Object Design Document - TecStore

Natale Guadagno, Paolo Patrone

February 18, 2022

Contenuti

Partecipanti

Nome	Matricola
Guadagno Natale	0512106546
Patrone Paolo	0512106153

1 Trade-off

1.1 Leggibilità del codice e sicurezza vs tempo di sviluppo

Se è vero che nel System Design Document sono state identificate le necessità di avere codice facilmente leggibile ed espandibile, i ristretti tempi di sviluppo impongono una minor attenzione a tale requisito, pur mantenendo un livello minimo. Ciò non deve avvenire a discapito della sicurezza del sistema, che deve rispettare i parametri identificati in precedenza.

1.2 Prestazioni vs Requisiti hardware

Nonostante i potenziali *drawback* del salvataggio delle immagini all'interno del database MySQL, come l'aumento dei tempi di backup e l'aumento dell'uso di memoria RAM, si è scelto di semplificare l'implementazione e utilizzare tale tecnica.

1.3 Prestazioni vs Tempi di sviluppo

Sebbene si sia identificato in precedenza il requisito di poter gestire dinamicamente il carico attraverso più nodi provvisti di *load balancer*, a causa dei tempi ristretti di sviluppo di è deciso di non dare priorità a tale requisito per evitare ulteriori ostacoli e rallentamenti durante lo sviluppo. Rimane un elemento importante che andrebbe aggiunto in una successiva revisione del sistema.

2 Convenzioni per la stesura del codice

2.1 Naming convention

È richiesto utilizzare nomi di variabili, classi e metodi che siano:

- Descrittivi del loro utilizzo, evitando quanto più possibile ambiguità e nomi poco descrittivi (e.g. "a", "abc", e così via).
- Pronunciabili, evitando il più possibile abbreviazioni ed evitando di utilizzare lingue diverse (e.s. unione italiano ed inglese nello stesso nome)
- Di lunghezza medio-corta (preferibilmente meno di 12 caratteri), per facilitare la lettura e la scrittura.
- Usando la pratica del "camel case" per tutti i nomi composti da più di una parola, con la prima lettera della prima parola minuscola per la prima parola e maiuscola per tutte le parole successive per i nomi di variabili e metodi, e tutte le parole con la prima lettera maiuscola per tutti i nomi delle classi.

2.2 Commenti

Eventuali commenti devono essere raggruppati all'inizio della funzione per facilitare la lettura della funzione stessa.

2.3 Classi

I nomi delle classi devono essere immediatamente riconoscibili e riconducibili al loro utilizzo e ogni classe deve essere racchiusa in un file separato.

La stesura delle classi deve seguire il seguente ordine:

1. Dichiarazione del package
2. Import di librerie e classi necessarie
3. Dichiarazione di classe pubblica
4. Dichiarazione di costanti
5. Dichiarazione di variabili di classe
6. Dichiarazione di variabili d'istanza
7. Dichiarazione del costruttore
8. Dichiarazione di metodi pubblici
9. Dichiarazione di Setter e Getter
10. Dichiarazione di metodi privati

3 Packages

3.1 Backend

L'implementazione del sistema è divisa in più *package* per semplificare la distinzione tra le classi:

- bean
- control
- model
- util

3.1.1 bean

Classe	Descrizione
ArticoloBean	Rappresentazione di un articolo, con informazioni sul nome, prezzo, quantità disponibile e altre informazioni rilevanti
FotoBean	Rappresentazione di una foto, contenente oltre alla foto stessa il riferimento all'articolo a cui fa riferimento
MessaggioBean	Rappresentazione di una risposta ad un ticket, contenente tutte le informazioni rilevanti, come l'autore e il ticket a cui fa riferimento
OrdineBean	Rappresentazione di un ordine, contenente le informazioni rilevanti come il cliente, la quantità, il codice del *tracking*
TicketBean	Rappresentazione di un ticket, contenente le informazioni rilevanti
UtenteBean	Rappresentazione di un utente, contenente varie informazioni necessarie

3.1.2 control

Carrello	
Classe	Descrizione
AggiornamentoQuantitaCarrelloServlet	Servlet adibita all'aggiornamento della quantità di un articolo presente nel carrello
AggiuntaAlCarrelloServlet	Servlet adibita all'aggiunta di un articolo al carrello
GetCarrelloServlet	Servlet adibita al recupero dell'elenco degli articoli nel carrello
RimozioneDalCarrelloServlet	Servlet adibita alla rimozione di un articolo dal carrello

