I.I.S “Benedetto Castelli”

Informatica e Telecomunicazioni

Classe 5^CI

Leonardo Lecci
Daniele Yang

a.s. 2017/2018

ARGOMENTI TRATTATI:

Sistemi e Reti:

- Posta elettronica
- Crittografia
- Firma digitale
- Certificati digitali
- SSL
- HTTPS
- Minacce informatiche

Informatica:

- Database e relativa teoria
- SQL

Educazione Fisica:

- Tattica e Strategia

Premessa

La decisione di sviluppare questo progetto come tesina d'esame nasce dall'idea di Daniele di voler creare un software gestionale per la fatturazione che vada a sostituire quello obsoleto presente nell'azienda dei suoi genitori. Una prima versione del programma di fatturazione era già stata sviluppata da Daniele durante l'inizio del precedente anno scolastico, tuttavia non avendo ancora conoscenze avanzate del linguaggio di programmazione Java e dei database, il programma utilizzava come database fogli di testo txt e le funzionalità erano molto limitate. Ora grazie alle nuove conoscenze acquisite attraverso la scuola e ai periodi di alternanza scuola-lavoro, l'obiettivo di Daniele è quello di creare un gestionale efficiente ed efficace, che sia facile e intuitivo da utilizzare e che integri funzionalità che oggigiorno sono essenziali in un'azienda.
Poco tempo dopo al progetto si è unito Leonardo, che guidato dalla sua passione in ambito economico e volendo sviluppare un'applicazione inerente al suo percorso di studi universitari, ha voluto espandere il progetto creando un sito web che svolga la cosiddetta attività di "Business Intelligence", ovvero vengono applicati concetti della matematica finanziaria per raccogliere dati e analizzare informazioni strategiche per l'azienda.
Il progetto è stato chiamato Kugo Company Manager e, Daniele e Leonardo non si limiteranno a sviluppare tale progetto come "tesina di maturità", ma hanno l'ambizione di continuare a lavorare anche dopo gli esami per farlo diventare un prodotto che riesca a competere nel mercato dei gestionali.

Introduzione

Descrizione

Kugo Company Manager (KCM) è una suite di programmi per la fatturazione e per l'analisi di informazioni strategiche e rappresenta la soluzione ideale per piccole e medie imprese, professionisti e chiunque abbia l'esigenza di emettere fatture e gestire l'azienda in modo semplice e innovativo. Grazie a KCM l'azienda velocizza i processi di produzione, comunicazione e archiviazione di documenti contabili, inoltre tramite avanzate ed attente analisi l'azienda è in grado di capire la miglior strategia di marketing da adottare.

KCM è costituito da due applicazioni tra di loro indipendenti che operano sullo stesso database:

- Kugo Invoice (KI): è un software gestionale per la fatturazione che lavora in **locale**, e permette di creare fatture e gestire le anagrafiche clienti e fornitori;
- Kugo Business Intelligence (KBI): è una **web application** di Business Intelligence, che selezionando dati tramite specifiche interrogazioni al database, è in grado di produrre avanzate analisi aziendali.

Il database utilizzato è comune ad entrambi i software, con la particolarità che Kugo Invoice popola il database, mentre Kugo Business Intelligence seleziona dati, tramite query specifiche, che serviranno poi per ottenere grafici vari ed indicatori che possano aiutare a prendere decisioni importanti a livello aziendale.

Organizzazione del lavoro

Come prima fase abbiamo definito tutti i requisiti a cui il progetto deve essere conforme; successivamente abbiamo steso il progetto concettuale, cercando di circoscrivere al meglio il programma in modo tale da avere meno problemi possibili nella fase di sviluppo; infine abbiamo realizzato il progetto logico e fisico, cioè l'insieme delle componenti software che implementeranno il sistema informatico.

Nel corso della prima fase è necessario raccogliere tutti quegli elementi che servono a definire le caratteristiche che il sistema informatico dovrà avere per soddisfare i bisogni degli utenti: il tempo e il costo di tutto il progetto dipendono dalla qualità del lavoro svolto in questa fase. La documentazione prodotta in questa fase di raccolta costituirà l'input della successiva fase di progettazione concettuale. Questa consiste nell'utilizzare gli elementi raccolti per definire un modello astratto del sistema informatico (progetto concettuale), destinato a definire le linee guida di riferimento e di lavoro per la successiva fase di realizzazione. Si può procedere quindi alla fase di progettazione logica e fisica, che consiste nella realizzazione effettiva del sistema informatico nelle sue varie componenti.

Essendo Kugo Company Manager un progetto molto ampio ed estremamente scalabile la sua produzione è stata suddivisa in due processi, nello specifico per la produzione separata di Kugo Invoice e Kugo BI; il primo è stato sviluppato da Daniele mentre il secondo da Leonardo. Ovviamente entrambi i software sono stati revisionati da tutti e due così da poter ridurre al minimo

eventuali errori, cercare metodi più efficienti e infine anche per individuare miglioramenti da apportare all'aspetto grafico e funzionale dei software.

Il Database

Generalità

Una base di dati (o database) è una collezione di dati gestita da un DBMS, cioè un sistema software in grado di gestire grandi collezioni di dati, assicurando loro affidabilità e privatezza. Come ogni prodotto informatico un DBMS deve essere efficiente ed efficace. I principali punti di forza dell'approccio fondato su DBMS sono i seguenti: integrazione, una base di dati è un insieme integrato di dati strutturati e permanenti memorizzati senza ridondanze superflue e organizzati in modo tale da poter essere usati da applicazioni diverse senza dipendere da alcuna di esse; indipendenza logica, i dati sono definiti indipendentemente dalle procedure che li gestiscono: in questo modo la struttura logica della base di dati può essere ampliata senza la necessità di modificare i programmi applicativi; integrità, è il sistema di gestione delle basi di dati e non le procedure di gestione a dover prevedere meccanismi per controllare che i dati inseriti o modificati soddisfino ai vincoli di integrità specificati.

Query SQL per creare il database

```
CREATE DATABASE kugoinvoice;
```

Come già specificato, KI e KBI utilizzano uno stesso database e per modellare lo scenario della realtà è stato utilizzato il diagramma E/R, che si basa su concetti base che sono le entità, le associazioni e gli attributi.

Le entità rappresentano classi di oggetti che hanno proprietà comuni. In un diagramma E/R ogni entità ha un nome che la identifica univocamente ed è rappresentata graficamente con un rettangolo etichettato con il nome dell'entità e l'elenco dei suoi attributi. Inoltre per ciascuna entità si definisce una chiave primaria (PK) scelta tra le chiavi candidate.

Le associazioni costituiscono il mezzo tramite il quale sono modellate le corrispondenze tra le istanze di due o più entità. Si distinguono associazioni di cardinalità 1:1, 1:N e N:N.

In genere, una volta modellizzato il database si passa alla fase di normalizzazione del database stesso con lo scopo di eliminare eventuali anomalie, che possono essere di inserimento o di selezione, ricorrendo a varie tecniche di normalizzazione tra cui le più note sono: la riduzione in prima forma normale, la riduzione in seconda forma e la riduzione in terza forma normale. Siccome, però siamo stati abituati sin dal principio a progettare database che fossero già normalizzati non abbiamo dovuto svolgere alcun processo di normalizzazione.

Entità

Durante la fase di analisi abbiamo individuato numerose entità (ovvero tabelle), tuttavia essendo il nostro progetto un programma gestionale non tutte le tabelle sono relazionate con altre tabelle, in quanto assumono la funzione di "supporto", ovvero sono tabelle che servono solo a fornire dati e

dunque sono utili per gestire l'applicazione in maniera più efficiente, evitando che l'operatore debba reinserire ogni volta valori uguali. Ad esempio, durante la compilazione di una fattura nell'inserire un metodo di pagamento si possono selezionare tra quelli predefiniti inseriti in precedenza dall'operatore tramite un'apposita schermata. Dunque la tabella "pagamenti" presente sul database non è relazionata ad alcuna altra tabella, ma viene utilizzata solo per fornire dati utili nel compilare una fattura.

Le principali entità individuate sono le seguenti:

Azienda

La tabella "azienda" contiene tutti i dati relativi all'azienda che acquista il pacchetto Kugo Company Manager o solo Kugo Invoice, e che quindi emette le fatture. Questa tabella non è relazionata ad alcuna altra tabella, ma viene utilizzata per recuperare dati dell'azienda quando si vuole creare un report o emettere una fattura.

Query SQL utilizzata per creare la tabella "azienda":

```
CREATE TABLE azienda (id INT PRIMARY KEY AUTO_INCREMENT, rag_soc VARCHAR(80), p_iva VARCHAR(11), cod_fisc VARCHAR(16), indirizzo VARCHAR(100), citta VARCHAR(100), cap VARCHAR(11), prov VARCHAR(2), nazione VARCHAR(100), telefono VARCHAR(50), fax VARCHAR(50), sito_web VARCHAR(80), email VARCHAR(50));
```

Cliente

La tabella "clienti" contiene i dati anagrafici e fiscali relativi ad un cliente. Al momento dell'aggiunta di un nuovo cliente ad esso viene assegnato un **id** che lo identifica univocamente. Tale id sarà dunque chiave primaria dell'entità e non sarà più modificabile.

Query SQL utilizzata per creare la tabella "clienti":

```
CREATE TABLE clienti (cod_cliente INT PRIMARY KEY, rag_soc VARCHAR(80), p_iva VARCHAR(11), cod_fisc VARCHAR(16), indirizzo VARCHAR(100), citta VARCHAR(100), cap VARCHAR(11), prov VARCHAR(2), nazione VARCHAR(100), telefono VARCHAR(50), fax VARCHAR(50), sito_web VARCHAR(80), email VARCHAR(50), agente VARCHAR (80), data_aggiunta TIMESTAMP DEFAULT CURRENT_TIMESTAMP);
```

Luogo Destinazione

Siccome nell'emissione di una fattura, la destinazione della merce potrebbe essere diversa dalla sede legale del Cliente, si ha la necessità di creare un'ulteriore tabella chiamata "luoghi_dest", In questo modo il cliente oltre ad avere l'indirizzo della sede legale, può avere indirizzi aggiuntivi, ad esempio per un magazzino o per un distaccamento. Dunque un cliente può avere N luoghi di destinazione, ma ad un luogo di destinazione è associato uno e un solo cliente.

Query SQL utilizzata per creare la tabella "luoghi_dest":

```
CREATE TABLE luoghi_dest (id INT PRIMARY KEY, descrizione VARCHAR(100),indirizzo VARCHAR(100), citta VARCHAR(100), cap VARCHAR(11), prov VARCHAR(2), nazione VARCHAR(100),
```

```
cod_cliente INT , FOREIGN KEY (cod_cliente) REFERENCES clienti(cod_cliente) ON DELETE CASCADE);
```

La dicitura “ON DELETE CASCADE”, è un vincolo di integrità referenziale di propagazione e significa che quando un’istanza della tabella “clienti” viene eliminata vengono eliminate anche tutte le istanza nella tabella “luoghi_dest” riferite a tale cliente.

Fattura Vendita

La tabella “fatture_vendita” contiene tutti i dati relativi alle fatture emesse dall’azienda, quali il cliente alla quale è stata emessa, il tipo di trasporto, la causale della fattura, i totali, …
Di questa tabella interessanti sono gli attributi stato e contabilizzata: il primo può assumere due valori “pagato” e “non pagato”, e viene poi utilizzato in Kugo Invoice per implementare una funzionalità che consente all’azienda di inviare tramite mail solleciti di pagamento; mentre il secondo attributo assume valori “si” o “no”, e viene utilizzato per bloccare una fattura, ovvero se una fattura è già stata contabilizzata dal commercialista, quest’ultima non sarà più modificabile.
La chiave primaria è **codice**, ed è così formata: anno-numero (es. 2018-22). Si è creata la necessità di creare una PK con anno e numero della fattura in quanto all’inizio di ogni anno la numerazione riparte da 1.

Query SQL utilizzata per creare la tabella “fatture_vendita”:

```
CREATE TABLE fatture_vendita(codice VARCHAR(20) PRIMARY KEY, tipo_doc VARCHAR(50), numero INT, data VARCHAR (20), cod_cliente INT, aspetto_beni VARCHAR(80), ncolli INT, trasporto VARCHAR (80), causale VARCHAR(80), pagamento VARCHAR(80), banca VARCHAR(80), destinazione VARCHAR(500), vettore VARCHAR(80), peso DECIMAL (15,2), volume DECIMAL (15,2), tot_colli INT, netto DECIMAL (15,2), iva DECIMAL (15,2), totale DECIMAL (15,2), note VARCHAR(500),data_aggiunta TIMESTAMP DEFAULT CURRENT_TIMESTAMP, toto_sconto DECIMAL (15,2), tot_lordo DECIMAL (15,2), stato VARCHAR(20), contabilizzata VARCHAR(10), FOREIGN KEY (cod_cliente) REFERENCES clienti(cod_cliente));
```

Riga Fattura

La tabella “righe_fattura_vendita” contiene la parte tabellare di tutte le fatture emesse, ovvero ogni riga della seguente tabella corrisponde ad una riga della fattura. Ogni riga viene associata alla sua corrispondente fattura tramite l’attributo “codice_ft”. Dunque la tabella “righe_fattura_vendita” è in associazione 1:N con “fatture_vendita”.

Query SQL utilizzata per creare la tabella “righe_fattura_vendita”:

```
CREATE TABLE righe_fattura_vendita(id INT AUTO_INCREMENT PRIMARY KEY, codice_ft VARCHAR(20), codice VARCHAR(80), descrizione VARCHAR(100), um VARCHAR(20), qta INT, prezzo DECIMAL (15,2), sconto INT, imponibile DECIMAL (15, 2), iva INT, tot_ivato DECIMAL (15,2), data_aggiunta TIMESTAMP DEFAULT CURRENT_TIMESTAMP, FOREIGN KEY (codice_ft) REFERENCES fatture_vendita (codice) ON DELETE CASCADE ON UPDATE CASCADE)
```

“ON DELETE CASCADE” e “ON UPDATE CASCADE” sono vincoli di integrità referenziale di propagazione e il primo significa che se una fattura viene eliminata, automaticamente vengono eliminate anche le sue righe mentre il secondo significa che quando viene modificata la Primary

Key nella tabella “fatture_vendita” viene modificata di conseguenza anche la Foreign Key alla quale è associato nella tabella “righe_fattura_vendita”.

Prodotto

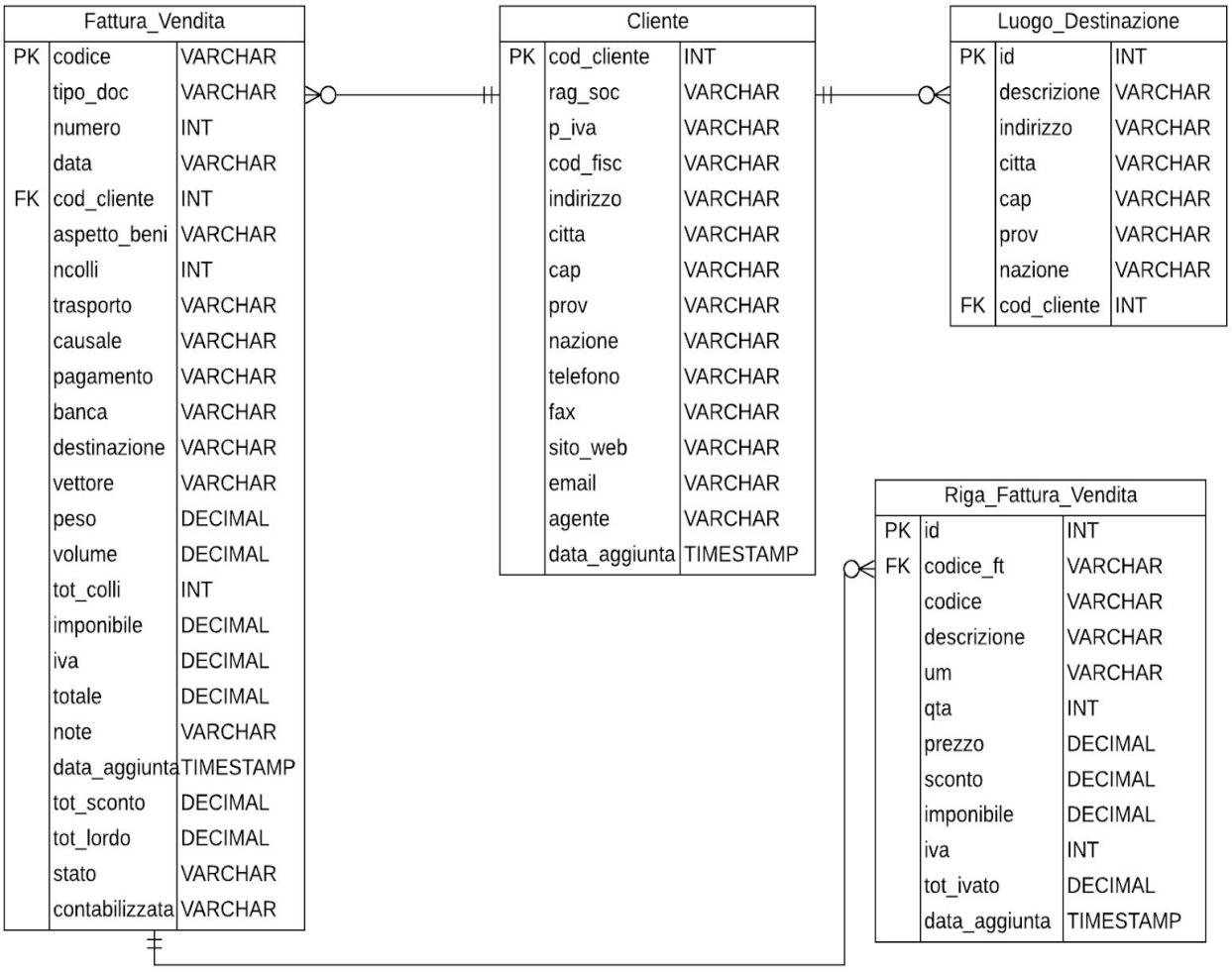
La tabella “prodotti” contiene tutti i prodotti presenti nel magazzino dell’azienda. Non presenta associazioni con altre entità e viene utilizzata per recuperare i prodotti mentre si sta compilando una fattura, in modo da facilitare e semplificare il lavoro all’operatore che potrà scegliere un prodotto tra quelli presenti anziché compilare manualmente tutte le righe ogni volta.

