Object Design Document

Indice

Linee guida per la documentazione dell'interfaccia....2
Packages....2
- JSP....2
- Servlet....2
- Bean....2
- Account....2
- Catalogo....2
- Ordine....2
- Ricetta....2

Glossario delle interfacce delle classi....3
- Manager....3
- Servlet....12
- Bean....22

Diagrammi UML per il Web....41

Linee guida per la documentazione dell'interfaccia

- Le classi sono denominate con sostantivi singolari
- I campi e i parametri sono denominati con sostantivi singolari
- Gli errori sono rappresentati tramite eccezioni e non valori di ritorno
- Le collezioni hanno operazioni che permettono di aggiungere, eliminare e recuperare tutti i propri elementi

Packages

Si sceglie di rappresentare ogni sottosistema tramite un package tranne UserInterface suddiviso a sua volta in due package “JSP” e “Servlet”.

JSP

Questo package contiene tutti i file JSP del sistema. Il package dipende dai package “Bean” e dal package “Servlet”.

Servlet

Questo package contiene tutte le Servlet del sistema. Il package dipende dai package relativi alle classi manager e da quello relativo ai Bean.

Bean

Questo package contiene tutte le classi Bean del sistema.

Account

Questo package contiene le classi AziendaDAO, ClienteDAO e AdminDAO del sistema. Il package dipende dal package relativo ai Bean.

Catalogo

Questo package contiene le classi ProdottoDAO e Carrello del sistema. Il package dipende dal package relativo ai Bean.

Ordine

Questo package contiene le classi FatturaDAO, ComposizioneDAO ed EffettuazioneDAO del sistema. Il package dipende dal package relativo ai Bean.

Ricetta

Questo package contiene le classi RicettaDAO, RecensioneDAO del sistema. Il package dipende dal package relativo ai Bean.

Glossario delle interfacce delle classi

Manager

UserDAO

Nome Classe: UserDAO
Descrizione Classe: Classe il cui compito è quello di poter creare e autenticare utenti.
Metodi: + login(String email, String password); + registration(ClienteBean cliente); + aggiornamento (String colonna, String valore, String chiave); + aggiornamento (String colonna, int valore, String chiave); + getcreditcards (ClienteBean cliente); + addcreditcards (String numero, String tipologia, String email); + deletecreditcards (String numero); - check (String email);

Nome Metodo: + login (String email, String password):ClienteBean
Descrizione Metodo: Inserite delle credenziali si controlla se l'utente è presente o meno nel database.

Nome Metodo: + registration(ClienteBean cliente):boolean
Descrizione Metodo: Compilati i campi di input, se i parametri immessi rispettano le condizione verrà registrato un nuovo cliente.

Nome Metodo: + aggiornamento (String colonna, String valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + aggiornamento (String colonna, int valore, String chiave): boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + getcreditcards (ClienteBean cliente): HashMap<String,String>
Descrizione Metodo: Permette di recuperare tutte le carte di credito associate a quell'utente.

Nome Metodo: + addcreditcards (String numero, String tipologia, String email):void
Descrizione Metodo: Permette di aggiungere una carta di credito associata ad un utente.

Nome Metodo: + deletecreditcards (String numero):void
Descrizione Metodo: Permette di rimuovere una carta di credito associata ad un utente.

Nome Metodo: - check (String email):boolean
Descrizione Metodo: Controlla se una data email è già presente nel database.

AziendaDAO

Nome Classe: AziendaDAO
Descrizione Classe: Classe il cui compito è quello di poter creare e autenticare aziende.
Metodi: + login(String email, String password); + registration(AziendaBean azienda); + aggiornamento (String colonna, String valore, String chiave); + aggiornamento (String colonna, int valore, String chiave); - check (String email);

Nome Metodo: + login(String email, String password):AziendaBean
Descrizione Metodo: Inserite delle credenziali si controlla se l'azienda è presente o meno nel database.

Nome Metodo: + registration(AziendaBean cliente):boolean
Descrizione Metodo: Compilati i campi di input, se i parametri immessi rispettano le condizione verrà registrato un nuovo cliente.

Nome Metodo: + aggiornamento (String colonna, String valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + aggiornamento (String colonna, int valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: - check (String email):Boolean
Descrizione Metodo: Controlla se una data email è già presente nel database.