Ordine	
Classe	Descrizione
AnnullamentoOrdineServlet	Servlet adibita all'annullamento di un ordine
ConfermaOrdineServlet	Servlet adibita alla conferma della spedizione di un ordine
CreazioneOrdineServlet	Servlet adibita alla conversione degli articoli presenti nel carrello in ordini
DettagliOrdineServlet	Servlet adibita al recupero dei dettagli di un ordine
ElencoOrdiniMagazziniereServlet	Servlet adibita al recupero dell'elenco degli ordini non ancora spediti
ElencoOrdiniMagazziniereServlet	Servlet adibita al recupero dell'elenco degli ordini per cui è stato richiesto un rimborso
RicercaOrdiniClienteServlet	Servlet adibita al recupero dello storico ordini di un cliente
RimborsoClienteServlet	Servlet adibita alla gestione di una richiesta di rimborso effettuata da un cliente
RimborsoMgazziniereServlet	Servlet adibita alla conferma o al rifiuto di un rimborso da parte di un magazziniere

Ticket	
Classe	Descrizione
ChiusuraTicketServlet	Servlet adibita alla chiusura di un ticket
CreazioneTicketServlet	Servlet adibita alla creazione di un ticket
DettagliTicketServlet	Servlet adibita al recupero dei dettagli di un ticket
ElencoTicketCentralinistaServlet	Servlet adibita al recupero dell'elenco dei ticket in attesa di risposta
ElencoTicketClienteServlet	Servlet adibita al recupero dell'elenco dei ticket per un cliente
RispostaTicketServlet	Servlet adibita alla gestione dei nuovi messaggi relativi ad un ticket

Utente	
Classe	Descrizione
AutenticazioneServlet	Servlet adibita all'autenticazione di un utente
Deautenticazione	Servlet adibita alla deautenticazione di un utente
DettagliUtenteServlet	Servlet adibita al recupero dei dettagli di un utente
ModificaUtenteServlet	Servlet adibita alla modifica dei dettagli di un utente
RegistrazioneUtenteServlet	Servlet adibita alla creazione di un nuovo utente
RicercaPersonaleServlet	Servlet adibita alla ricerca tra i dipendenti della piattaforma
RimozioneUtenteServlet	Servlet adibita alla rimozione dei dettagli di un utente

Vendita	
Classe	Descrizione
AnnullamentoVenditaServlet	Servlet adibita all'annullamento di una vendita
AutorizzazioneVenditaServlet	Servlet adibita all'autorizzazione di una vendita da parte di un centralinista
DettagliArticoloServlet	Servlet adibita al recupero dei dettagli di un articolo
DettagliVenditaServlet	Servlet adibita al recupero dei dettagli di una vendita
ElencoVenditeCentralinistaServlet	Servlet adibita al recupero dell'elenco delle vendite in attesa di autorizzazione
ImmaginiServlet	Servlet adibita al recupero e all'inserimento delle immagini per gli articoli
InserimentoNuovoArticoloServlet	Servlet adibita alla creazione di un nuovo articolo
ModificaArticoloServlet	Servlet adibita alla modifica dei dati di un articolo
RicercaArticoloServlet	Servlet adibita alla ricerca di un articolo
RicercaVenditaServlet	Servlet adibita alla ricerca di un articolo in vendita per un particolare utente
RimozioneArticoloServlet	Servlet adibita alla rimozione di un articolo

3.1.3 Model

Classe	Descrizione
GestioneAccount	Insieme di funzioni relative alla gestione degli account
GestioneAssistenza	Insieme di funzioni relative alla gestione dei ticket e dei messaggi
GestioneCarrello	Insieme di funzioni relative alla gestione del carrello
GestioneOrdine	Insieme di funzioni relative alla gestione degli ordini
GestioneOrdine	Insieme di funzioni relative alla gestione degli articoli e delle vendite

3.1.4 Metodi delle classi

Tutte le servlet descritte rispondono unicamente a richieste di tipo POST, ignorando totalmente le richieste di tipo GET, fatta eccezione per la servlet "ImmaginiServlet", che risponde con il file dell'immagine richiesta (se esiste) per le richieste GET e gestisce l'inserimento per le richieste POST.

Pertanto, l'interfaccia di tutte le classi Servlet è del tipo

```
package control.<>;

import ...;

@WebServlet("/<nomeServlet>")

public class <NomeClasse> extends HttpServlet{
        protected void doGet(HttpServletRequest request,
                                        HttpServletResponse response)
                        throws ServletException, IOException
                {
                        ...
                        ...
                        ...
                }

        protected void doPost(HttpServletRequest request,
                                        HttpServletResponse response)
                        throws ServletException, IOException
                {
                        ...
                        ...
                        ...
                }
}
```