Query SQL utilizzata per creare la tabella “prodotti”:

```
CREATE TABLE prodotti(id INT PRIMARY KEY, codice VARCHAR(80) UNIQUE, ean BIGINT UNIQUE,descrizione VARCHAR(100), prezzo acquisto DECIMAL (15,2), prezzo DECIMAL (15,2), um VARCHAR(20), iva INT, fornitore VARCHAR(80), data_aggiunta TIMESTAMP DEFAULT CURRENT_TIMESTAMP)
```

Diagramma E/R

Di seguito riportiamo il diagramma E/R delle tabelle che sono relazionate tra di loro:



Programmi e tecnologie utilizzate

Per realizzare e mantenere il database abbiamo usato le seguenti tecnologie:

SQL

SQL (Structured Query Language) è un linguaggio standardizzato per database basati sul modello relazionale. SQL è sia un DDL, un DTL che un DQL infatti viene utilizzato per creare e modificare schemi di database, inserire, modificare e gestire dati memorizzati ed interrogare i dati memorizzati.

MySQL

MySQL è un Relational Database Management System (RDBMS). Abbiamo deciso di usare MySQL in quanto esso è un software open-source, rappresenta una delle tecnologie più note e diffuse e grazie alla sua alta capacità di integrazione con i principali linguaggi di programmazione e ambienti di sviluppo.

XAMPP

XAMPP è un pacchetto software gratuito contenente Apache HTTP Server, il database MySQL e tutti gli strumenti necessari per utilizzare linguaggi di programmazione come PHP e Perl. Noi lo abbiamo usato principalmente come web server che ospita il nostro database.

PhpMyAdmin

PhpMyAdmin è un'applicazione web che consente di amministrare un database MySQL non solo attraverso il linguaggio SQL, ma anche tramite un'interfaccia grafica, rendendo l'accesso al database e ai suoi dati molto più semplice.

Connessione al Database

Per la connessione al database sia da KugoInvoice che da Kugo BI è stata creata una classe apposita denominata “DatabaseManager”, in cui vengono settati l’URL del database e le credenziali per accedervi. Quando un metodo deve connettersi al database per effettuare operazioni di aggiunta, modifica o ricerca invoca i metodi return e recupera la connessione al DB.

```
//metodi return nella classe DatabaseManager
public String returnURL(){
   return URL;
   }
public String returnUser(){
   return user;
   }
public String returnPassword(){
   return password;
   }
```

```
//metodo che recupera la connessione al database nelle altre classi del progetto
DatabaseManager db = new DatabaseManager();
Connection con = DriverManager.getConnection(db.returnURL(), db.returnUser(),
db.returnPassword());
```

Kugo Invoice

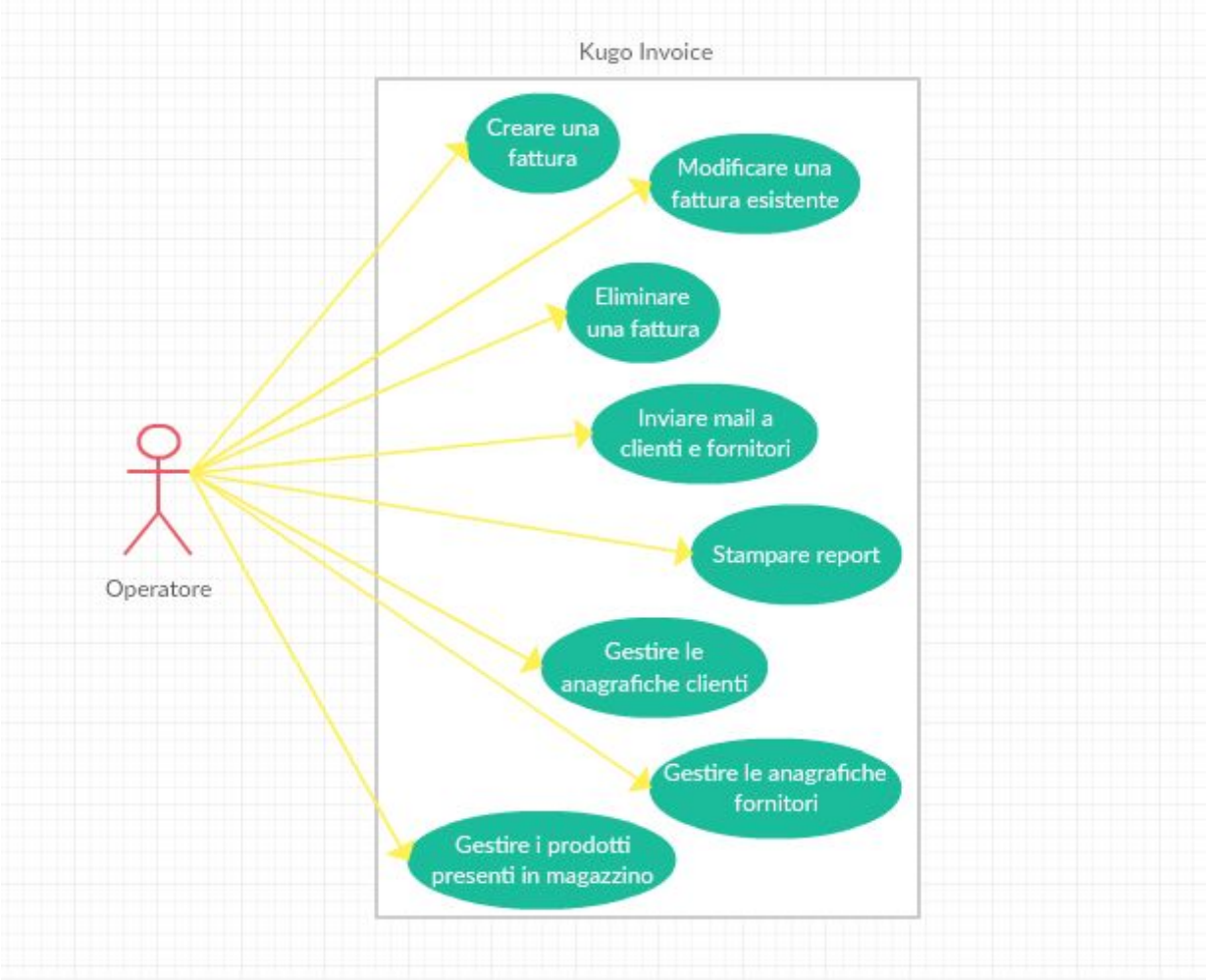
Introduzione

Kugo Invoice (KI) rappresenta una parte del sistema “Kugo Company Manager”, ed è un programma gestionale per la fatturazione che lavora in locale, e permette alle piccole e medie imprese di creare fatture e di gestire le anagrafiche dei clienti e dei fornitori. I punti di forza di Kugo Invoice sono il design semplice, ma intuitivo che consente alle aziende di emettere una fattura in pochi minuti e l'implementazione di avanzate funzionalità per far fronte alle esigenze dettate dal mondo sempre più tecnologico in cui viviamo.

Analisi

Durante la fase di analisi abbiamo prodotto la seguente documentazione:

Use Case Diagram



Gli Use Case Diagram sono diagrammi dedicati alla descrizione delle funzioni o servizi offerti da un sistema, così come sono percepiti e utilizzati dagli attori che interagiscono col sistema stesso. In un diagramma di questo tipo sono presenti tre elementi:

- SISTEMA: Il sistema nel suo complesso è rappresentato come un rettangolo vuoto;
- ATTORI: Gli attori sono rappresentati graficamente nel diagramma da un'icona che rappresenta un uomo stilizzato; essi rappresentano un ruolo coperto da un certo insieme di entità interagenti col sistema;
- USE CASE: Un caso d'uso è rappresentato graficamente come un'ellisse contenente il nome del caso d'uso ed esso rappresenta una funzione o servizio offerto dal sistema a uno o più attori.

Realizzazione

Il progetto "Kugo Invoice" è costituito dai seguenti package:

- com.kugoinvoice.database: contenente la classe DatabaseManager che ritorna la connessione al database ai metodi che la richiedono;
- com.kugoinvoice.gui : contente le schermate dell'applicazione (Graphical User Interface);
- com.kugoinvoice.dialog : contenente i dialog;
- com.kugoinvoice.img: contenente le immagini che verranno usate dall'applicazione;
- com.kugoinvoice.listeners: contenente le classi che implementano gli eventi listeners;
- com.kugoinvoice.service: contenente le classi che implementano i vari metodi che verranno richiamati, ad esempio durante la creazione di una fattura o durante l'inserimento di un nuovo cliente.

Programmi e tecnologie utilizzate

Nel realizzare Kugo Invoice abbiamo utilizzato i seguenti programmi e tecnologie:

Java

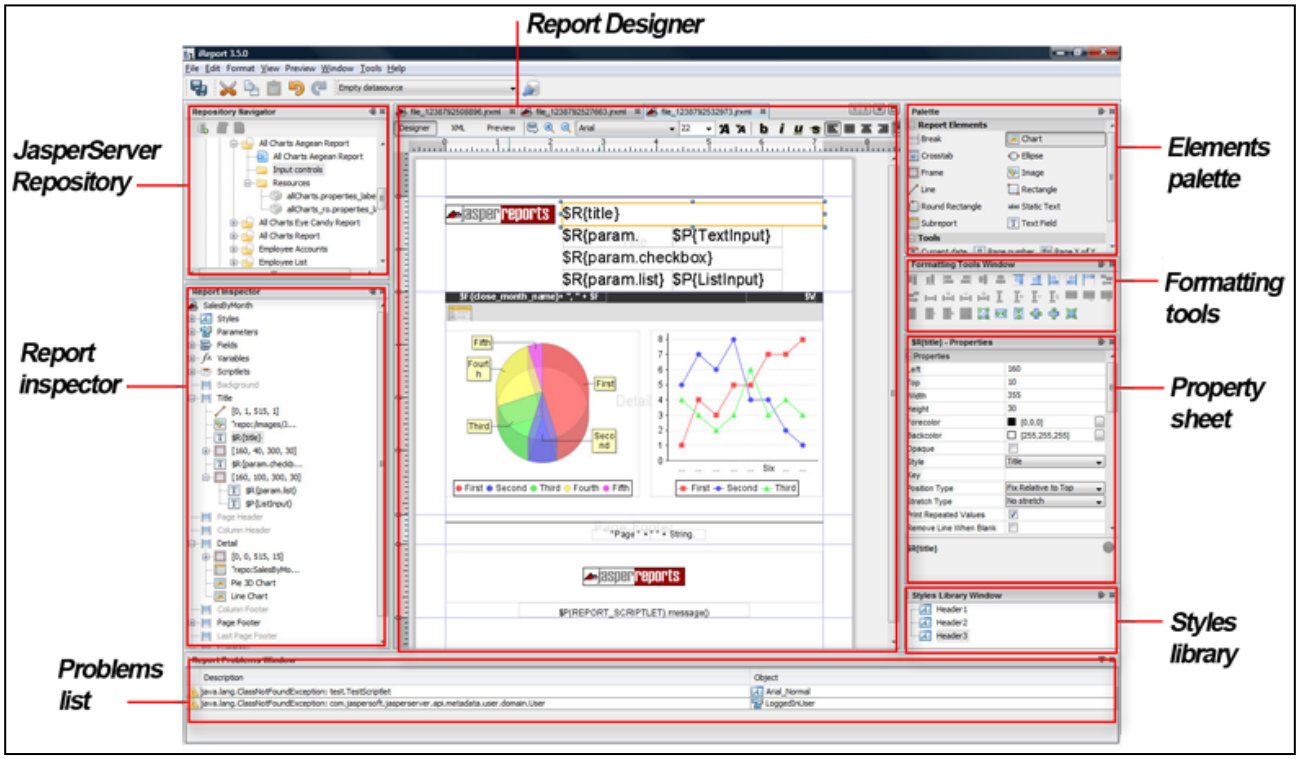
Come unico linguaggio di programmazione abbiamo scelto Java, un linguaggio di orientato agli oggetti creato da James Gosling per Sun Microsystems nel 1995, perché semplice da utilizzare e dispone di molte librerie, anche di terze parti, necessarie per implementare funzionalità come la stampa e l'invio tramite posta elettronica di documenti e messaggi. Java organizza il progetto in "package", ovvero cartelle che contengono delle classi in modo da semplificare l'organizzazione del codice. In genere è buona pratica separare la logica dalla grafica dell'applicazione, dunque in questo caso i package sono utili per dividere la classe dei metodi da quella della grafica. In aggiunta Java è una tecnologia standard che rende Kugo Invoice stabile e multipiattaforma grazie alla JVM (Java Virtual Machine).
Siccome durante quest'ultimo anno scolastico ci siamo concentrati interamente sulla creazione di applicazioni lato web studiando linguaggi come HTML, PHP, Javascript, abbiamo colto quest'occasione per riprendere e approfondire la programmazione in Java.

NetBeans

Come ambiente di sviluppo (IDE) abbiamo scelto NetBeans, dato che è stato ampiamente utilizzato nei progetti scolastici ed è provvisto di numerosi plug-in che lo rendono utile ad ogni necessità.

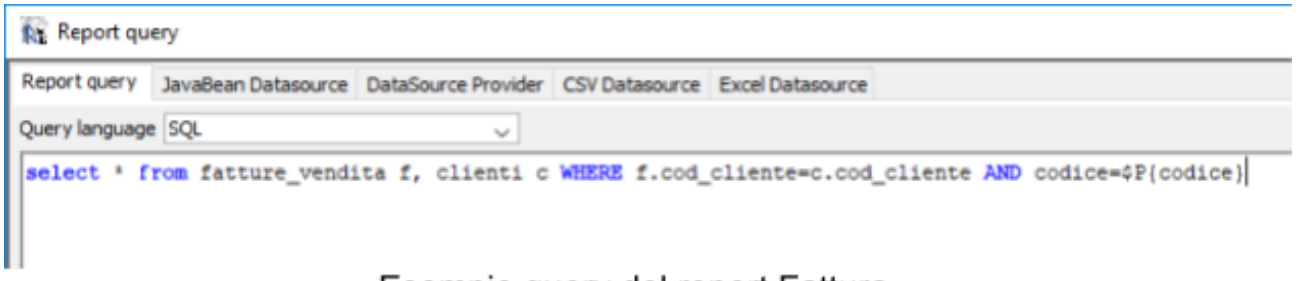
iReport Designer

iReport è un visual report designer per la libreria JasperReports, progettato specificatamente per essere integrato con NetBeans. Questo programma permette di creare dei report in modo molto semplice, senza dover scrivere alcuna riga di codice; infatti dopo aver impostato la connessione al database, dal quale verranno presi i dati per popolare il report, basta effettuare il drag and drop dei componenti (es. testo statico, tabella, ecc.) nell'area di lavoro e settare le opzioni per avere un report (in formato .jrxml) pronto da integrare nell'applicazione.