RicettaDAO

Nome Classe: RicettaDAO
Descrizione Classe: Classe il cui compito è quello di poter recuperare ed eliminare ricette.
Metodi: + categoria(String categoria); + Onericetta (String titolo); + provenienza (String provenienza);

Nome Metodo: + categoria(String categoria):ArrayList<RicettaBean>
Descrizione Metodo: Indicata un data categoria vengono restituite tutte le ricette relative a questa.

Nome Metodo: + Onericetta (String titolo):RicettaBean
Descrizione Metodo: Restituisce la ricetta il cui titolo corrisponde a quello dato in input.

Nome Metodo: + Provenienza (String provenienza):ArrayList<RicettaBean>
Descrizione Metodo: Indicata un data provenienza vengono restituite tutte le ricette relative a questa.

ProdottoDAO

Nome Classe: ProdottoDAO
Descrizione Classe: Classe il cui compito è quello di poter creare, recuperare ed eliminare prodotti.
Metodi: + unprodotto (String nome); + categoriaprodotto (String codice, int contatore); + prodottofferta (int contatore); + searchbar (int contatore, String nome); + aggiornamento (String colonna, int valore, String chiave); + aggiornamento (String colonna, String valore, String chiave); + aggiornamento (String colonna, double valore, String chiave); + cancellazione (String nome); + aggiunta (int prodotto); + unprodottocodice (String codice);

Nome Metodo: + unprodotto (String nome):ProdottoBean
Descrizione Metodo: Dato il nome di un prodotto, verrà restituito un prodotto con quel nome.

Nome Metodo: + categoriaprodotto (String codice, int contatore):ArrayList<ProdottoBean>
Descrizione Metodo: Data una categoria di prodotti, verrà restituito un insieme di prodotti appartenenti ad essa.

Nome Metodo: + prodottofferta (int contatore):ArrayList<ProdottoBean>
Descrizione Metodo: Permette di restituire tutti i prodotti in offerta.

Nome Metodo: + searchbar (int contatore, String nome):ArrayList<ProdottoBean>
Descrizione Metodo: Dato un criterio di ricerca, verrà restituito un insieme di prodotti inerenti ad esso.

Nome Metodo: + aggiornamento (String colonna, int valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + aggiornamento (String colonna, String valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + aggiornamento (String colonna, double valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare le occorrenze nel database.

Nome Metodo: + cancellazione (String nome):void
Descrizione Metodo: Permette di eliminare una occorrenza di un prodotto dal database.

Nome Metodo: + aggiunta (int prodotto):void
Descrizione Metodo: Permette di inserire una occorrenza di un prodotto all’interno del database.

Nome Metodo: + unprodottocodice (String codice):ProdottoBean
Descrizione Metodo: Dato un codice, verrà restituito un prodotto inerente ad esso.

FatturaDAO

Nome Classe: FatturaDAO
Descrizione Classe: Classe il cui compito è quello di poter creare, recuperare ed eliminare fatture.
Metodi: + inserimento (FatturaBean fattura) + getfatture (String email) + gettuttefatture (String datamin, String datamax) + gettuttefatture (String datamin, String datamax, String email)

Nome Metodo: + inserimento (FatturaBean fattura):void
Descrizione Metodo: Permette di inserire una nuova occorrenza all'interno del database.

Nome Metodo: +getfatture (String email):ArrayList<FatturaBean>
Descrizione Metodo: Data l'email di un cliente, restituisce tutte le fatture inerenti ad esso.

Nome Metodo: + gettuttefatture (String datamin, String datamax):ArrayList<FatturaBean>
Descrizione Metodo: Data l'email di un cliente ed una data, restituisce tutte le fatture inerenti ad esso a partire da quella data.

Nome Metodo: + gettuttefatture (String datamin, String datamax, String email): ArrayList<FatturaBean>
Descrizione Metodo: Data l'email di un cliente ed un intervallo di tempo, restituisce tutte le fatture inerenti a quel cliente presenti in quell'intervallo.

RecensioneDAO

Nome Classe: RecensioneDAO
Descrizione Classe: Classe il cui compito è quello di poter creare, recuperare ed eliminare recensioni.
Metodi: + inserimento (RecensioneBean recensione) + getrecensione (String email, String titolo) + eliminarecensione (String email, String titolo)

Nome Metodo: + inserimento (RecensioneBean recensione):void
Descrizione Metodo: Permette di inserire una occorrenza all'interno del database.

