunta organizzata per bancale. Un esempio di stampa è presente nella *Figura 2.9* a fine capitolo.

La seconda e la terza stampa sono uguali perché usano lo stesso report ma si diversificano per la query utilizzata. Nella stampa "Per Articolo" abbiamo per ogni spunta il totale di ogni articolo. Un esempio è presente nella *Figura 2.10* a fine capitolo.

Invece nella stampa "Per Articolo Raggruppata" non ci interessa la divisione per spunta ma si ha un totale per articolo generale.

Queste ultime due stampe utilizzano anche una tabella temporanea perché come possiamo vedere nella *Figura 2.10* si ha una riga con tutte le misure disponibili per quell'articolo e le quantità, ma non è possibile farlo tramite un report come nella *Figura 2.9* perché ci è permesso la visualizzazione di un record di una query alla volta per riga. Quindi è stato necessario utilizzare una funzione che inserisce ad ogni colonna una misura e la sua quantità e una query per il report che ne estrae i risultati per metterli tutti in un'unica riga.



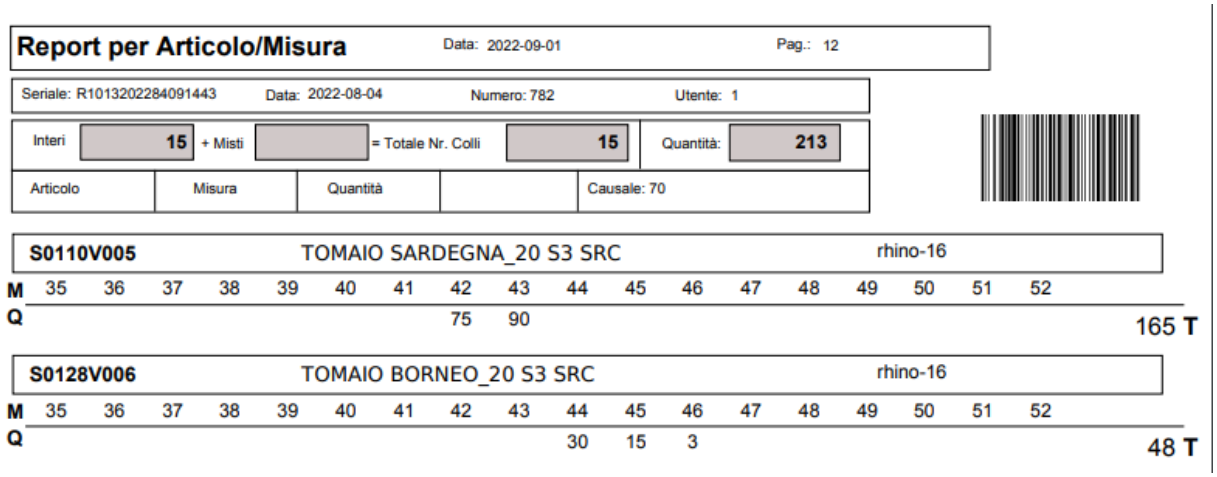
Report Bancali Data 2022-09-01 Pag. 1

Seriale R20142022911350 Data 2022-09-01 Numero 866 Utente

Interi 13 + Misti = Totale Colli 13 Quantità 43

Tipo	Serie	Numero	Riga	Articolo		Misura	Quantità	Barcode
Bancale 1				ORD 75276				
C1	R2	0028028	13	**A0212V006**	HALBSCHUH ECO S3 SRC	**36**	**1**	R2C100280280128A0212V00636__01
C1	R2	0028029	12	**A0212V006**	HALBSCHUH ECO S3 SRC	**37**	**1**	R2C100280290128A0212V00637__01
C1	R2	0028030	11	**A0212V006**	HALBSCHUH ECO S3 SRC	**38**	**1**	R2C100280300128A0212V00638__01
C1	R2	0028031	10	**A0212V006**	HALBSCHUH ECO S3 SRC	**39**	**1**	R2C100280310128A0212V00639__01