Creare un report

Dopo aver impostato la connessione al database, bisogna effettuare una query al database per recuperare i dati che popoleranno il report.



Esempio query del report Fattura

$P{codice} è un parametro che viene passato dal programma al report builder, cosicché si recuperino tutti i dati relativi ad una data fattura.
La query ritorna tutti gli attributi selezionati sotto forma di "field", ad esempio l'attributo numero della fattura sarà $F{numero}:

DOCUMENTO	$F{tipo_doc}	P. IVA	$F{p_iva}	C.F.	$F{cod_fisc}
NUMERO	$F{numero}	DATA	$F{data}	PAGAMENT	$F{pagamento}	COD CLIENTE	$F

Creazione del report

DOCUMENTO	Fattura	P. IVA	01214840223	C.F.	01214840223
NUMERO	36	DATA	05/06/2018	PAGAMENTO	CONTANTI	COD CLIENTE	21

Anteprima report

Principali librerie esterne utilizzate

Nel realizzare Kugo Invoice sono state utilizzate varie librerie di terze parti per implementare funzionalità come l'invio della mail e la stampa.

JavaMail

La libreria javax.mail fornisce le API JavaMail che permettono di inviare e ricevere email tramite i protocolli SMTP, POP3, IMAP. Nel progetto abbiamo utilizzato questa libreria per permettere all'azienda di inviare mail, con o senza allegati, ai propri clienti e fornitori.

```
//metodo della classe MailService per inviare messaggi senza allegati
public void send(String to, String sub, String msg) {
        try {
            //recupero credenziali per accedere alla casella di posta e configurazioni dal DB
            String user = returnUtente();
            String pass = returnPassword();
            String host = returnHost();
            String porta = returnPorta();
            //setto le proprietà della connessione
            Properties props = new Properties();
            props.put("mail.smtp.auth", "true");
            props.put("mail.smtp.starttls.enable", "true");
            props.put("mail.smtp.host", host);
            props.put("mail.smtp.port", porta);
            //creo la sessione e autentico il mittente
            Session session = Session.getDefaultInstance(props, new Authenticator() {
                @Override
                protected PasswordAuthentication getPasswordAuthentication() {
                    return new PasswordAuthentication(user, pass);
                }
            });
            //compongo le parti del messaggio
            try {
                Message message = new MimeMessage(session);
                message.setFrom(new InternetAddress(user));
                message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(to));
                message.setSubject(sub);
                message.setText(msg);
                //invio il messaggio
                Transport.send(message);
                JOptionPane.showMessageDialog(null, "Email inviata con successo");
            } catch (MessagingException e) {
                    throw new RuntimeException(e);
            }
        } catch (SQLException ex) {
            Logger.getLogger(MailService.class.getName()).log(Level.SEVERE, null, ex);
        }
}
```

JasperReports

JasperReports è una libreria di reportistica open source che può essere integrata in qualsiasi applicazione Java. I report JasperReports sono definiti nel formato XML chiamato JRXML, i quali possono essere modificati a mano, generati oppure disegnati usando dei tool appositi (iReport Designer). E' possibile la visualizzazione di report, la loro stampa e il salvataggio nei formati più comuni come HTML, PDF, Excel, OpenOffice, Word, XML.

Di seguito, porzione di codice che permette di creare un report dalla Java Application e di esportarlo nel formato PDF:

```
//creo la connessione al DB
Connection con = DriverManager.getConnection(db.returnURL(), db.returnUser(), db.returnPassword());
//richiamo il file .jrxml
InputStream is = new FileInputStream(new File("./ireport/Lista_Clienti.jrxml"));
JasperDesign jd = JRXmlLoader.load(is);
JasperReport jr = JasperCompileManager.compileReport(jd);
//dopo aver caricato il file .jrxml faccio creare il report
JasperPrint jp = JasperFillManager.fillReport(jr, null, con);
//salvo il report nel formato PDF
OutputStream os = new FileOutputStream(new File(opzionis.returnReportPath() + "\\Lista_Clienti.pdf"));
JasperExportManager.exportReportToPdfStream(jp, os);
os.close();
JOptionPane.showMessageDialog(null, "Documento esportato in PDF come Lista_Clienti.pdf");
```

Nel precedente esempio al report non si è passato alcun parametro, in quanto bisognava semplicemente creare una lista di tutti i clienti. Ma se si dovesse creare una fattura, allora sarebbe necessario passare al report il codice della fattura affinché la query sia in grado di recuperare i dati relativi a tale fattura.

In tal caso si crea un HashMap contenente i parametri necessari e si passa la HashMap al JasperFillManager assieme al report e alla connessione.

```
HashMap param = new HashMap();
param.put("codice", pk);
JasperReport jr = JasperCompileManager.compileReport(jd);
JasperPrint jp = JasperFillManager.fillReport(jr, param, con);
```

Spring Security

La libreria org.springframework fornisce tutte le funzionalità di Spring Security, un framework che fornisce servizi di autenticazione e autorizzazione alle applicazioni Java. Ho scelto di utilizzare questa libreria in quanto il punto di forza di Spring Security è la facilità con cui può essere esteso per soddisfare i requisiti dell'utente. Di questa libreria in particolare è stata utilizzata la classe BCrypt, che è una funzione di hash (o hashing) di password basata sulla cifratura Blowfish. Oltre a incorporare un sale per proteggere la password contro attacchi "tabella arcobaleno", BCrypt è una funzione adattiva: col tempo, il conteggio dell'iterazione può essere aumentato per renderla più lenta, in modo da essere resistente ad attacchi di forza bruta anche con capacità computazionale crescente. Abbiamo scelto di utilizzare la funzione di hash e non un algoritmo di crittografia, in quanto la funzione di hash non è invertibile, quindi dato un hash non è possibile risalire al testo originale. Questo rappresenta sicuramente un punto di forza, dato che la "plain" password non viene mai salvata sul database, ma solo il "fingerprint". Quando si deve verificare se la password

digitata dall'utente è corretta, si calcola la "fingerprint" della password inserita e la si confronta con quella presente nel database.
Come già accennato è molto semplice effettuare l'hashing delle password con BCrypt:

```
//con questo metodo creo l'Hash di una password in chiaro
public String createHash(String plain_password){
        String pw_hash = BCrypt.hashpw(plain_password, BCrypt.gensalt());
        return pw_hash;
    }

//questo metodo controlla che la password inserita dall'utente combaci con l'Hash presente nel database
public boolean checkPassword(String user, String candidate_password){
        boolean correct = false;
        //viene recuperato l'Hash presente nel DB
        String stored_hash=returnStoredHash(user);
        //viene confrontata la password con l'Hash
        if (BCrypt.checkpw(candidate_password, stored_hash)){
            correct=true;
        }else{
            correct=false;
        }
        return correct;
    }
```

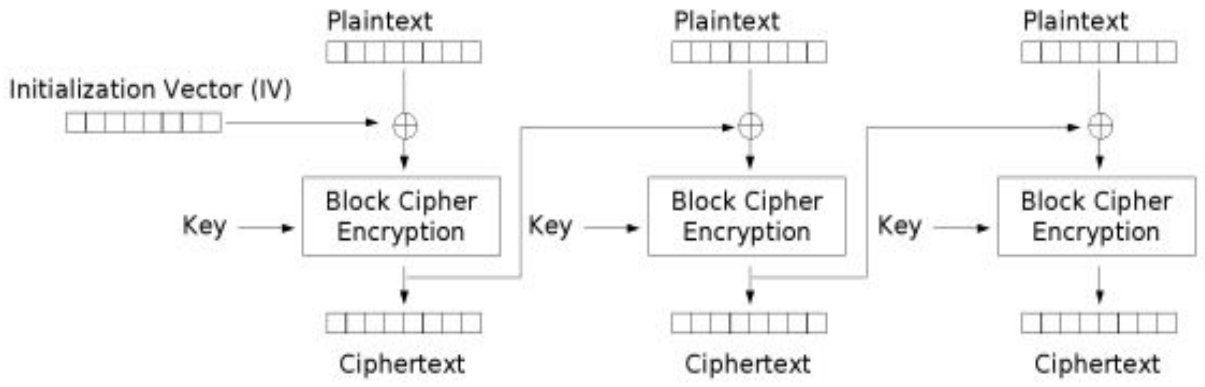
Apache Commons

Apache Commons è un progetto di Apache Foundation ed è costituito da una serie di librerie di utilizzo comune, pronte per essere incluse nei progetti Java. In questo modo il programmatore non deve implementare da zero alcune funzionalità, ma può semplicemente importarle dalle librerie Apache, riducendo notevolmente la mole di lavoro. Per il progetto è stata utilizzata la classe Base64, della libreria "org.apache.commons.codec", che è un sistema di codifica che consente la traduzione di dati binari in stringhe di testo ASCII, rappresentando i dati sulla base di 64 caratteri ASCII diversi. In questo caso Base64 viene utilizzato per convertire la password della casella di posta elettronica cifrata con AES in un coded Base64. Il motivo per cui abbiamo scelto di utilizzare AES, che è un algoritmo di crittografia a chiave simmetrica, anziché una funzione di Hash come BCrypt per salvare la password della mail verrà ampiamente spiegato nel paragrafo "Sicurezza".

Sicurezza

Siccome le risorse più importanti di ogni azienda sono i dati, dato che grazie a questi le aziende operano sui mercati e prendono decisione tattiche e strategiche, Kugo Invoice assicura che tutte le password inserite per accedere alla piattaforma Kugo Business Intelligence e alla casella di posta siano criptate, per proteggersi da eventuali attacchi mirati a ottenere tali credenziali per poi rubare dati aziendali (spionaggio industriale). Come già accennato in precedenza, mentre per salvare la password per accedere alla piattaforma Kugo Business Intelligence e alle aree riservate all'interno di Kugo Invoice si utilizza BCrypt (che come ho detto è una funzione di Hash), per salvare la password della casella di posta elettronica, si ricorre ad un algoritmo di crittografia a chiave simmetrica, che prevede un'unica chiave sia per la cifratura che per la decifratura. Questa scelta è dettata dalla necessità di dover passare alla sessione instaurata con il mail server la password in chiaro, e siccome la funzione di Hash non è invertibile abbiamo optato per un algoritmo di crittografia simmetrico. Durante la sessione di autenticazione il problema dell'intercettazione delle credenziali viene risolto in quanto viene automaticamente implementato il protocollo SSL (acronimo di Secure Socket Layer) che garantisce la privacy delle trasmissioni che avvengono

sulla rete Internet. Sostanzialmente utilizzando SSL i dati vengono criptati con algoritmo di crittografia simmetrici, mittente e destinatario vengono autenticati con algoritmi di crittografia asimmetrici e viene infine incluso un ulteriore controllo sull'integrità dei messaggi.
In particolare abbiamo utilizzato AES (acronimo di Advanced Encryption Standard) che fu scelto verso la fine degli anni Novanta come nuovo standard crittografico per sostituire l'ormai insicuro DES. AES fu progettato sulla base di tre caratteristiche fondamentali: resistenza contro tutti i tipi di attacchi; velocità e compattezza del codice su un ampia gamma di piattaforme; semplicità progettuale. Si tratta di un cifrario a blocchi con lunghezza del blocco di 128 bit, e chiavi di cifratura che possono essere di 128, 192 o 256 bit, ed effettua una combinazione di sostituzioni e permutazioni. In questo caso AES viene utilizzato in modalità CBC (acronimo di Cipher Block Chaining) che è più veloce più sicuro ed è raccomandato per i servizi di autenticazione. Funzionalmente, la modalità CBC garantisce che se uno stesso blocco in chiaro viene ripetuto, vengono prodotti blocchi di testo cifrati differenti.



Cipher Block Chaining (CBC) mode encryption

Esempio di criptaggio e decriptaggio con AES in modalità CBC:

```
//Si deve inizializzare l'initVector con una stringa di 128 bit
private final String initVector = "encryptionIntVec";
//encryption
public String EncryptAES(String value) {
        try {
            //recupero la chiave segreta
            String secretKey = getAESSecretKey();
            IvParameterSpec iv = new IvParameterSpec(initVector.getBytes("UTF-8"));
            SecretKeySpec skeySpec = new SecretKeySpec(secretKey.getBytes("UTF-8"), "AES");

            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec, iv);

            byte[] encrypted = cipher.doFinal(value.getBytes());
            return Base64.encodeBase64String(encrypted);

        } catch (Exception ex) {
            ex.printStackTrace()
        }
        return null;
    }
```

```
//decryption
public String DecryptAES(String encrypted) {
        try {
            String secretKey = getAESSecretKey();
            IvParameterSpec iv = new IvParameterSpec(initVector.getBytes("UTF-8"));
            SecretKeySpec skeySpec = new SecretKeySpec(secretKey.getBytes("UTF-8"), "AES");

            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
            cipher.init(Cipher.DECRYPT_MODE, skeySpec, iv);
            byte[] original = cipher.doFinal(Base64.decodeBase64(encrypted));

            return new String(original);
        } catch (Exception ex) {
            ex.printStackTrace()
        }

        return null;
    }
```

La chiave segreta di 128 bit viene salvata in “passphrase.txt” all’interno della cartella del progetto e recuperata tramite il metodo getAESSecretKey().

Commenti nel codice - JavaDoc

Il punto di forza di un progetto informatico oltre ad un codice efficiente ed efficace è sicuramente la descrizione del codice stesso, in modo tale da permettere a chi modifica o revisiona il codice di comprendere il significato di ciascuna riga. In Java esiste Javadoc, che è un applicativo incluso nel JDK (Java Development Kit), utilizzato per la generazione automatica della documentazione del codice sorgente.

Principali funzionalità

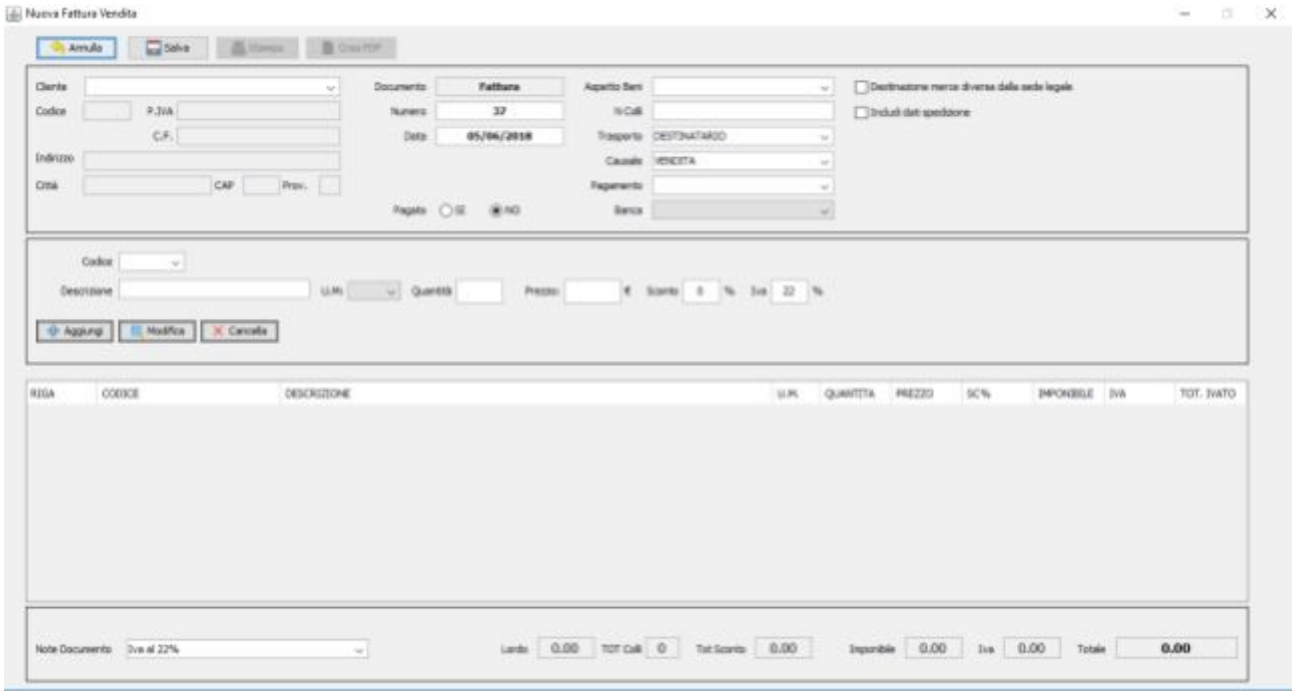
La schermata principale di Kugo Invoice è molto intuitiva.
Presenta un menù bar, che consente di accedere a tutte le funzioni del programma, in particolar modo a quelle avanzate ovvero gestire i comuni, le nazioni, i codici iva, le modalità di pagamento, ecc.
Vengono poi presentate le funzionalità principali sotto forma di bottoni che sono:

1. Creazione di una nuova fattura;
2. Gestire le fatture già emesse;
3. Gestire le anagrafiche dei clienti;
4. Gestire le anagrafiche dei fornitori;
5. Gestire i prodotti nel magazzino;
6. Accedere alla piattaforma Kugo Business Intelligence, la quale verrà ampiamente spiegata nel prossimo paragrafo.