Nome Metodo: + getrecensione (String email, String titolo):RecensioneBean
Descrizione Metodo: Verrà restituita la recensione i quali criteri corrispondono a quelli passati come parametro.

Nome Metodo: + eliminarecensione (String email, String titolo):boolean
Descrizione Metodo: Permette di eliminare un'occorrenza dal database.

EffettuazioneDAO

Nome Classe: EffettuazioneDAO
Descrizione Classe: E' una classe il cui compito è quello di aggiungere ricorrenze di effettuazioni.
Metodi: + aggiunta (EffettuazioneBean eff)

Nome Metodo: + aggiunta (EffettuazioneBean eff):void
Descrizione Metodo: Permette di aggiungere un'occorrenza dal database.

ComposizioneDAO

Nome Classe: ComposizioneDAO
Descrizione Classe: E' una classe il cui compito è quello di aggiungere ricorrenze di composizioni che si possono aggiungere.
Metodi: + aggiunta (ComposizioneBean comp)

Nome Metodo: + aggiunta (ComposizioneBean comp):void
Descrizione Metodo: Permette di aggiungere un'occorrenza dal database.

AdminDAO

Nome Classe: AdminDAO
Descrizione Classe: Classe il cui compito è quello di autenticare Admin, gestire le ricette, i prodotti e gli ordini.
Metodi: + login(String email,String password) + aggiornamento (String colonna, String valore, String chiave)

Nome Metodo: + login(String email,String password):AdminBean
Descrizione Metodo: Inserite delle credenziali si controlla se l'admin è presente o meno nel database.

Nome Metodo: + aggiornamento (String colonna, String valore, String chiave):boolean
Descrizione Metodo: Permette di aggiornare l'occorrenza nel database.

Servlet

ModificaDatiPersonaliServlet

Nome Classe: ModificaDatiPersonaliServlet
Descrizione Classe: Si occupa dell'aggiornamento dei dati relativi all'utente.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiornamento dei dati di un utente.
Postcondizione: context ModificaDatiPersonaliServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Cliente”).getNome() <> request.getSession().getAttribute(“Cliente”)@pre.getNome() OR request.getSession().getAttribute(“Cliente”).getCognome() <> request.getSession().getAttribute(“Cliente”)@pre.getCognome() OR request.getSession().getAttribute(“Cliente”).getEmail() <> request.getSession().getAttribute(“Cliente”)@pre.getEmail() OR request.getSession().getAttribute(“Cliente”).getPassword() <> request.getSession().getAttribute(“Cliente”)@pre.getPassword() OR request.getSession().getAttribute(“Cliente”).getCitta() <> request.getSession().getAttribute(“Cliente”)@pre.getCitta() OR request.getSession().getAttribute(“Cliente”).getVia() <> request.getSession().getAttribute(“Cliente”)@pre.getVia() OR request.getSession().getAttribute(“Cliente”).getCap() <> request.getSession().getAttribute(“Cliente”)@pre.getCap() OR request.getSession().getAttribute(“Cliente”).getNCivico() <> request.getSession().getAttribute(“Cliente”)@pre.getNCivico() OR request.getSession().getAttribute(“Cliente”).getDataDiNascita() <> request.getSession().getAttribute(“Cliente”)@pre.getDataDiNascita() OR request.getSession().getAttribute(“Cliente”).getNTelefono() <> request.getSession().getAttribute(“Cliente”)@pre.getNTelefono()

ModificaDatiAdminServlet

Nome Classe: ModificaDatiAdminServlet
Descrizione Classe: Si occupa dell'aggiornamento dei dati relativi all'admin.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiornamento dei dati di un admin.
Postcondizione: context ModificaDatiAdminServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Admin”).getNome() <> request.getSession().getAttribute(“Admin”)@pre.getNome() OR request.getSession().getAttribute(“Admin”).getCognome() <> request.getSession().getAttribute(“Admin”)@pre.getCognome() OR request.getSession().getAttribute("Admin”).getEmail() <> request.getSession().getAttribute(“Admin”)@pre.getEmail() OR request.getSession().getAttribute(“Admin”).getPassword() <> request.getSession().getAttribute(“Admin”)@pre.getPassword() OR request.getSession().getAttribute(“Admin”).getCitta() <> request.getSession().getAttribute(“Admin”)@pre.getCitta() OR request.getSession().getAttribute(“Admin”).getVia() <> request.getSession().getAttribute(“Admin”)@pre.getVia() OR request.getSession().getAttribute(“Admin”).getCap() <> request.getSession().getAttribute(“Admin”)@pre.getCap() OR request.getSession().getAttribute(“Admin”).getNCivico() <> request.getSession().getAttribute(“Admin”)@pre.getNCivico() OR request.getSession().getAttribute(“Admin”).getDataDiNascita() <> request.getSession().getAttribute(“Admin”)@pre.getDataDiNascita() OR request.getSession().getAttribute(“Admin”).getNTelefono() <> request.getSession().getAttribute(“Admin”)@pre.getNTelefono()