Figura 2.9 Stampa dettaglio spunte



Report per Articolo/Misura Data: 2022-09-01 Pag.: 12

Seriale: R1013202284091443 Data: 2022-08-04 Numero: 782 Utente: 1

Interi 15 + Misti = Totale Nr. Colli 15 Quantità: 213

Articolo | Misura | Quantità | | Causale: 70

S0110V005 TOMAIO SARDEGNA_20 S3 SRC rhino-16

M	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	
Q								75	90										165 T

S0128V006 TOMAIO BORNEO_20 S3 SRC rhino-16

M	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	
Q										30	15	3							48 T

Figura 2.10 Stampa per Articolo spunte

CAPITOLO 3

Implementazione

In questo capitolo verranno illustrate le parti più tecniche e specifiche d'implementazione riguardo la maschera più complessa del progetto.

Verranno mostrate anche parti di codice sorgente.

3.1 GESTIONE DOPPIA INTERFACCIA STAMPA ETICHETTE

Come possiamo notare nella *Figura 2.2* è divisa in due parti, una sviluppata nel back-end e una nel front-end. Questa scelta è stata dettata dal fatto che inizialmente non era previsto l'inserimento di una tabella orizzontale per le misure degli articoli e l'unico strumento che ci permetteva di farlo è PortalStudio front-end, quindi si è optato per una doppia interfaccia per motivi di tempistiche perché la parte back-end già era stata sviluppata.

Questo problema è una limitazione tecnica dello strumento SitePainter back-end poiché in java è possibile creare tabelle graficamente ma non in questo particolare ambiente soprattutto visto che è possibile già farlo con PortalStudio in modo più semplice.

Notata questa limitazione le future implementazioni grafiche sono state fatte in PortalStudio e si è limitati solo a funzioni complesse con il SitePainter come la generazione delle etichette.

Questa scelta si è dimostrata inoltre molto vantaggiosa perché ci ha permesso di creare delle funzioni brevi e semplici velocemente con PortalStudio, e dato che il linguaggio utilizzato è Javascript non richiede tempi di compilazione.