In aggiunta, dal menu bar si potrà accedere alle seguenti sezioni:

- Dati aziendali
- Conti correnti aziendali
- Tabella dei codici Iva
- Tabella delle unità di misura
- Tabella delle nazioni
- Tabella dei comuni
- Tabella delle causali nei documenti
- Tabella dei vettori
- Tabella del tipo di porto (ad esempio. assegnato o franco)
- Tabella degli aspetti esteriori dei beni
- Tabella delle modalità di trasporto
- Tabella delle modalità di pagamento
- Impostazioni generali (area protetta da password)
- Impostazioni della posta elettronica (area protetta da password)
- Richiesta assistenza

Creazione di una nuova fattura



All'apertura della finestra di creazione della fattura, verranno automaticamente compilate la data e il numero (che ovviamente è progressivo). Questa finestra è molto semplice e intuitiva infatti presenta tutti i campi da compilare su un'unica schermata, in modo che l'operatore riesca a prima vista a vedere se ha dimenticato o meno di compilarne qualcuno, riducendo notevolmente il tasso di errore/dimenticanza.
L'operatore può scegliere dal menù a tendina in alto a sinistra il cliente e automaticamente verranno compilati tutti i campi relativi a tale cliente quali indirizzo, partita iva, codice fiscale, ecc.
Inoltre se si vuole inserire un indirizzo di destinazione diverso dalla sede legale, è sufficiente cliccare sul checkbox "Destinazione merce diversa dalla sede legale" e selezionare dal menù a tendina quello desiderato.
L'iva di default sarà quella impostata in "impostazioni" o in alternativa quella definita nella fase di inserimento dei prodotti. Ma l'utente sarà anche libero di modificare l'iva in quanto il campo non è stato settato come "non editable".
Successivamente l'operatore può compilare la parte tabellare scegliendo i prodotti tra quelli già presenti nel database o inserendone uno nuovo nel pannello centrale. Per modificare una riga della fattura è sufficiente cliccare sulla riga e ricompilare le descrizione nel pannello centrale e confermare cliccando sul bottone "modifica"; mentre per eliminare una riga bisogna cliccare su di essa e cliccare sul bottone "elimina".
Una volta che la fattura è stata compilata allora questa dovrà essere salvata cliccando sull'apposito bottone e solo dopo sarà possibile procedere alla stampa o all'esportazione nel formato PDF.
Durante l'atto di salvataggio il sistema controlla che la data sia corretta e che il numero della fattura sia valido, infatti questi due parametri verranno passati a dei metodi che ne controllano la correttezza.

```
//viene controllato che la data sia giusta, ovvero che sia nel formato gg/mm/aaaa
public boolean isThisDateValid(String dateToValidate) {
        String dateFromat = "dd/MM/yyyy";
```

```
        //se la stringa data è nulla allora ritorna falso
        if (dateToValidate == null) {
            return false;
        }
        SimpleDateFormat sdf = new SimpleDateFormat(dateFromat);
        sdf.setLenient(false);
        try {
            //se il formato della data non è corretto si crea un eccezione
            Date date = sdf.parse(dateToValidate);
        } catch (ParseException e) {
            //ritorna false perchè il formato non è corretto
            return false;
        }
        //altrimenti significa che il formato è corretto
        return true;
    }
```

Controllata che la data sia corretta viene calcolata la Primary Key della fattura e viene controllato che una fattura con quell'anno e quel numero non sia già stata emessa.

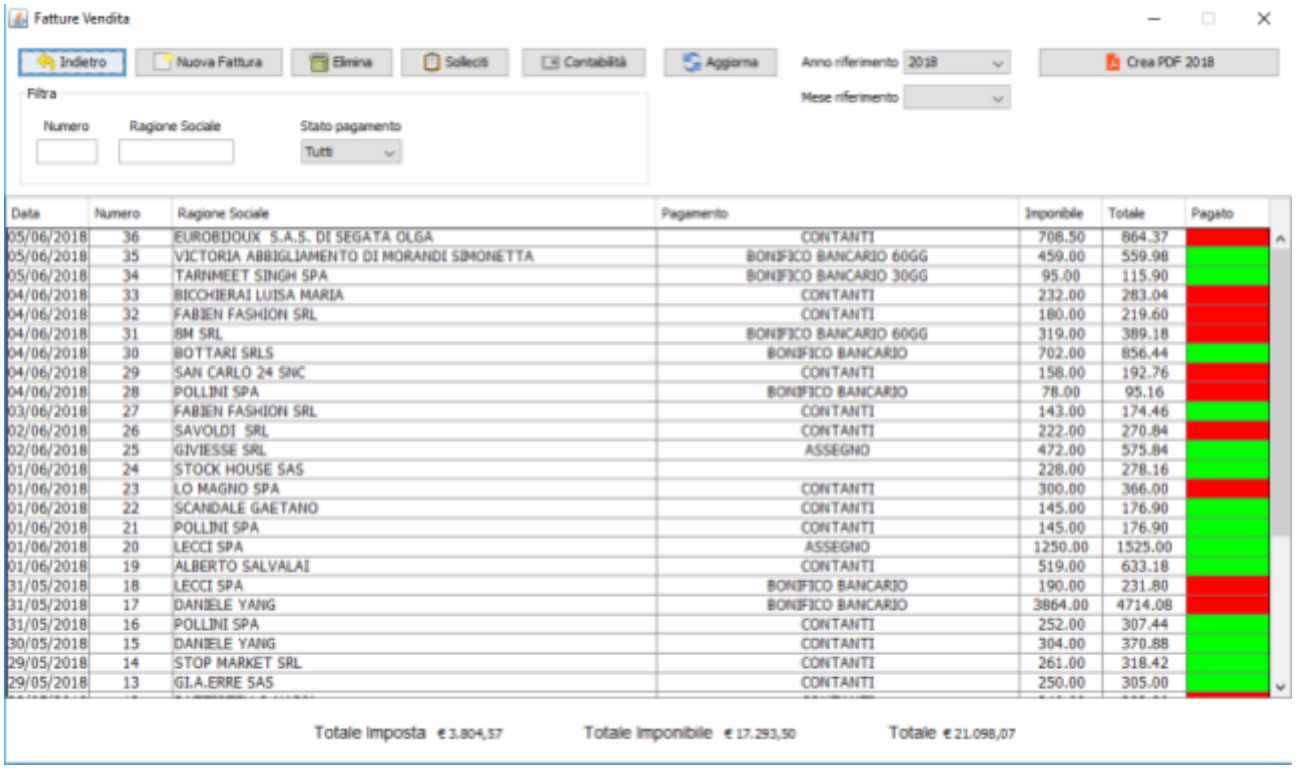
```
//dati il numero della fattura e la data viene calcolata la PK nel formato anno-numero
public String returnInvoicePK(String id, String data) {
        String pk = null;
        //si estrae l'anno dalla data
        int index = data.lastIndexOf("/");
        String anno = data.substring(index + 1, data.length());
        pk = anno + "-" + id;
        return pk;
    }

//viene poi controllato che la PK della fattura sia valido
public boolean isInvoicePKCorrect(String id) throws SQLException {
        //si effettua una connessione al database e si controlla che la PK da inserire non sia già presente
        Connection con = DriverManager.getConnection(db.returnURL(), db.returnUser(), db.returnPassword());
        PreparedStatement st = con.prepareStatement("SELECT codice FROM `fatture_vendita`");
        ResultSet resultSet = st.executeQuery();
        while (resultSet.next()) {
            String id_db = resultSet.getString("codice");
            if (id.equals(id_db)) {
                //se c'è una corrispondenza allora si tratta di un duplicato e dunque ritorna false
                return false;
            }
        }
        con.close();
        return true;
    }
```

L'operatore potrà poi anche inviare la fattura via mail al cliente, in questo caso dovrà accedere alla sezione "gestione delle fatture" e cliccare sulla fattura da inviare. Nel caso in cui accedendo ad una fattura si nota che il bottone "invia mail" è disabilitato significa che per il cliente non è stata inserita la mail.

Gestione delle fatture

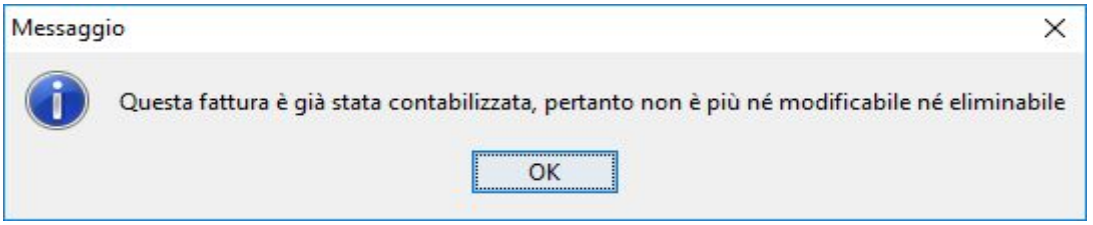
Nella schermata di gestione delle fatture, l'operatore può modificare una fatture emessa se questa non è già stata contabilizzata, modificare lo stato di pagamento della fattura accedendo alla stessa (verde se è già stata pagata mentre rosso se è da pagare).



Data	Numero	Ragione Sociale	Pagamento	Imponibile	Totale	Pagato
05/06/2018	36	EUROBIDOUX S.A.S. DI SEGATA OLGA	CONTANTI	708.50	864.37	
05/06/2018	35	VICTORIA ABBIGLIAMENTO DI MORANDI SIMONETTA	BONIFICO BANCARIO 60GG	459.00	559.98	
05/06/2018	34	TARNMEET SINGH SPA	BONIFICO BANCARIO 30GG	95.00	115.90	
04/06/2018	33	BICCHIERAI LUISA MARIA	CONTANTI	232.00	283.04	
04/06/2018	32	FABIEN FASHION SRL	CONTANTI	180.00	219.60	
04/06/2018	31	BM SRL	BONIFICO BANCARIO 60GG	319.00	389.18	
04/06/2018	30	BOTTARI SRLS	BONIFICO BANCARIO	702.00	856.44	
04/06/2018	29	SAN CARLO 24 SNC	CONTANTI	158.00	192.76	
04/06/2018	28	POLLINI SPA	BONIFICO BANCARIO	78.00	95.16	
03/06/2018	27	FABIEN FASHION SRL	CONTANTI	143.00	174.46	
02/06/2018	26	SAVOLDI SRL	CONTANTI	222.00	270.84	
02/06/2018	25	GIVIESSE SRL	ASSEGNO	472.00	575.84	
01/06/2018	24	STOCK HOUSE SAS		228.00	278.16	
01/06/2018	23	LO MAGNO SPA	CONTANTI	300.00	366.00	
01/06/2018	22	SCANDALE GAETANO	CONTANTI	145.00	176.90	
01/06/2018	21	POLLINI SPA	CONTANTI	145.00	176.90	
01/06/2018	20	LECCI SPA	ASSEGNO	1250.00	1525.00	
01/06/2018	19	ALBERTO SALVALAI	CONTANTI	519.00	633.18	
31/05/2018	18	LECCI SPA	BONIFICO BANCARIO	190.00	231.80	
31/05/2018	17	DANIELE YANG	BONIFICO BANCARIO	3864.00	4714.08	
31/05/2018	16	POLLINI SPA	CONTANTI	252.00	307.44	
30/05/2018	15	DANIELE YANG	CONTANTI	304.00	370.88	
29/05/2018	14	STOP MARKET SRL	CONTANTI	261.00	318.42	
29/05/2018	13	GI.A.ERRE SAS	CONTANTI	250.00	305.00	

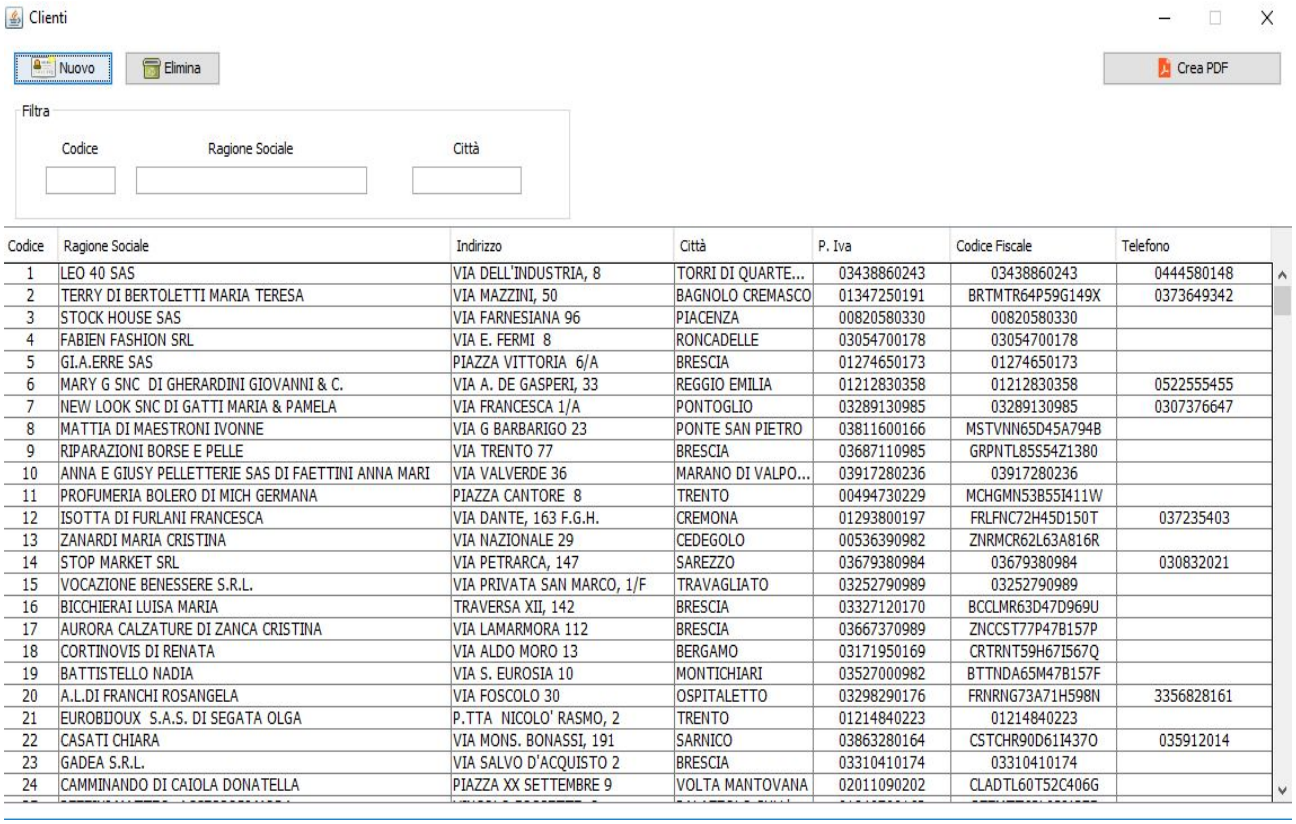
Inoltre accedendo all'area "solleciti", cliccando sul relativo bottone, si potrà inviare una mail standard di sollecito pagamento ai clienti.

Oltre al messaggio viene allegata anche la fattura da pagare per semplificare le operazioni al cliente, nel caso avesse perso la fattura.
Accedendo invece all'area "contabilità" è possibile bloccare le fatture che sono già state elaborate dal commercialista, dunque le fatture contabilizzate non potranno più essere modificate. Quando si apre una fattura contabilizzata il bottone modifica verrà disabilitato e comparirà il seguente messaggio:



Infine si può esportare una lista delle fatture annuali o mensili, per velocizzare la contabilità. Ad esempio per il calcolo mensile o trimestrale dell'IVA da pagare.