ModificaDatiAziendaServlet

Nome Classe: ModificaDatiAziendaServlet
Descrizione Classe: Si occupa dell'aggiornamento dei dati relativi all'azienda.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiornamento dei dati di un'azienda.
Postcondizione: context ModificaDatiAdminServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Azienda”).getNome() <> request.getSession().getAttribute(“Azienda”)@pre.getNome() OR request.getSession().getAttribute("Azienda”).getEmail() <> request.getSession().getAttribute(“Azienda”)@pre.getEmail() OR request.getSession().getAttribute(“Azienda”).getPassword() <> request.getSession().getAttribute(“Azienda”)@pre.getPassword() OR request.getSession().getAttribute(“Azienda”).getCitta() <> request.getSession().getAttribute(“Azienda”)@pre.getCitta() OR request.getSession().getAttribute(“Azienda”).getVia() <> request.getSession().getAttribute(“Azienda”)@pre.getVia() OR request.getSession().getAttribute(“Azienda”).getCap() <> request.getSession().getAttribute(“Azienda”)@pre.getCap() OR request.getSession().getAttribute(“Azienda”).getNCivico() <> request.getSession().getAttribute(“Azienda”)@pre.getNCivico() OR request.getSession().getAttribute(“Azienda”).getDataDiNascita() <> request.getSession().getAttribute(“Azienda”)@pre.getDataDiNascita() OR request.getSession().getAttribute(“Azienda”).getNTelefono() <> request.getSession().getAttribute(“Azienda”)@pre.getNTelefono()

SearchbarServlet

Nome Classe: SearchbarServlet
Descrizione Classe: Si occupa del prelievo dei prodotti il cui nome corrisponde con ciò che è stato digitato dall'utente nella barra di ricerca.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo dei prodotti il cui nome corrisponde con quanto digitato dall'utente.

RicercaFatturaServlet

Nome Classe: RicercaFatturaServlet
Descrizione Classe: Si occupa del prelievo di tutte le fatture corrispondenti ai criteri di ricerca.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo delle fatture che soddisfano i criteri di ricerca.

RegistrationServlet

Nome Classe: RegistrationServlet
Descrizione Classe: Si occupa di aggiungere un'occorrenza di utente o azienda nel database.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiunta di un'occorrenza di utente o azienda all'interno del database.

ProvenienzaServlet

Nome Classe: ProvenienzaServlet
Descrizione Classe: Si occupa del prelievo di tutte le ricette in base ad una certa provenienza.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo di tutte le ricette sulla base di una certa provenienza.

ProdottoServlet

Nome Classe: ProdottoServlet
Descrizione Classe: Si occupa del prelievo di alcuni prodotti dal catalogo.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo dei prodotti dal catalogo.

ProdottoOffertaServlet

Nome Classe: ProdottoOffertaServlet
Descrizione Classe: Si occupa del prelievo dei prodotti in offerta dal catalogo.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo dei prodotti in offerta dal catalogo.

OneProductServlet

Nome Classe: OneProductServlet
Descrizione Classe: Si occupa del prelievo di un prodotto dal catalogo in base al suo nome.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo di un prodotto in base al nome.

RicercaFatturaServlet

Nome Classe: RicercaFatturaServlet
Descrizione Classe: Si occupa del prelievo di tutte le fatture corrispondenti ai criteri di ricerca.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo delle fatture che soddisfano i criteri di ricerca.

ModificaProdottoServlet

Nome Classe: ModificaProdottoServlet
Descrizione Classe: Si occupa della modifica dei dati relativi ad un prodotto.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiornamento dei dati relativi al prodotto.

LogoutServlet

Nome Classe: LogoutServlet
Descrizione Classe: Si occupa di eliminare il bean dell'utente autenticato dalla sessione, effettuando così il logout.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al logout.