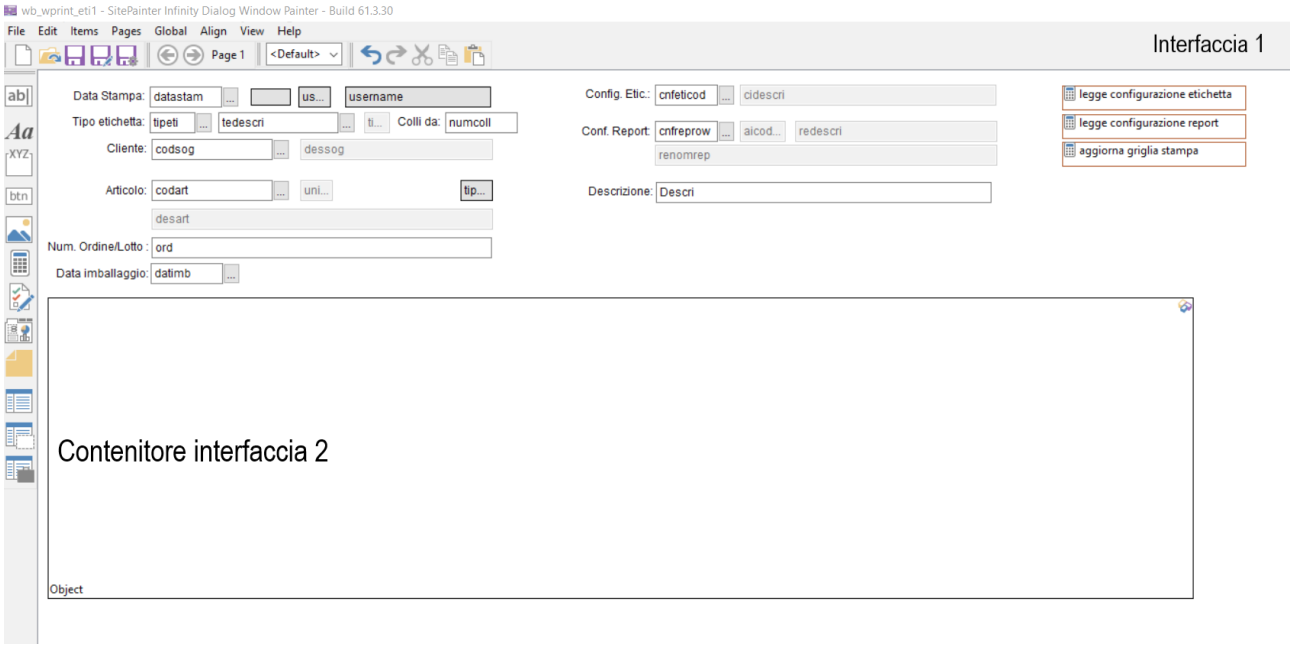


Figura 3.1 Interfaccia 1 Back-end

Nella *Figura 3.1* possiamo vedere com'è stata sviluppata l'interfaccia di generazione delle etichette.

Per poter unire le due interfacce è stato utilizzato un Object che importa un portlet creato tramite PortalStudio e ci permette di visualizzarlo al suo interno e adattare in modo dinamico le sue dimensioni di visualizzazione, altrimenti vedremo ai lati dello schermo delle barre verticali e/o orizzontali di scorrimento.

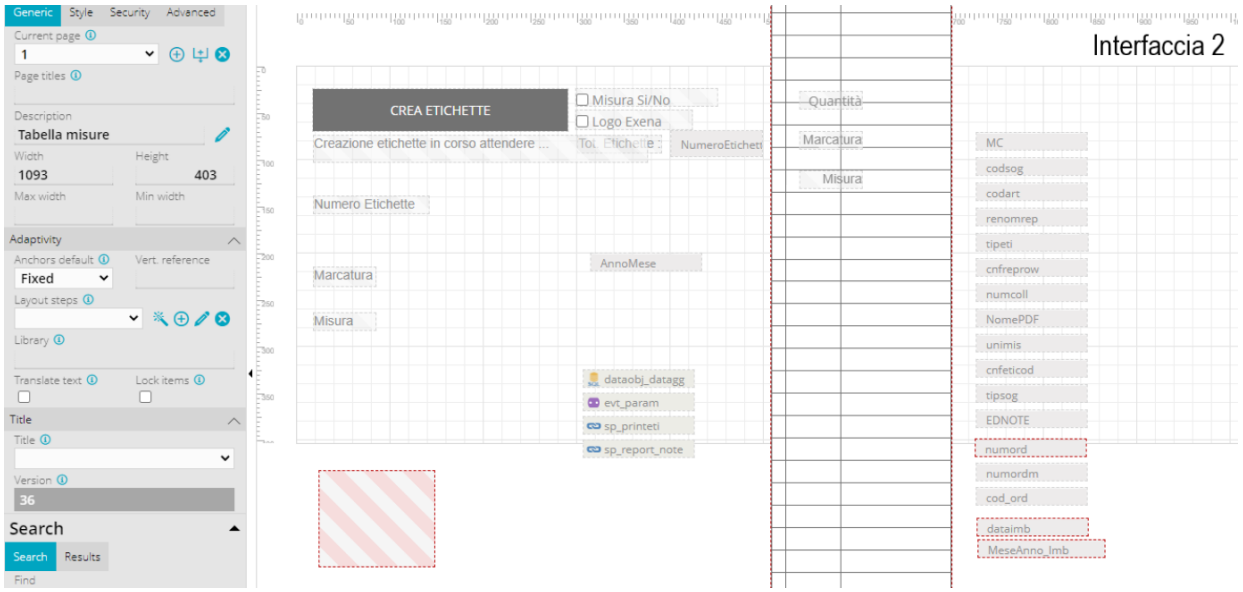


Figura 3.2 Interfaccia 2 Front-end

Nella *Figura 3.2* possiamo vedere com'è stata sviluppata la seconda interfaccia, la tabella in questa presentazione apparirà in verticale ma tramite l'utilizzo di codice css è stato possibile girarla.