Gestione anagrafiche clienti e fornitori

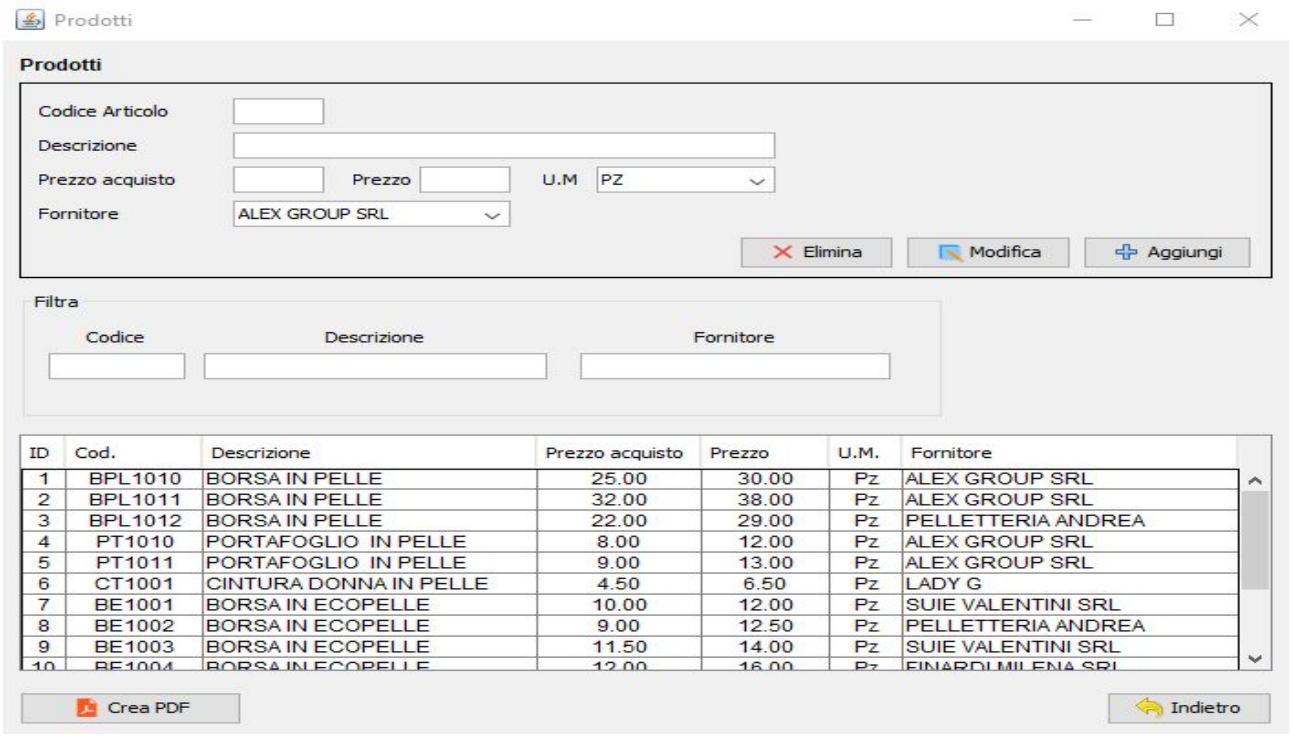


Codice	Ragione Sociale	Indirizzo	Città	P. Iva	Codice Fiscale	Telefono
1	LEO 40 SAS	VIA DELL'INDUSTRIA, 8	TORRI DI QUARTE...	03438860243	03438860243	0444580148
2	TERRY DI BERTOLETTI MARIA TERESA	VIA MAZZINI, 50	BAGNOLO CREMASCO	01347250191	BRTMTR64P59G149X	0373649342
3	STOCK HOUSE SAS	VIA FARNESIANA 96	PIACENZA	00820580330	00820580330	
4	FABIEN FASHION SRL	VIA E. FERMI 8	RONCADELLE	03054700178	03054700178	
5	GI.A.ERRE SAS	PIAZZA VITTORIA 6/A	BRESCIA	01274650173	01274650173	
6	MARY G SNC DI GHERARDINI GIOVANNI & C.	VIA A. DE GASPERI, 33	REGGIO EMILIA	01212830358	01212830358	0522555455
7	NEW LOOK SNC DI GATTI MARIA & PAMELA	VIA FRANCESCA 1/A	PONTOGLIO	03289130985	03289130985	0307376647
8	MATTIA DI MAESTRONI IVONNE	VIA G BARBARIGO 23	PONTE SAN PIETRO	03811600166	MSTVNN65D45A794B	
9	RIPARAZIONI BORSE E PELLE	VIA TRENTO 77	BRESCIA	03687110985	GRPNTL85S54Z1380	
10	ANNA E GIUSY PELLETTERIE SAS DI FAETTINI ANNA MARI	VIA VALVERDE 36	MARANO DI VALPO...	03917280236	03917280236	
11	PROFUMERIA BOLERO DI MICH GERMANA	PIAZZA CANTORE 8	TRENTO	00494730229	MCHGMN53B55I411W	
12	ISOTTA DI FURLANI FRANCESCA	VIA DANTE, 163 F.G.H.	CREMONA	01293800197	FRLFNC72H45D150T	037235403
13	ZANARDI MARIA CRISTINA	VIA NAZIONALE 29	CEDEGOLO	00536390982	ZNRMCR62L63A816R	
14	STOP MARKET SRL	VIA PETRARCA, 147	SAREZZO	03679380984	03679380984	030832021
15	VOCAZIONE BENESSERE S.R.L.	VIA PRIVATA SAN MARCO, 1/F	TRAVAGLIATO	03252790989	03252790989	
16	BICCHIERAI LUISA MARIA	TRAVERSA XII, 142	BRESCIA	03327120170	BCCLMR63D47D969U	
17	AURORA CALZATURE DI ZANCA CRISTINA	VIA LAMARMORA 112	BRESCIA	03667370989	ZNCCST77P47B157P	
18	CORTINOVIS DI RENATA	VIA ALDO MORO 13	BERGAMO	03171950169	CRTRNT59H67I567Q	
19	BATTISTELLO NADIA	VIA S. EUROSIA 10	MONTICHIARI	03527000982	BTTNDA65M47B157F	
20	A.L.DI FRANCHI ROSANGELA	VIA FOSCOLO 30	OSPITALETTO	03298290176	FRNRNG73A71H598N	3356828161
21	EUROBIJOUX S.A.S. DI SEGATA OLGA	P.TTA NICOLO' RASMO, 2	TRENTO	01214840223	01214840223	
22	CASATI CHIARA	VIA MONS. BONASSI, 191	SARNICO	03863280164	CSTCHR90D61I437O	035912014
23	GADEA S.R.L.	VIA SALVO D'ACQUISTO 2	BRESCIA	03310410174	03310410174	
24	CAMMINANDO DI CAIOLA DONATELLA	PIAZZA XX SETTEMBRE 9	VOLTA MANTOVANA	02011090202	CLADTL60T52C406G	

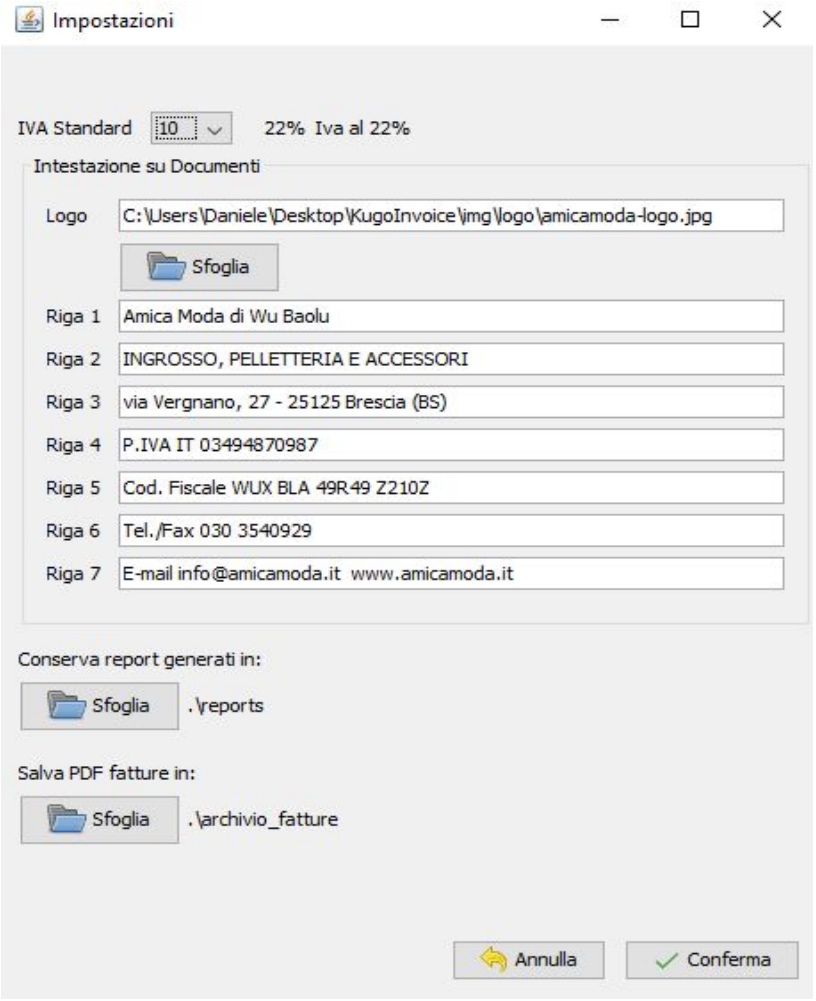
Le finestre di gestione dei clienti e dei fornitori sono simili, entrambe permettono di inserire una nuova anagrafica e di esportare in PDF una lista delle anagrafiche. Le uniche due differenze sono che nell'aggiungere un cliente si possono aggiungere anche dei luoghi di destinazione della merce alternative alla sede legale, mentre nell'aggiungere un fornitore si possono aggiungere anche eventuali coordinate bancarie.
Dopo che è stata inserita un'anagrafica è possibile modificare/aggiornare le informazioni ad essa connesse selezionando semplicemente l'anagrafica dalla tabella. Sempre da questa schermata è possibile inoltre inviare una mail al cliente/fornitore.
Non è stato inserito alcun tipo di controllo sulla né sulla Partita Iva né sul Codice Fiscale in quanto Kugo Invoice, come molti altri gestionali vuole lasciare una certa "libertà" ai suoi utilizzatori.

Gestione dei prodotti

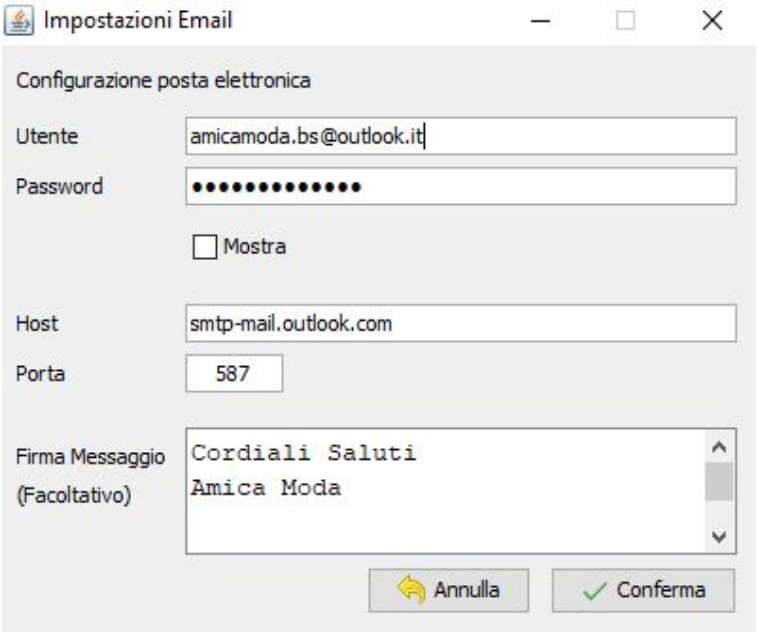
Nella sezione gestione dei prodotti, l'azienda può inserire gli articoli che sono presenti in magazzino ed esportare la lista nel formato PDF. Al momento la gestione dei prodotti è molto semplice dato che non è stato implementato il carico e scarico del magazzino, che sarà una funzionalità che vorrei aggiungere in futuro.
Il pannello per aggiungere nuovi prodotti è molto semplice da utilizzare, infatti basta compilare tutti i campi, selezionare il fornitore dal menu a tendina e cliccare sul bottone "Aggiungi"; per modificare invece un prodotto bisogna cliccare nella tabella la riga corrispondente e poi dal pannello compilare i campi e cliccare sul bottone "Modifica"; invece per eliminare un prodotto, dopo aver cliccato la riga corrispondente bisogna fare clic sul bottone "Elimina"



Impostazioni generali e mail



Agli utenti amministratori verranno rilasciate delle credenziali per accedere ad aree riservate all'interno di Kugo Invoice, che consentirà loro di modificare il codice iva in uso, aggiornare le righe dell'intestazione da salvare sulla fattura e sui vari documenti generati e impostare le cartelle predefinite in cui salvare i file PDF.



Sempre attraverso le credenziali, gli amministratori potranno aggiornare le credenziali per l'accesso alla posta elettronica e le relative configurazioni. Tuttavia la modifica di quest'area, in particolar modo i campi "host" e "porta", è consigliata solo agli esperti, perché l'incorrettezza di uno solo di questi campi compromette il funzionamento della posta elettronica.

Sviluppi futuri

Come si sa un programma informatico non si può mai dire "completo", infatti si dice che il processo di progettazione di un sistema informatico è ciclico perché con l'uso del sistema gli utenti avanzano richieste correttive/evolutive del software. Inoltre soprattutto un software gestionale deve evolversi adeguandosi alla realtà.

Sito Web

Oramai un sito web costituisce il biglietto da visita ideale, ed è per questo che uno dei primi sviluppi futuri sarà proprio quello di creare un sito web per far conoscere al mercato Kugo Invoice e fornire alla comunità un supporto ufficiale.

Gestione magazzino

Come già accennato in precedenza, una delle funzionalità che vorremmo implementare è la gestione del magazzino che permetterà di semplificare il lavoro sia dal punto di vista contabile, in quanto verranno registrate tutte le fatture di acquisto e dunque sarà molto più comodo calcolare l'IVA da pagare nel F24, sia dal punto di vista operativo in quanto si saprà in ogni momento l'effettiva quantità di merce presente in magazzino

Fatturazione elettronica

La fatturazione elettronica è un sistema digitale di emissione, trasmissione e conservazione delle fatture che permette di abbandonare il supporto cartaceo e tutti i relativi costi di stampa spedizione e conservazione. Il nuovo formato in cui le fatture elettroniche devono essere prodotte, trasmesse, archiviate e conservate è l'XML (eXtensible Markup Language), un metalinguaggio di markup che consente di definire gli elementi contenuti in un documento, verificando così le informazioni ai fini dei controlli previsti per legge. Attualmente la fatturazione elettronica è in vigore per le fatture emesse a favore della Pubblica Amministrazione. La fatturazione elettronica è dunque un sistema

che coinvolge diversi attori: il fornitore o il suo intermediario, il Sistema di Interscambio nazionale (SdI) e la Pubblica Amministrazione destinataria della fattura. Per prima cosa la Pubblica Amministrazione comunica al proprio fornitore un codice alfanumerico univoco. Questo codice deve essere riportato nella fattura elettronica insieme ai dati fiscali. Una volta compilata, la fattura deve essere firmata digitalmente dal soggetto emittente. Firmato digitalmente significa che il soggetto dovrà firmare il documento utilizzando un dispositivo di firma sicuro (smart card o USB) rilasciato da appositi enti certificatori, i quali accertano l'identità del richiedente in questo modo la PA è in grado di verificare l'identità del soggetto che ha emesso una fattura. Una volta firmata, la fattura transita dal Sistema di Interscambio, che per legge è il punto di passaggio obbligato per tutte le fatture emesse verso la PA. Il Sistema di Interscambio verifica che il formato del documento ricevuto sia corretto e che i dati inseriti siano completi e provvede ad inviarlo all'ufficio della PA che quindi potrà procedere al pagamento del proprio fornitore.

Kugo Business Intelligence

Introduzione

Kugo Business Intelligence (KBI) rappresenta l'ultima parte del sistema "Kugo Company Manager", ed è una web application per analisi di dati aziendali che al momento lavora in locale; permette ai manager delle piccole e medie imprese di consultare e studiare grafici e indici finanziari per costruire un'idea della salute economica e finanziaria aziendale. I punti di forza di Kugo Business Intelligence sono la facilità con cui si può cambiare la visione di dati, la possibilità di esportare i singoli grafici ed indici ed infine le brevi spiegazioni sul significato delle informazioni presentate all'utente, rendendola così un'applicazione accessibile anche alle figure meno esperte nel mondo dell'economia e della finanza.