LoginServlet

Nome Classe: LoginServlet
Descrizione Classe: Si occupa di effettuare l'autenticazione di un utente in base ad una e-mail ed una password.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al login.

FinalRicettaServlet

Nome Classe: FinalRicettaServlet
Descrizione Classe: Si occupa del prelievo di una ricetta in base al titolo.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo di una ricetta in base al titolo.

DeleteItemServlet

Nome Classe: LogoutServlet
Descrizione Classe: Si occupa della cancellazione di un determinato prodotto dal carrello.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative alla cancellazione di un prodotto dal carrello.
Postcondizione: context DeleteItemServlet:: doGet(request:HttpServletRequest,response:HttpServletResponse) post: request.getSession().getAttribute("Carrello") → size()=request.getSession().getAttribute("Carrello") → @pre.size()-1.

AcquistoServlet

Nome Classe: AcquistoServlet
Descrizione Classe: Si occupa di gestire le modifiche nel database legate all'acquisto dei prodotti nel carrello.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'acquisto del contenuto nel carrello.
Postcondizione: context AcquistoServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Cliente”).getNpunti() >= request.getSession().getAttribute(“Cliente”)@pre.getNpunti() AND request.getSession().getAttribute(“Carrello”) → size()=0

CategoriaServlet

Nome Classe: CategoriaServlet
Descrizione Classe: Si occupa del prelievo di tutte le ricette in base ad una certa categoria.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo delle ricette di una certa categoria.

CategoriaProdottoServlet

Nome Classe: CategoriaProdottoServlet
Descrizione Classe: Si occupa di prelevare tutti i prodotti in base ad una certa categoria.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative al prelievo dei prodotti di una certa categoria.

CancellazioneProdottoServlet

Nome Classe: CancellazioneProdottoServlet
Descrizione Classe: Si occupa della cancellazione di un certo prodotto dal database.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative alla cancellazione di un prodotto dal database.

AggiungiProdottoServlet

Nome Classe: AggiungiProdottoServlet
Descrizione Classe: Si occupa di aggiungere un'occorrenza di prodotto al database.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiunta di un prodotto al database.

AddCreditCardServlet

Nome Classe: AddCreditCardServlet
Descrizione Classe: Si occupa di aggiungere una carta di credito relativa all'utente nel database.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiunta della carta di credito.
Postcondizione: context AddCreditCartServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Cliente”).getCreditCard() → size()=request.getSession().getAttribute(“Cliente”)@pre.getCreditCart() → size()+1

DeleteCreditCardServlet

Nome Classe: DeleteCreditCardServlet
Descrizione Classe: Si occupa di rimuovere una carta di credito associata all'utente dal database.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'eliminazione di una carta di credito.
Postcondizione: context AddCreditCartServlet:: doGet(request:HttpServletRequest, response:HttpServletResponse) post request.getSession().getAttribute(“Cliente”).getCreditCard() → size()=request.getSession().getAttribute(“Cliente”)@pre.getCreditCart() → size()-1

AddCartServlet

Nome Classe: AggiungiProdottoServlet
Descrizione Classe: Si occupa di aggiungere un prodotto al carrello.
Metodi: + doGet(HttpServletRequest request,HttpServletResponse response)

Nome Metodo: + doGet(HttpServletRequest request,HttpServletResponse response):void
Descrizione Metodo: Compie le operazioni relative all'aggiunta di un prodotto al carrello.
Postcondizione: context AddCartServlet::doGet(request:HttpServletRequest,response:HttpServletResponse) post: request.getSession().getAttribute("Carrello")->size()=request.getSession().getAttribute("Carrello")->@pre.size()+1

Bean

ProdottoBean

Nome Classe: ProdottoBean

Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi ai prodotti.