Ogni model implementa più funzioni e, in alcuni casi, più versioni della stessa funzione, con diversi parametri di input o altre modifiche di lieve entità. Per brevità, sarà riportata solo la versione principale di ciascuna di queste funzioni.

```
package model;

import ...;

public class GestioneAccount {
        // Dato un codice fiscale,
        // restituisce true se esiste nel database
        public boolean exists(String) { }

        // Dati email e password,
        // restituisce true se esistono nel database e coincidono
        public boolean autenticazione(String, String) { }

        // Data una stringa,
        // ne restituisce la sua versione cifrata
        public String encryptString(String) { }

        // Data una stringa,
        // ne restituisce la sua versione decifrata
        public String decryptString(String) { }

        // Dato un codice fiscale,
        // ne restituisce l'UtenteBean completo
        public UtenteBean dettagliUtente(String) { }

        // Dato il codice fiscale di chi sta compiendo l'operazione
        // e il codice fiscale dell'utente da eliminare,
        // restituisce true se l'eliminazione va a buon fine
        public boolean eliminaUtente(String, String) { }

        // Data una lunghezza,
        // restituisce una stringa alfanumerica pseudocasuale
        public String generatePassword(int) { }

        // Dato un codice fiscale,
        // restituisce la tipologia utente
        public int getTipologia(String) { }

        // Dato il codice fiscale di chi sta compiendo l'operazione
        // e i nuovi dettagli,
        // restituisce true se la modifica dell'utente va a buon fine
        public boolean modificaUtente(String, UtenteBean) { }

        // Dati i dettagli dell'utnete da registrare,
        // restituisce true se la registrazione dell'utente va a buon fine
        public boolean registrazioneUtente(UtenteBean) { }
```

```
        // Dato il nome, cognome o il codice fiscale di un dipendente,
        // restituisce un elenco di dipendenti corrispondenti
        public ArrayList<UtenteBean> ricercaDipendenti(String) { }
}

package model;

import ...;

public class GestioneAssistenza {
        // Dato l'ID di un ticket e il nuovo stato,
        // restituisce true se il cambiamento di stato va a buon fine
        public boolean cambiaStato(String, String) { }

        // Dato il codice fiscale del creatore, la tipologia di ticket
        // e il primo messaggio
        // restituisce true se la creazione del ticket va a buon fine
        public boolean creazioneTicket(String, String, String) { }

        // Dato l'ID di un ticket,
        // ne restituisce i dettagli
        public TicketBean dettagliTicket(String) { }

        // Dato l'ID di un ticket,
        // ne restituisce i messaggi associati
        public ArrayList<MessaggioBean> elencoMessaggiTicket(String) { }

        // Dato il limite di risultati da ottenere,
        // restituisce l'elenco dei ticket in attesa di risposta
        public ArrayList<TicketBean> elencoTicketCentralinista(int) { }

        // Dato un codice fiscale e il limite di risultati da ottenere,
        // restituisce l'elenco dei ticket di un cliente
        public ArrayList<TicketBean> elencoTicketCliente(String, int) { }

        // Dato l'ID di un ticket,
        // il codice fiscale di chi scrive e un messaggio
        // restituisce true se l'inserimento del messaggio va a buon fine
        public boolean rispostaTicket(String, String, String) { }

}
```

```
package model;

import ...;

public class GestioneCarrello {
        // Dati il codice fiscale dell'utente,
        // l'ID dell'articolo e la nuova quantita',
        // restituisce true se l'aggiornamento del carrello va a buon fine
        public boolean aggiornamentoQuantita(String, String, int) { }

        // Dati il codice fiscale dell'utente,
        // l'ID dell'articolo e la nuova quantita',
        // restituisce true se l'aggiunta al carrello va a buon fine
        public boolean aggiuntaArticolo(String, String, int) { }

        // Dato il codice fiscale dell'utente,
        // restituisce l'elenco degli articoli nel carrello
        public static ArrayList<ArticoloBean> GetCarrello(String) { }


        // Dati il codice fiscale dell'utente e
        // l'ID dell'articolo
        // restituisce true se la rimozione va a buon fine
        public boolean rimozioneArticolo(String, String)  { }
}
```

```
package model;

import ...;

public class GestioneOrdine {
        // Dato l'ID di un ordine e il nuovo stato,
        // restituisce true se il cambiamento di stato va a buon fine
        public boolean cambiaStato(String, String) { }

        // Dato il codice fiscale di un utente,
        // restituisce true se gli ordini vengono creati correttamente
        public boolean creazioneOrdine(String) { }

        // Dato l'ID di un ordine,
        // ne restituisce i dettagli
        public OrdineBean dettagliOrdine(String) { }

        // Restituisce l'elenco degli ordini in attesa di spedizione
        public ArrayList<OrdineBean> elencoOrdiniMagazziniere() { }

        // Dato il limite di risultati da ottenere,
        // restituisce l'elenco degli ordini in attesa di rimborso
        public ArrayList<OrdineBean> elencoRimborsi(int) { }

        // Dati il codice fiscale del cliente,
        // il nome dell'articolo interessato e
        // il limite di risultati da ottenere,
        // restituisce l'elenco degli articoli ordinati corrispondenti
        public ArrayList<OrdineBean> ricercaOrdiniCliente
                                        (String, String, int) { }

        // Dati l'ID dell'ordine e il codice del tracking,
        // restituisce true se l'inserimento va a buon fine
        public boolean setCodiceTracciamento(String, String) { }
}
```

```
package model;

import ...;

public class GestioneVendita {
        // Dato l'ID di un articolo e il nuovo stato,
        // restituisce true se il cambiamento di stato va a buon fine
        public boolean cambiaStato(String, String) { }

        // Dato l'ID di un articolo,
        // ne restituisce i dettagli
        public ArticoloBean dettagliArticolo(String) { }

        // Dato il limite di risultati da ottenere,
        // restituisce l'elenco degli articoli in attesa di approvazione
        public ArrayList<ArticoloBean> elencoVenditeCentralinista(int) { }

        // Dato l'ID di una foto,
        // restituisce la foto stessa
        public byte[] getFoto(String) { }

        // Dato l'ID di un articolo e una foto,
        // restituisce true se l'inserimento va a buon fine
        public boolean inserimentoFoto(String, InputStream) { }

        // Dati i dettagli di un nuovo articolo,
        // restituisce l'ID del nuovo articolo se l'inserimento va a buon fine
        public String inserimentoNuovoArticolo(ArticoloBean) { }

        // Dati i dettagli dell'articolo,
        // restituisce true se la modifica va a buon fine
        public String modificaArticolo(ArticoloBean) { }

        // Data una parola chiave,
        // restituisce l'elenco degli articoli in vendita corrispondenti
        public ArrayList<ArticoloBean> ricercaArticolo(String) { }

        // Dato il codice fiscale di chi esegue l'operazione e
        // l'ID dell'articolo,
        // restituisce true se la rimozione va a buon fine
        public boolean rimozioneArticolo(String, String) { }

        // Dato l'ID dell'articolo,
        // restituisce true se la rimozione delle foto va a buon fine
        public boolean rimozioneFoto(String) { }
}
```