Il codice necessario per farlo è il seguente:

transform: rotate(-90deg);

E apparirà come nella precedente Figura 3.2, ed è da notare che anche le varie celle della tabella sono state fatte girare tramite ccs di 90 gradi ma verso destra con lo stesso codice.

3.2 GESTIONE EVENTI

Per poter comunicare tra due o più interfacce come nel nostro caso si deve generare un evento e avere un ascoltatore dall'altra parte per poterlo intercettare e leggere le informazioni. Nel caso di questo progetto è stato possibile tramite l'utilizzo di una funzione scritta in java che emette un evento e l'utilizzo di un particolare strumento di PortalStudio per la sua ricezione.

Nella parte in java si è creata la funzione presente nella *Figura 3.3* a fine pagina per poter emettere un evento.

```
p_tipop Numeric(1, 0)
If p_tipop = 1
    If Check()
        Exec ManualBlock SetControlsValue();
        Exec ManualBlock EnableControlsUnderCondition();
        Exec ManualBlock var parmobj={ptipet:w_tipeti,ptipsog:w_tipsog,pcodsog:w_codsog,pcodart:w_codart,punimis:w_unimis,pnumcol:w_numcol,pcnfeticod:w_cnfeticod,prenomrep:w_renomrep,pcnfreprow:w_cnfreprow,pnumord:w_numord,pnumordm:w_numordm,pdataimb:w_dataimb,ptipsta:w_tipsta};
        Exec ManualBlock window.ZtVWeb.raiseEvent('evt_param',parmobj);
    Else
        DisplayErrorMessage()
    End If
Else
```

Figura 3.3 Sorgente EventEmitter

Nella *Figura 3.3* Dopo il Check di controllo per verificare se l'operazione è andata a buon fine ci sono due istruzioni per inizializzare delle librerie proprietarie Zucchetti per accedere a delle funzioni presenti in Portal-Studio e scrivere addirittura del codice javascript in SitePainter.

Nella terza istruzione notiamo subito che paramobj è una variabile scritta in javascript, e che contiene una lista di valori che sono i parametri che verranno inviati tramite evento, e nel nostro caso alla seconda interfaccia, strutturati come *<nome-variabile:valore>*.

L'ultima istruzione serve ad emettere un evento sulla finestra attualmente aperta passando come parametri l'oggetto che dovrà ricevere le informazioni nella seconda interfaccia e i dati da inviare.

Invece su PortalStudio la situazione è molto più semplice perché ci basta creare un oggetto ricevitore che intercetta l'evento. Dato che in questo caso passiamo dei valori verranno inseriti in delle variabili presenti in PortalStudio come appoggio. Alla fine, si è utilizzata anche una funzione "aggiorna()"

```
function aggiorna() {

    this.dataobj_datagg.ChangeQuery("wb_qmisure1")
    this.dataobj_datagg.Query()

    globaleFocus = document.getElementsByClassName("grid_table");
    globaleIndex = this.Grid7;

}
```

Figura 3.4 Sorgente lancio query PortalStudio con Javascript

La funzione presente nella *Figura 3.4* richiama una query che serve a popolare la tabella delle misure. Le altre due funzioni servono per poter scorrere gli elementi presenti nell'interfaccia con la pressione del pulsante invio invece di usare tab.

L'oggetto che riceve questo evento è presente nella *Figura 3.5* con il nome di "evt_param" che contiene una struttura del genere:

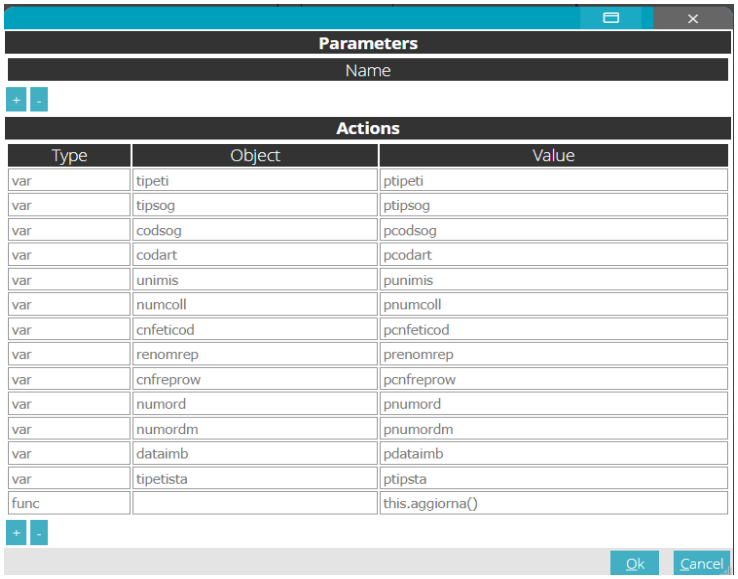


Figura 3.5 Interfaccia event receiver

Nella Figura 3.5 a sinistra abbiamo il tipo se è una variabile (var) oppure una funzione (func) e segue l'ordine di ricezione dei parametri per l'esecuzione di ogni riga. Nella colonna centrale ci sono le variabili di destinazione e a destra i valori che andranno a valorizzarla.

Nel progetto questa comunicazione si è limitata tra back-end e front-end ma è possibile eseguirla anche fra due o più interfacce front-end sulla stessa pagina web.

CONCLUSIONI E OSSERVAZIONI

Il progetto è stato provato da un cliente installando il sito su una macchina cloud con un servizio Tomcat attivo per renderlo accessibile tramite un indirizzo Internet.

Per i pochi giorni che sono rimasti del tirocinio ho potuto monitorare insieme ai miei colleghi l'uso fatto del sito e come interagiva con i dati inviati dal palmare, ed è servito soprattutto a valutare in modo pratico se lo sviluppo del progetto era corretto.

Il sito web è stato progettato per un utilizzo molto semplice ma utile in quelle situazioni dove monitorare le varie fasi di produzione era essenziale. L'utilizzo che ne ha fatto il cliente è quello di stampare solamente etichette con tracciabilità nonostante la possibilità di scelta, questo perché si sono rivelate molto più utili. L'interfaccia di stampa si è rivelata un po' troppo complessa per un primo approccio ma una volta capita la logica di utilizzo si è rivelata molto veloce, anche perché bastano tre campi (tipo etichetta, cliente e articolo) per stampare un'etichetta.

Questi sono dei campi minimi necessari perché nel caso in futuro vengano caricati nuovi layout di etichette possiamo decidere se legarla a un articolo o a un cliente.

Le spunte con le causali sono state utilizzate per diversi scopi. Quelli che abbiamo osservato principalmente sono quello di avere un inventario del magazzino sempre aggiornato, controllo della produzione giornaliera, verificare quale articolo si trova in diverse fasi di produzione e se c'era merce in arrivo da un terzista. Questi utilizzi sono stati tutti resi possibili grazie alle causali che possono essere inserite facilmente nel sito web e poi rese disponibili nel palmare dell'operatore. Questo mi ha

fatto notare come la presenza di un singolo campo possa essere così determinante per una corretta analisi e successo di un progetto.

La cronologia di creazione delle etichette invece si è rivelata superflua perché volume di etichette stampato era molto elevato, e anche se c'erano delle etichette create e stampate per errore non era importante perché non utilizzabili al di fuori di questo progetto a causa del barcode personalizzato.

Invece l'analisi della struttura delle etichette si è rivelata molto completa grazie alla possibilità di inserire molte descrizioni differenti, misure con barcode e immagini.

In conclusione, il progetto si è rivelato utile per lo scopo per cui lo si era progettato anche se in alcuni punti l'analisi andrebbe rivista e alcune funzionalità migliorate.

Bibliografia

[1] DEFINIZIONE DATABASE ORACLE

https://www.oracle.com/it/database/what-is-database/

[2] DEFINIZIONE JAVASCRIPT

https://it.wikipedia.org/wiki/JavaScript

[3] DEFINIZIONE DBMS

https://it.wikipedia.org/wiki/Database_management_system

[4] TOMCAT WIKIPEDIA

https://it.wikipedia.org/wiki/Apache_Tomcat

[5] DEFINIZIONE CSS

https://it.wikipedia.org/wiki/CSS