Metodi:

+getCodice()
+setCodice(String codice)
+getNome_Prodotto()
+setNome_Prodotto(String nome_prodotto)
+getDescrizione()
+setDescrizione(String descrizione)
+getConservazione()
+setConsevazione(String conservazione)
+getCategoria()
+setCategoria(String categoria)
+getIva()
+setIva(Int iva)
+getQuantitaDisponibili()
+setQuantitaDisponibili(int quantita_disponibili)
+getPrezzo_Base()
+setPrezzo_Base(double prezzo_base)
+getPrezzo_Totale()
+setPrezzo_Totale(double prezzo_totale)
+isOfferta()
+setOfferta(boolean offerta)
+getImmagine()
+setImmagine(Blob immagine)

Invariante:

context Prodotto inv:
self.allInstances()→isUnique(Codice);

context Prodotto inv:
self.codice.toInteger() >= 1;

context Prodotto inv:
self.categoria ==”Latticini” OR
self.categoria ==” Formaggi” OR
self.categoria ==” Sottovuoto” OR
self.categoria ==”Pasta” OR
self.categoria ==”Olio” OR
self.categoria ==”Vino”;

```
context Prodotto inv:
self.quantitàDisponibili >= 0;

context Prodotto inv:
self.prezzoBase >= 0;

context Prodotto inv:
self.iva >= 0   AND
self.iva <=100;

context Prodotto inv:
self.prezzoTotale = self.prezzoBase+(self.prezzoBase/100)*self.iva;
```

Nome Metodo: + getCodice():String
Descrizione Metodo: Viene visualizzato il codice relativo al prodotto.

Nome Metodo: + setCodice(String codice):void
Descrizione Metodo: Viene settato il codice relativo al prodotto.

Nome Metodo: + getNome_Prodotto():String
Descrizione Metodo: Viene visualizzato il nome relativo al prodotto.

Nome Metodo: + setNome_Prodotto(String nome_prodotto):void
Descrizione Metodo: Viene settato il nome relativo al prodotto.

Nome Metodo: + getDescrizione():String
Descrizione Metodo: Viene visualizzata descrizione relativa al prodotto.

Nome Metodo: + setDescrizione(String descrizione):void
Descrizione Metodo: Viene settata la descrizione relativa al prodotto.

Nome Metodo: + getConservazione():String
Descrizione Metodo: Viene visualizzata la data di conservazione relativa al prodotto.

Nome Metodo: + setConsevazione(String conservazione):void
Descrizione Metodo: Viene settata la data di conservazione relativa al prodotto.

Nome Metodo: + getCategoria():String
Descrizione Metodo: Viene visualizzata la categoria relativa al prodotto.

Nome Metodo: + setCategoria(String categoria):void
Descrizione Metodo: Viene settata la categoria relativa al prodotto.

Nome Metodo: + getIva():int
Descrizione Metodo: Viene visualizzata l’iva relativa al prodotto.

Nome Metodo: + setIva(int Iva):void
Descrizione Metodo: Viene settata l’iva relativa al prodotto.

Nome Metodo: + getQuantitaDisponibili():int
Descrizione Metodo: Viene visualizzata la quantità di prodotto disponibile.

Nome Metodo: + setQuantitaDisponibili(Int quantita_disponibili):void
Descrizione Metodo: Viene settata la quantità di prodotto disponibile.

Nome Metodo: + getPrezzo_Base():double
Descrizione Metodo: Viene visualizzata il prezzo base relativo al prodotto.

Nome Metodo: + setPrezzo_Base(double prezzo_base):void
Descrizione Metodo: Viene settato il prezzo base relativa al prodotto.

Nome Metodo: + getPrezzo_Totale():double
Descrizione Metodo: Viene visualizzata il prezzo totale relativo al prodotto.

Nome Metodo: + setPrezzo_Totale(double prezzo_totale):void
Descrizione Metodo: Viene settato il prezzo totale relativo al prodotto.

Nome Metodo: + isOfferta():boolean
Descrizione Metodo: Controlla se il prodotto è in offerta.

Nome Metodo: + setOfferta(Boolean offerta):void
Descrizione Metodo: Un prodotto viene messo in offerta o l’offerta su di esso cessa.

Nome Metodo: + getImmagine():Blob
Descrizione Metodo: Viene visualizzata l’immagine relativa al prodotto.

Nome Metodo: + setImmagine(Blob immagine):void
Descrizione Metodo: Viene settata l’immagine relativa al prodotto.

FatturaBean

Nome Classe: FatturaBean

Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi alle fatture.