Analisi

Use Case

Il caso d'uso (Use Case) è una tecnica utilizzata nei processi di ingegneria del software per effettuare una raccolta esaustiva dei requisiti al fine di produrre un software di qualità. Essa consiste nel valutare ogni requisito focalizzandosi sugli attori che interagiscono col sistema, valutandone le varie interazioni, successivamente tutti i casi d'uso raccolti vengono descritti dal Use Case Diagram.
Un caso d'uso correttamente individuato deve avere un senso compiuto per gli attori principali (quelli che avviano il caso d'uso) nel senso che deve consentirgli di raggiungere un obiettivo per lui significativo. D'altra parte, un caso d'uso deve essere elementare, cioè non scomponibile in casi d'uso più semplici che abbiano ancora senso compiuto per gli attori coinvolti.

Di seguito sono elencati i casi d'uso trovati per KBI:

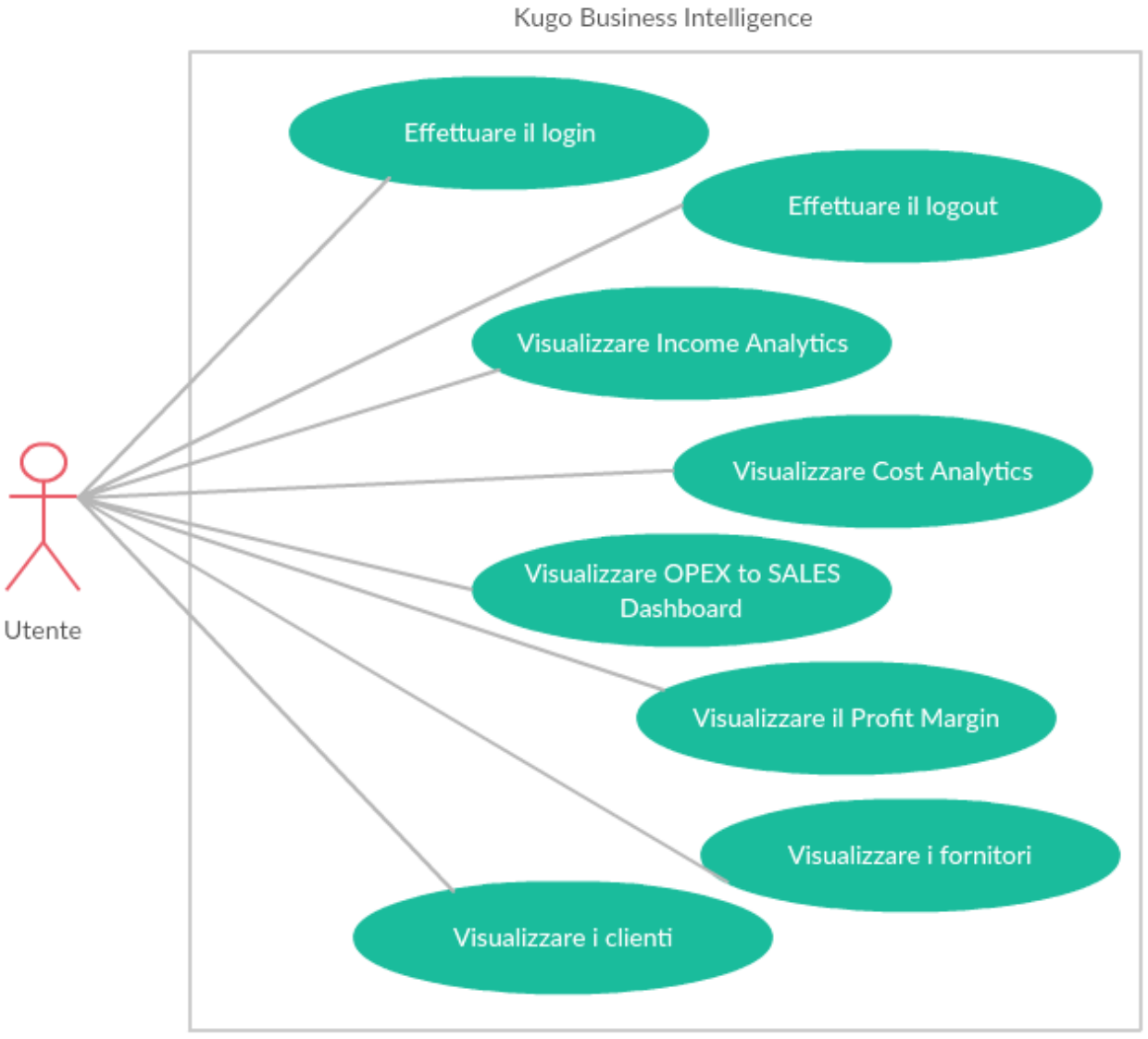
- Come Utente voglio poter effettuare il login nel sistema;
- Come Utente voglio poter effettuare il logout dal sistema;
- Come Utente voglio poter visualizzare la pagina dell'analisi delle vendite;
- Come Utente voglio poter visualizzare la pagina dell'analisi dei costi;
- Come Utente voglio poter visualizzare la pagina dell'analisi sull'indice di Opex to Sales;
- Come Utente voglio poter visualizzare la pagina dell'indice di Profit Margin;
- Come Utente voglio poter ottenere una visuale sui fornitori ed eseguire una ricerca per ragione sociale;
- Come Utente voglio poter visualizzare la pagina sui clienti ed eseguire una ricerca per ragione sociale;

Use Case Diagram

Come descritto in precedenza, in UML (unified modeling language), gli Use Case Diagram (diagrammi dei casi d'uso) sono diagrammi dedicati alla descrizione delle funzioni o servizi offerti da un sistema, così come sono percepiti e utilizzati dagli attori che interagiscono col sistema stesso. Sono impiegati soprattutto nel contesto della Use Case View (vista dei casi d'uso) di un modello, e in tal caso si possono considerare come uno strumento di rappresentazione dei requisiti funzionali di un sistema.
In un diagramma di questo tipo sono presenti tre elementi:

- SISTEMA: Il sistema nel suo complesso è rappresentato come un rettangolo vuoto;
- ATTORI: Gli attori sono rappresentati graficamente nel diagramma da un'icona che rappresenta un uomo stilizzato; essi rappresentano un ruolo coperto da un certo insieme di entità interagenti col sistema;
- USE CASE: Un caso d'uso è rappresentato graficamente come un'ellisse contenente il nome del caso d'uso ed esso rappresenta una funzione o servizio offerto dal sistema a uno o più attori.

Di seguito viene rappresentato il diagramma riguardante i casi d'uso riguardanti Kugo Business Intelligence:



Realizzazione

Il progetto “Kugo Business Intelligence” è costituito dai seguenti package:

- com.kugobi.classes: contenente le classi che descrivono l’oggetto cliente, l’oggetto fornitore e l’oggetto utente;
- com.kugobi.database: contenente la classe DatabaseManager che ritorna la connessione al database ai metodi che la richiedono;
- com.kugobi.service: contenente le classi che implementano i vari metodi che verranno richiamati, ad esempio per arrotondare un attributo “double” a due cifre decimali;
- com.kugobi.servlet: contenente le servlet richiamate per le varie funzioni richieste dalla web applications durante l’utilizzo della stessa da parte dell’utente.

Programmi e tecnologie utilizzate

Abbiamo deciso di sviluppare Kugo Business Intelligence come una web application perchè in questo modo in futuro sarà possibile rendere l’applicazione accessibile da Internet.
Per sviluppare KBI abbiamo inoltre deciso di adottare i seguenti linguaggi e tecnologie:

JSP & Servlet

Come gestione della web application dal punto di vista della parte server abbiamo deciso di utilizzare le pagine jsp con le chiamate alle servlet. Le Java Server Pages sono una tecnologia di programmazione Web in Java per lo sviluppo della logica di presentazione di applicazioni Web, fornendo contenuti dinamici in formato HTML. Si basa su un insieme di speciali tag, all'interno di una pagina HTML, con cui possono essere invocate funzioni predefinite sotto forma di codice Java e/o funzioni JavaScript sia all’interno della JSP stessa sia facendo riferimento ad una servlet. Quest’ultima è oggetto scritto in linguaggio Java (descritto nel paragrafo “Programmi e tecnologie utilizzate” del capitolo Kugo Invoice) che opera all'interno di un server web (nel nostro caso GlassFish) permettendo la creazione di web application.

NetBeans

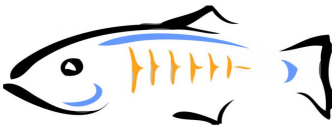
Come ambiente di sviluppo (IDE) abbiamo scelto NetBeans, dato che è stato ampiamente utilizzato nei progetti scolastici ed è provvisto di numerosi plug-in che lo rendono utile ad ogni necessità.

MySQL Workbench

Nello sviluppo di una business intelligence le query devono essere strutturate in modo corretto per evitare perdite di dati o al contrario, l’acquisizione di dati errati. Esse devono anche essere revisionate di continuo per evitare eventuali errori o bug. Per questo motivo, durante lo sviluppo di Kugo Business Intelligence, abbiamo trovato estremamente utile MySQL Workbench, infatti quest ultimo ci ha permesso di

provare diverse query su fogli di lavoro differenti in contemporanea, tenendo traccia inoltre di tutte le prove effettuate.

GlassFish



GlassFish è un application server e in particolare consente agli sviluppatori di creare applicazioni aziendali che siano portabili e scalabili, offrendo pieno supporto alle specifiche Java EE 8 con le ultime versioni delle API per le tecnologie quali ad esempio: Java Servlet 4, JavaServer Pages.

CanvasJS

CanvasJS offre delle facili ed intuibili API in javascript che permettono al programmatore di disegnare facilmente numerosi grafici. CanvasJS inoltre supporta molte tecnologie e frameworks diversi, rendendolo portabile oltre che semplice ed intuibile.

JSON

JSON è l'acronimo di JavaScript Object Notation ed è un formato standard, aperto, adatto a immagazzinare varie tipologie di informazioni e a scambiare queste informazioni tra applicazioni, sia stand alone che nel web. JSON è costituito da due sole strutture: un insieme di coppie (nome, valore) e una lista ordinata di valori.La facilità di scrittura e di analisi della notazione JSON ne ha fatto uno dei suoi punti di forza e ne ha diffuso l'utilizzo tra gli sviluppatori.

Principali librerie esterne utilizzate

Nel realizzare Kugo Business Intelligence abbiamo utilizzato varie librerie di terze parti per implementare funzionalità come la crittografia della password utilizzata nella pagina di accesso all'applicazione.

Spring Security

Come è già stato spiegato nel capitolo di Kugo Invoice per l'accesso a zone critiche della applicazione è stato aggiunto un form di login per garantire la sicurezza delle informazioni più delicate. Per effettuare correttamente il login è necessario l'inserimento del nome utente e della password; a causa di varie questioni di sicurezza, la chiave di accesso alla web app non può assolutamente essere memorizzata nel database aziendale in chiaro e per questo abbiamo utilizzato la libreria esterna org.springframework.

I metodi utilizzati

Questa classe java offre molti metodi utili, ma per il form di login abbiamo utilizzato semplicemente il metodo "checkPassword", come indicato in seguito:

```
 //controllo se il nome inserito è quello corretto
if (paramName.equals(name)) {
                //controllo se le password coincidono
                if (cripto.checkPassword(paramName, paramPassword)) {
                    //Email e password corrette, si esegue ...
                 }
```

Nella classe a cui fa riferimento l'oggetto "cripto" il codice è il seguente:

```
public boolean checkPassword(String user, String candidate_password) throws SQLException,
ClassNotFoundException {
        boolean correct = false;
        String stored_hash = returnStoredHash(user); //prendo la password dal DB
         //controllo se le psw coincidono
        if (BCrypt.checkpw(candidate_password, stored_hash)) {
            correct = true;
        } else {
            correct = false;
        }
        return correct;
    }
```

Tramite l'oggetto BCrypt viene richiamato il metodo per controllare se le due password coincidono, in particolare prende quella inserita dall'utente, vi applica la funzione di Hash e confronta la stringa ottenuta con quella selezionata dal database; se le password coincidono allora il metodo ritorna "true" altrimenti ritorna al il valore "false".

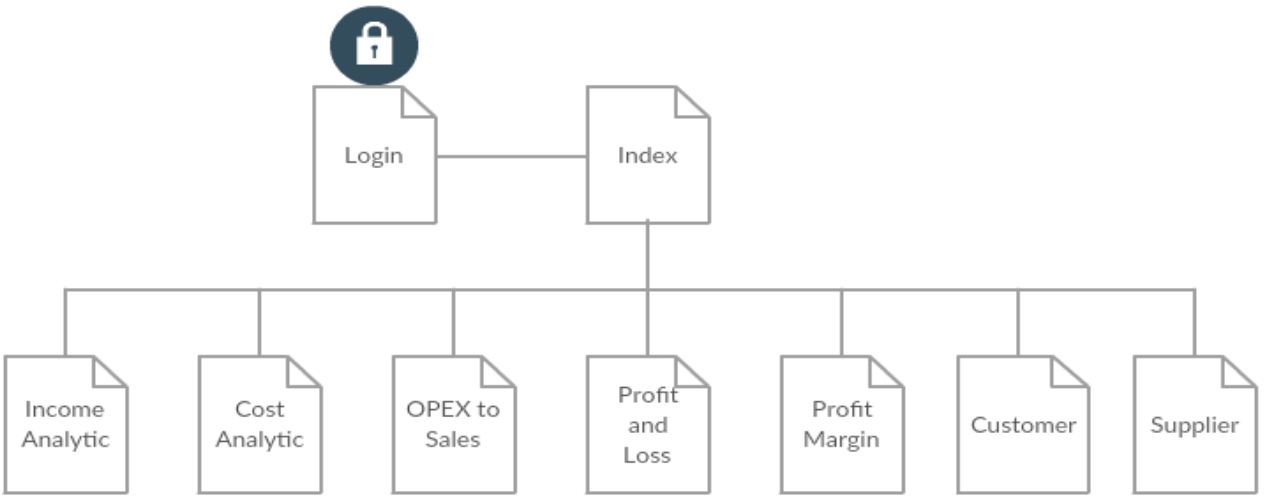
Apache Commons

Analogamente a Kugo Invoice, anche per lo sviluppo del form di login della business intelligence abbiamo bisogno della classe Apache Commons. La classe in questione è un progetto di Apache Foundation ed è costituito da una serie di librerie di utilizzo comune, pronte per essere incluse nei progetti Java. Anche per Kugo Business Intelligence è stata utilizzata la classe Base64, della libreria "org.apache.commons.codec", che è un sistema di codifica che consente la traduzione di dati binari in stringhe di testo ASCII. In questo caso Base64 viene utilizzato per convertire la password di login inserita dall'utente cifrata con AES in un coded Base64. In tale circostanza, a differenza della password della casella postale citata nel capitolo di Kugo Invoice, alle password di Kugo Business Intelligence viene applicata la funzione di Hash.

Sicurezza

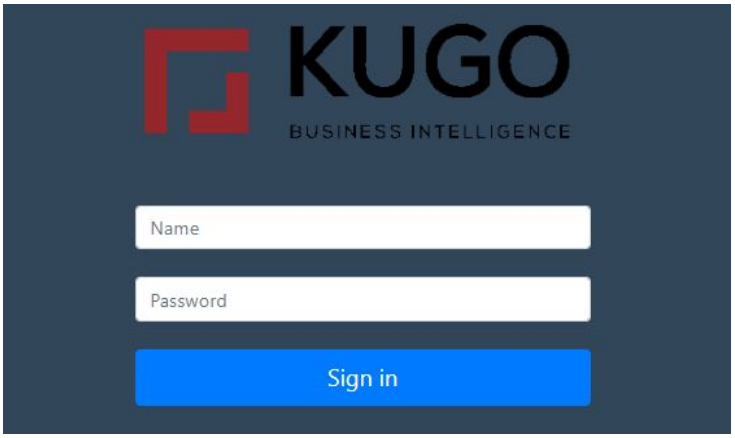
Come è già stato detto in precedenza, le risorse più importanti di ogni azienda sono i dati, poiché grazie a questi le aziende operano sui mercati e prendono decisione tattiche e strategiche. Le password sono state criptate per proteggersi da eventuali attacchi mirati a ottenere tali credenziali per poi rubare dati aziendali (spionaggio industriale).
Per l'accesso a Kugo Business Intelligence è richiesto il nome utente e la password che, una volta memorizzate all'interno del database, vengono applicate con la funzione di Hash alla stringa originale. Per cifrare la password è stata usata una libreria esterna che usa anch'essa la funzione di Hash in riferimento alla stringa desiderata. Poiché la funzione di Hash non è invertibile non ci si deve preoccupare di eventuali attacchi mirati a decodificare la stringa.