3.2 Frontend

Tutte le pagine sono racchiuse nel package di default, “webapp”.

File	Descrizione
areaVenditori	Pagina iniziale per un cliente che vuole vendere un articolo o visualizzare i dettagli di un articolo messo in vendita in precedenza
autenticazione	Modulo per l'autenticazione di un utente
carrello	Pagina per la visualizzazione del contenuto del carrello
centroassistenza	Pagina per la visualizzazione dell'elenco dei ticket aperti
creazioneTicket	Pagina per la creazione di un nuovo ticket
dettagliArticolo	Pagina per la visualizzazione dei dettagli di un articolo
dettagliOrdine	Pagina per la visualizzazione dei dettagli di un ordine
dettagliutente	Pagina per la visualizzazione dei dettagli di un utente, con possibilità di modifica
elencoOrdiniMagazziniere	Pagina per la visualizzazione dell'elenco degli ordini in attesa di spedizione
elencoRimborsiMagazziniere	Pagina per la visualizzazione dell'elenco dei rimborsi in attesa di conferma
elencoVenditeCentralinista	Pagina per la visualizzazione dell'elenco delle vendite in attesa di approvazione
errore	Pagina per la visualizzazione di un messaggio di errore
gestioneAssistenza	Pagina per la visualizzazione dell'elenco dei ticket in attesa di risposta
gestionepersonale	Pagina iniziale per un amministratore del personale, da cui ricercare un dipendente
index	Pagina iniziale per un cliente, da cui può accedere a tutte le funzioni
inserimentoCartaDiCredito	Modulo per l'inserimento dei dati della carta di credito da utilizzare per pagare gli ordini
inserimentoImmagini	Modulo per l'inserimento di foto dopo la creazione di una vendita
modificaArticolo	Modulo per la modifica di un articolo in vendita
modificaPassword	Modulo per la modifica della password di un cliente
nuovaVendita	Modulo per la creazione di una nuova vendita
paginainiziale	Pagina iniziale per i dipendenti della piattaforma
registrazione	Modulo per la registrazione di un utente
rispostaTicket	Modulo per l'inserimento di una risposta ad un ticket
risultatiRicerca	Pagina per la visualizzazione dei risultati di una ricerca
storicoOrdini	Pagina per la visualizzazione dello storico degli ordini per un cliente
successo	Pagina per la visualizzazione dei messaggi di conferma per alcune operazioni