Site Map



Principali funzionalità

Login nel sistema



Una volta aperta la sezione KBI da Kugo Invoice, l'utente vedrà il portale di Login al sistema. Per accedere a Kugo Business Intelligence viene richiesto il nome utente e la password.
Una volta che viene premuto il bottone di "Sign In" viene inviata una richiesta di post alla Servlet che in questo caso è denominata "LoginServlet.java".
All'interno di quest'ultima risiede il codice java che permette l'accesso al sistema. La servlet, tramite il metodo "doPost", prende i dati inseriti dall'utente nel form di login ed effettua una query al database richiedendo la password assegnata al nome utente specificato:

```
PreparedStatement ps = con.prepareStatement("SELECT * FROM credenziali WHERE " + "user = ?");
ps.setString(1, paramName);//passo il nome utente inserito nel form
ResultSet resultSet = ps.executeQuery();
```

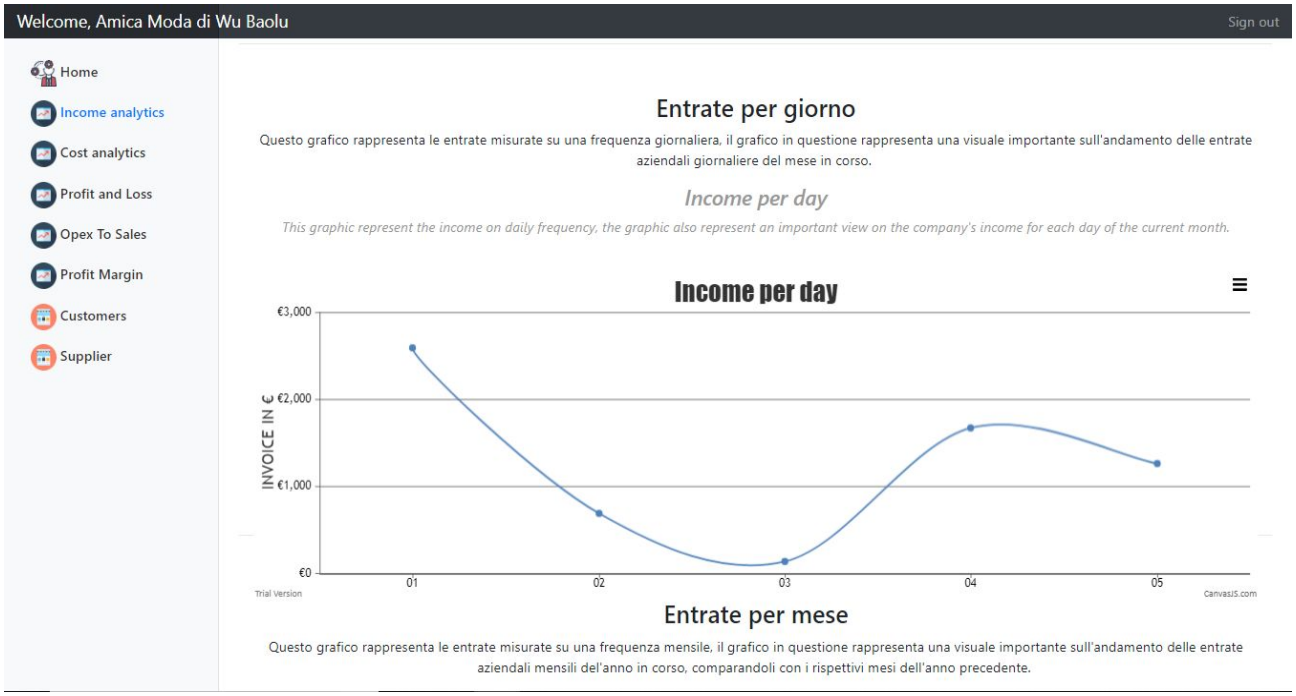
Una volta eseguita la query si controllano i risultati, se non ritorna alcun valore il nome utente è sbagliato; se invece la query ritorna dei valori significa che il nome utente è corretto, si può quindi procedere al controllo sulla password che, come descritto in precedenza, viene eseguito tramite il metodo contenuto nella libreria esterna "CriptoService". Se il controllo va a buon fine si crea una sessione a cui viene assegnata i vari attributi dell'utente e successivamente l'operatore verrà reindirizzato alla home page.

```
User user = new User(email, pass, p_iva, cod_fisc, businessName);
session.setAttribute("email", user.getEmail());
```

```
session.setAttribute("password", user.getPassword());
session.setAttribute("p_iva", user.getP_iva());
session.setAttribute("cod_fisc", user.getCod_fisc());
session.setAttribute("businessName", user.getBusinessName());
session.setMaxInactiveInterval(120 * 60);
resp.sendRedirect("Index.jsp");
```

Se invece il controllo non va a buon fine, l'operatore rimane sulla pagina di Login.

Income Analytics



Una delle visuali sugli "statement" finanziari aziendali è l'income analytics, questa visuale mostra all'utente le entrate per giorno del mese corrente, per mese confrontandoli con i corrispondenti mesi dell'anno precedente, e per anno.
Per creare i grafici abbiamo utilizzato le API in javascript che offre CanvasJS poiché risultano facili e veloci da implementare, mentre la parte complessa è la produzione dei dati da poi stampare tramite i grafici. Per prima cosa è necessario eseguire una query che prenda i dati corretti relativi al grafico in questione, per esempio se si sta per produrre il grafico su frequenza giornaliera si deve limitare i dati al mese corrente raggruppandoli per giorno;
la query sarà la seguente:

```
SELECT
            SUM(imponibile) as totImp,
            SUBSTRING( data , 1 ,2) as dataImp
 FROM
            fatture_vendita
 WHERE
            SUBSTRING( data , 7 ,4)>? AND SUBSTRING( data , 4 ,2)>? AND
            SUBSTRING( data , 4 ,2)<=? GROUP BY dataImp
```

Da ciò che ritorna la query si calcolano i dati da poi inserire nel grafico, perciò valori delle ordinate e delle ascisse, in questo caso si ottiene una stringa in cui è contenuto il giorno del mese corrente e si ottiene inoltre un valore double che indica le entrate per quel giorno.

Per ogni punto del grafico trovato si crea una HashMap in cui la coppia di valori sono le x e le y, ognuna di queste mappe verrà aggiunta ad una lista dalla quale si ottiene il JSON da passare alla API di CanvasJS che crea il grafico. Questo procedimento viene fatto con opportuni controlli in base al grafico che si vuole calcolare, e quindi in questo caso in base alla frequenza con la quale si vuole creare il grafico.
Per calcolare il JSON destinato al grafico delle entrate con frequenza giornaliera si esegue il seguente codice:

```
while (resultSett.next()) {
            varImponibile = resultSett.getString("totImp");
            b = Double.valueOf(varImponibile);
            varData = resultSett.getString("dataImp");
            mapMonth = new HashMap<Object, Object>();
            mapMonth.put("y", b);
            mapMonth.put("label", varData);
            listMonth.add(mapMonth);
            dataPointsMonth = gsonMonth.toJson(listMonth);
            if (Integer.valueOf(varData) > maxint)
            {
                max = varData;
            }
        }
```

Poi si specifica nel codice HTML dove inserire il grafico e lo stile che deve avere.

```
<div id="FatturatoPerGiorno" style="height: 370px; width: 100%; float: left;"></div>
<script src="${pageContext.request.contextPath}/scripts/kbijs.min.js"></script>
```

Infine si inserisce in fondo alla pagina HTML il tag <script> con tutte le specifiche del grafico e passando inoltre il JSON precedentemente creato.

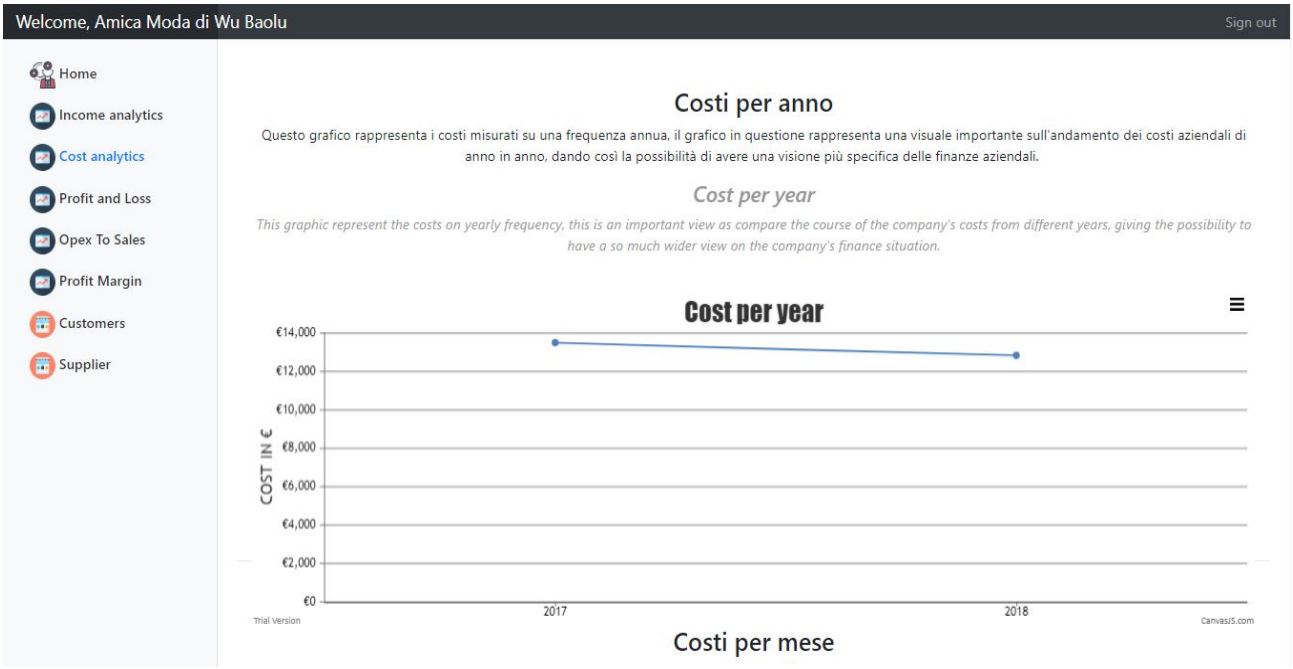
```
<script type="text/javascript">
        window.onload = function () {
        <% if (dataPointsBar != null) { %>
            var chart = new CanvasJS.Chart("FatturatoPerGiorno", {
                animationEnabled: true,
                zoomEnabled: true,
                exportEnabled: true,
                title: {
                    text: "Income per day"
                },

                axisY: {
                    prefix: "€",
                    title: "INVOICE IN €"
                },

                data: [{
                        type: "spline", //change type to bar, line, area, pie, etc
                        dataPoints: <%out.print(dataPointsBar);%>
                      }]
            });
            chart.render();
        <% } else {
                out.print("no data was found!");
            }
        %>
    }
```

```
</script>
```

Cost Analytics

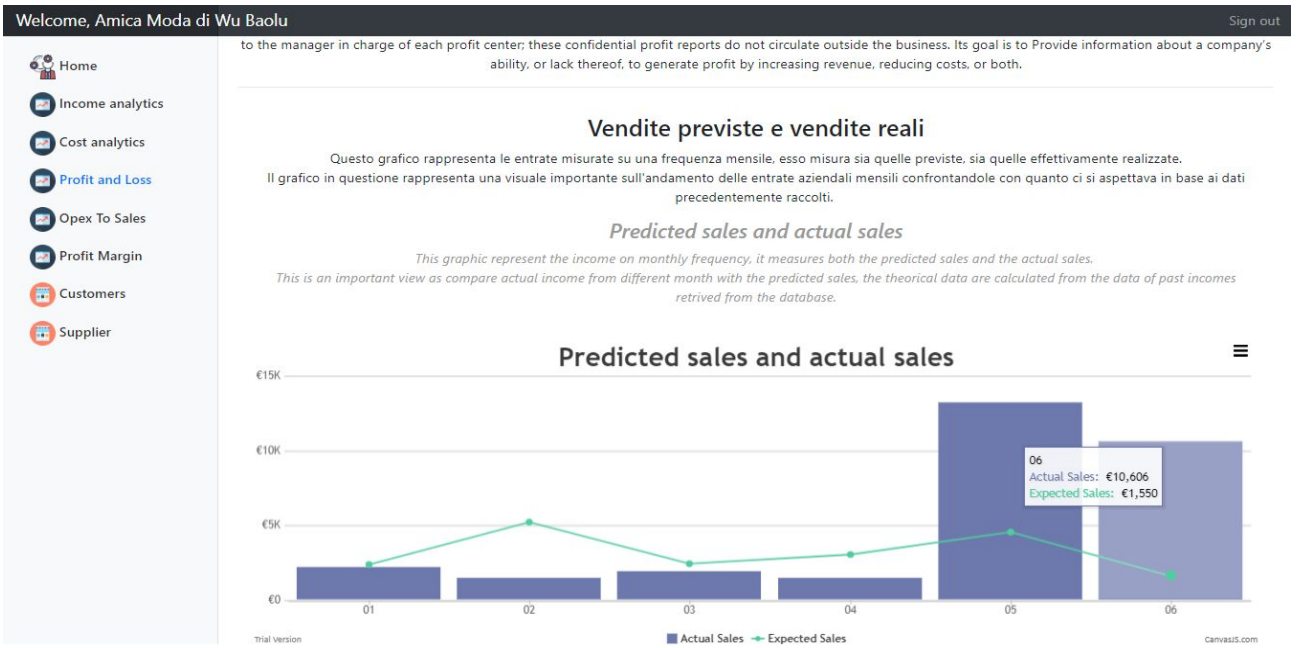


Un'altra visuale sugli "statement" finanziari aziendali è il cost analytics, questa visuale permette all'operatore di avere un'idea generale dei costi che l'azienda deve sostenere, in particolare crea grafici simili a quelli generati dalla income analytics, senza però la visione con frequenza giornaliera poiché non ritenuta abbastanza rilevante.
Il procedimento per produrre i JSON è simile a quello descritto nel paragrafo precedente, con qualche differenza nei campi selezionati dalla query e nel calcolo dei valori da inserire nel grafico in quanto i costi non derivano dalla semplice somma di un campo, ma dal prodotto tra la quantità di prodotti acquistati per il prezzo al quale questi vengono acquisiti.
Per esempio la query per calcolare i costi annuali è la seguente:

```
SELECT
                SUBSTRING( fatture_vendita.data , 7 ,4) as dataf,
                righe_fattura_vendita.codice as codiceProd,
                SUM(prodotti.prezzo_acquisto*righe_fattura_vendita.qta) as spese
FROM
                righe_fattura_vendita,
                prodotti,
                fatture_vendita
WHERE
                righe_fattura_vendita.codice_ft=fatture_vendita.codice AND
                righe_fattura_vendita.codice=prodotti.codice
GROUP BY
                fatture_vendita.data ORDER BY dataf
```

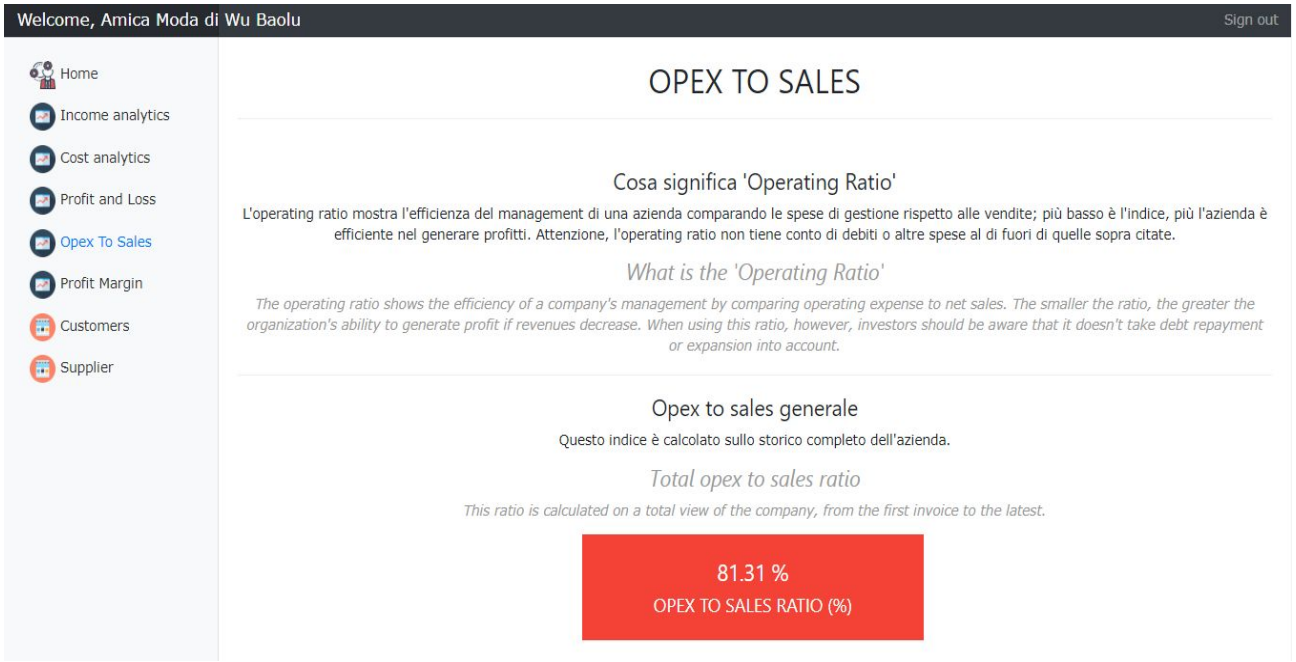
I passaggi successivi per creare il JSON, per definire il grafico da creare nel codice HTML e infine il tag <script> per costruire il grafico sono molto simili a quelli descritti in precedenza, ma con qualche significativa modifica in modo da ottenere il grafico dei costi sia a frequenza annua sia a frequenza mensile.

Profit and Loss



In questa sezione di Kugo Business Intelligence vengono rappresentate le vendite previste e le vendite effettivamente avvenute. Le vendite effettive vengono normalmente calcolate richiedendo tramite una query esatta i campi corretti al database; le vendite previste invece sono state calcolate inserendo un aumento costante del 10% sulle vendite effettuate nel mese corrispondente a quelli quelli calcolati, ma dell'anno precedente.
A livello informatico i procedimenti sono simili a quelli descritti in precedenza, ovviamente sono stati effettuati opportuni cambiamenti alle query di selezione, al codice volto alla produzione dei JSON e anche agli script che permettono di indicare le opzioni del grafico.

OPEX to Sales

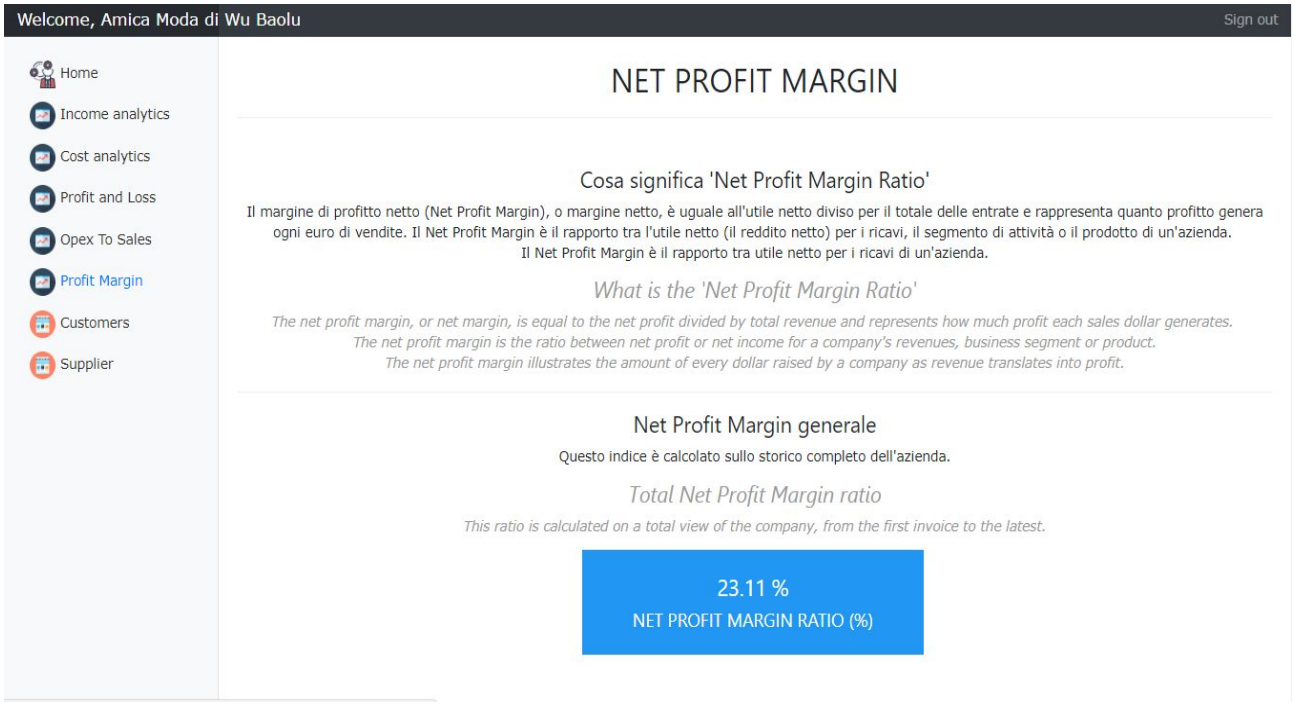


L’OPEX to Sales ratio è un’importante indicatore finanziario, abbiamo deciso di inserirlo in KBI in quanto è un indicatore base e molto utilizzato.
Il rapporto Opex-to-Sales aumenta tanto quanto il costo gestionale dell’azienda è più vicino rispetto a quanto sono i guadagni derivanti dalle vendite.
La formula per calcolare questo indicatore è la seguente:

$$OPEX = \frac{Spese\ generali}{Entrate}$$

In questa sezione inoltre sono presenti due grafici a punti(scatter/point chart), il primo rappresenta OPEX to sales ratio medio per ogni mese di vendita dell’anno corrente, il secondo invece rappresenta sempre lo stesso indice finanziario ma calcolato per ogni prodotto. Per produrre questi grafici sono state applicate le stesse metodologie applicate in precedenza nelle altre sezioni.

Net Profit Margin



L’indice di “Net Profit Margin” è l’ultima analisi svolta da questa versione di Kugo Business Intelligence, anche questo è un importante indicatore sulla salute finanziaria aziendale.
Il “Net Profit Margin” è uguale all'utile netto diviso per il totale delle entrate, e rappresenta quanto profitto genera ogni euro di vendite.
La formula per calcolare questo indice è la seguente:

$$NPM = \frac{Utile\ Netto}{Entrate\ Totali}$$

Come per l’OPEX to Sales ratio, anche per il Net Profit Margin l’indice è calcolato tramite un’interrogazione effettuata al database aziendale; a differenza delle precedenti, questa sezione di KBI non ha alcun grafico, ma presenta solo l’indice appena spiegato.

Altre funzionalità

In questa versione di Kugo Business Intelligence abbiamo deciso di inserire una visuale anche sui fornitori e sui clienti, così che l'operatore possa avere sempre ad un click le due liste quando sono necessarie. Sia la lista dei clienti sia quella dei fornitori sono state rappresentate come tabelle; per creare queste ultime, la pagina JSP effettua delle chiamate alle servlet apposite: "SupplierServlet" nel caso dei fornitori, e "ClientServlet" nel caso dei clienti.
Per entrambe le tabelle è stata aggiunta la possibilità di potere filtrare le tabelle per il campo della ragione sociale della tabella in questione. Per effettuare tale ricerca abbiamo creato una semplice funzione in JavaScript.

```
<script>
    function SearchBusinessNameFunction() {
        var input, filter, table, tr, td, i;
        input = document.getElementById("myInput");
        filter = input.value.toUpperCase();
        table = document.getElementById("myTable");
        tr = table.getElementsByTagName("tr");
        for (i = 0; i < tr.length; i++) {
            td = tr[i].getElementsByTagName("td")[0];
            if (td) {
                if (td.innerHTML.toUpperCase().indexOf(filter) > -1) {
                    tr[i].style.display = "";
                } else {
                    tr[i].style.display = "none";
                }
            }
        }
    }
</script>
```

Sviluppi futuri

Come si sa un programma informatico non si può mai dire "completo", infatti si dice che il processo di progettazione di un sistema informatico è ciclico perché con l'uso del sistema gli utenti avanzano richieste correttive/evolutive del software. Una web application come la business intelligence per di più è in continua evoluzione poiché le analisi da poter eseguire e la data warehouse contenente tutti i dati sono sempre in aggiornamento ed in espansione.

Accessibilità di KBI dal web

Il prossimo passo da eseguire per Kugo Business Intelligence è quello di renderlo accessibile dal web, diventando così di dominio pubblico. Una volta completato questo processo nasceranno diverse problematiche, la prima sarà quella di progettare un sistema distribuito che permetta di gestire tutte le richieste effettuate dagli utenti tramite internet, aggiungendo quindi costi piuttosto elevati sia di acquisto che poi di gestione e manutenzione dei server e delle attrezzature correlate.
Un'altra questione importante è la sicurezza in rete: Kugo Business Intelligence gestisce dati e informazioni aziendali che sono sensibili e quindi non comunicabili all'esterno dell'azienda, perciò è inammissibile che i pacchetti che il server e il client si scambiano possano essere intercettati. Per ovviare a questo problema ci sono varie soluzioni, una di queste è utilizzare il protocollo SSL per crittografare la connessione client-server, che utilizzeranno quindi il protocollo HTTPS invece che

HTTP. Nella connessione tramite il protocollo HTTPS è necessario che il server possegga un certificato digitale per certificare la sua identità durante la connessione.

Il protocollo SSL

Il protocollo SSL garantisce la sicurezza del collegamento mediante tre funzionalità fondamentali:

- **privatezza del collegamento**: la riservatezza del collegamento viene garantita mediante algoritmi di crittografia a chiave simmetrica;
- **autenticazione**: l'autenticazione dell'identità viene effettuata con la crittografia a chiave pubblica: in questo modo si garantisce ai client di comunicare con il server corretto, introducendo a tale scopo anche meccanismi di certificazione sia del server che del client;
- **affidabilità**: il livello di trasporto include un controllo sull'integrità del messaggio con un sistema detto MAC (Message Authentication Code) che utilizza funzioni hash sicure come SHA e MD5: avviene la verifica di integrità sui dati spediti in modo da avere la certezza che non siano stati alterati durante la trasmissione.

Certificati digitali

Il Certificato Digitale è un documento informatico firmato digitalmente dal certificatore. I dati contenuti nel certificato sono quindi seguenti:

- **dati del proprietario**, tra cui il nome, cognome e data di nascita del titolare e la chiave pubblica;
- **dati del certificato**, tra cui la data di scadenza e il numero di serie del certificato;
- **dati della Certification Authority**, ovvero la ragione sociale del certificatore, il codice identificativo del titolare presso il certificatore e la firma digitale.

Le pratiche relative alla identificazione dell'utente prima della emissione del certificato vengono fatte dalla Registration Authority che, in base alla tipologia del soggetto che richiede il certificato, svolge le necessarie indagini e attiva le relative procedure per l'identificazione certa del richiedente: la Certification Authority si occupa invece più specificatamente del "ciclo di vita del certificato", gestendo la sua pubblicazione online e relativa manutenzione.
Registration Authority e Certification Authority, per la delicatezza del ruolo che svolgono, sono enti pubblici o privati accreditati selezionati che devono aver richiesto e ottenuto il riconoscimento del possesso dei requisiti più elevati in termini di qualità e di sicurezza.
Un certificato digitale può avere diversi formati, tra i quali i più diffusi sono: chiavi PGP/GPG e certificati X.509; i primi non sono emanati da enti pubblici e quindi possono essere generati da chiunque; i secondi invece sono emanati solo da enti certificatori, rendendo quindi i certificati molti più accreditati.

Miglioramento continuo

Come accennato in precedenza una business intelligence è sempre in evoluzione e anche Kugo BI non dovrà essere da meno, per questo è necessario rilasciare periodicamente degli aggiornamenti software volti non solo ad aggiungere nuove funzionalità al software, come nuove analisi avanzate, ma anche per risoluzione di eventuali bug e errori.
Possiamo quindi riassumere questo concetto come un ciclo continuo in cui ci sono 4 fasi: plan, do, check e act. Nella prima fase vengono stabiliti gli obiettivi e anche definiti i processi necessari per produrre risultati in accordo con i requisiti dei clienti e con le politiche dell'organizzazione. La seconda fase consiste nell'implementazione dei processi pianificati. Nella terza fase si prevede il monitoraggio e la misurazione delle azioni intraprese per confrontare i risultati raggiunti con gli

obiettivi attesi. La quarta fase consiste nell'avvio di nuove azioni per correggere eventuali processi non adeguati e per ottenere un miglioramento continuo delle prestazioni tramite la preparazione di un nuovo ciclo P-D-C-A.

Conclusioni Finali

Con la realizzazione di questo progetto abbiamo capito come gli studi compiuti in questi ultimi tre anni possano applicarsi al mondo del lavoro, e in particolare che l'informatica non è un mondo isolato, ma bensì una realtà su cui il mondo del futuro sta costruendo le proprie fondamenta. Inoltre siamo riusciti a riprendere e ad approfondire la conoscenza del linguaggio Java riuscendo a sfruttare a pieno le funzionalità che offre e scoprendo librerie di cui prima ignoravamo l'esistenza. Per questo motivo siamo molto felici del percorso di studi fatto fino ad ora e quindi vorremmo ringraziare tutti i professori che ci hanno accompagnato durante il triennio, in particolar modo il prof. Andrea Pollini e il prof. Angelo Lo Magno, che ci hanno dato numerosi consigli e aiuti durante lo svolgimento del progetto.

Bibliografia

- L. Lo Russo, E. Bianchi, “Sistemi e Reti 3”, Hoepli
- F. Formichi, G. Meini, “Corso di informatica 3”, Zanichelli
- M.Conte, R.Nikolassy, P.Camagni, “Gestione del progetto e organizzazione d’impresa”, Hoepli

Sitografia

- Vincoli di integrità referenziale di propagazione - https://technet.microsoft.com/it-it/library/ms186973(v=sql.105).aspx
- BCrypt - https://it.wikipedia.org/wiki/Bcrypt
- Hashing Password in Java with BCrypt - https://www.stubbornjava.com/posts/hashing-passwords-in-java-with-bcrypt
- Hashing Password in Java with BCrypt - https://dzone.com/articles/hashing-passwords-in-java-with-bcrypt
- Framework - https://www.androidworld.it/2016/01/01/framework-373794/
- Spring - https://spring.io/projects/spring-security
- Spring Security - https://docs.spring.io/spring-security/site/docs/5.0.0.M4/api/org/springframework/security/crypto/bcrypt/BCrypt.html
- Use Case Diagram - https://it.wikipedia.org/wiki/Use_Case_Diagram
- Jasper Reports - https://it.wikipedia.org/wiki/JasperReports
- Apache Commons - http://www.html.it/articoli/apache-commons-beanutils-1/
- Modalità di funzionamento dei cifrari a blocchi - https://it.wikipedia.org/wiki/Modalit%C3%A0_di_funzionamento_dei_cifrari_a_blocchi
- Java AES - https://www.javacodegeeks.com/2018/03/aes-encryption-and-decryption-in-javacbc-mode.html
- Fatturazione Elettronica - https://fatturaelettronica.infocamere.it/fpmi/comeFunziona
- Java - https://it.wikipedia.org/wiki/Java_(linguaggio_di_programmazione)
- Java Mail - https://docs.oracle.com/javaee/7/api/javax/mail/package-summary.html
- Java Mail - https://stackoverflow.com/questions/16117365/sending-mail-attachment-using-java?utm_medium=organic&utm_source=google_rich_qa&utm_campaign=google_rich_qa
- Base64 - https://commons.apache.org/proper/commons-codec/apidocs/org/apache/commons/codec/binary/Base64.html
- Tabella Arcobaleno - https://it.wikipedia.org/wiki/Tabella_arcobaleno
- SQL - https://www.w3schools.com/
- Javascript - https://www.w3schools.com/
- Indici finanziari - https://www.investopedia.com/
- Esempi Dashboard finanziarie - https://www.sisense.com/
- Hashmap - https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html
- CanvasJS - https://canvasjs.com/
- Casi d’uso - https://creately.com/