Metodi:

+ getN_documento()
+ setN_documento(String n_documento)
+ getVia()
+ setVia(String via)
+ getDestinatario()
+ setDestinatario(String destinatario)
+ getTotale_imponibile()
+ setTotale_imponibile(double totale_imponibile)
+ getTotale_imposta()
+ setTotale_imposta(double totale_imposta)
+ getCosto_totale()
+ setCosto_totale(double costo_totale)
+ getData()
+ setData(Date data)

Invariante:

context Fattura inv:
self.allInstances()→isUnique(nDocumento);

context Fattura inv:
self.nDocumento.toInteger <> invalid;

context Fattura inv:
self.via = self.cliente.cap
(self.cliente.città.concat(self.cliente.via.concat(self.cliente.numeroCivico)));

context Fattura inv:
self.destinatario = self.cliente.nome.concat(self.cliente.cognome);

context Fattura inv:
self.totaleImponibile > 0;

context Fattura inv:
self.totaleImposta > 0;

context Fattura inv:
self.costoTotale > 0;

Nome Metodo: + getN_documento():String
Descrizione Metodo: Viene visualizzato il numero del documento relativo alla fattura.

Nome Metodo: + setN_documento(String n_documento):void
Descrizione Metodo: Viene settato il numero del documento relativo alla fattura.

Nome Metodo: + getVia():String
Descrizione Metodo: Viene visualizzata la via relativa alla fattura.

Nome Metodo: + setVia(String via):void
Descrizione Metodo: Viene settata la via relativa alla fattura.

Nome Metodo: + getDestinatario():String
Descrizione Metodo: Viene visualizzato il destinatario relativo alla fattura.

Nome Metodo: + setDestinatario(String destinatario):void
Descrizione Metodo: Viene settata il destinatario relativo alla fattura.

Nome Metodo: + getTotale_imponibile():double
Descrizione Metodo: Viene visualizzato il totale imponibile relativo alla fattura.

Nome Metodo: + setTotale_imponibile(double totale_imponibile):void
Descrizione Metodo: Viene settata il totale imponibile relativo alla fattura.

Nome Metodo: + getTotale_imposta():double
Descrizione Metodo: Viene visualizzato il totale imposto relativo alla fattura.

Nome Metodo: + setTotale_imposta(double totale_imposta):void
Descrizione Metodo: Viene settata il totale imposto relativo alla fattura.

Nome Metodo: + getCosto_totale():double
Descrizione Metodo: Viene visualizzato il costo totale relativo alla fattura.

Nome Metodo: + setCosto_totale(double costo_totale):void
Descrizione Metodo: Viene settata il costo totale relativo alla fattura.

Nome Metodo: + getData():String
Descrizione Metodo: Viene visualizzata la data relativa alla fattura.

Nome Metodo: + setData(Date data):void
Descrizione Metodo: Viene settata la data relativa alla fattura.

ComposizioneBean

Nome Classe: ComposizioneBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi alle composizioni.
Metodi: + getCodice_prodotto() + setCodice_prodotto(String codice_prodotto) + getNumero_fattura() + setNumero_fattura(String numero_fattura) + getQuantita_acquistate() +setQuantita_acquistate(int quantita_acquistate) + getIva_acquisto() + setIva_acquisto(int iva_acquisto) + getPrezzo_acquisto() + setPrezzo_acquisto(double prezzo_acquisto) + getData() + setData(Date data) + getNome_prodotto() + setNome_prodotto(String nome_prodotto)

Invariante:
context Composizione inv:
self.quantitàAcquistate > 0;
 	
context Composizione inv:
self.codiceProdotto = prodotto.codice;
 	
context Composizione inv:
self.numeroFattura = fattura.numero;

Nome Metodo: + getCodice_prodotto():int
Descrizione Metodo: Viene visualizzato il codice del prodotto relativo alla composizione.

Nome Metodo: + setCodice_prodotto(String codice_prodotto):void
Descrizione Metodo: Viene settato il codice del prodotto relativo alla composizione.

Nome Metodo: + getNumero_fattura():int
Descrizione Metodo: Viene visualizzato il numero della fattura relativa alla composizione.

Nome Metodo: + setNumero_fattura(String numero_fattura):void
Descrizione Metodo: Viene settato il numero della fattura relativa alla composizione.

Nome Metodo: + getQuantita_acquistate():int
Descrizione Metodo: Viene visualizzata la quantità di prodotto acquistata relativa alla composizione.

Nome Metodo: + setQuantita_acquistate(int quantita_acquistate):void
Descrizione Metodo: Viene settata la quantità di prodotto acquistata relativa alla composizione.

Nome Metodo: + getIva_acquisto():int
Descrizione Metodo: Viene visualizzata l’iva relativa alla composizione.

Nome Metodo: + setIva_acquisto(int iva_acquisto):void
Descrizione Metodo: Viene settata l’iva relativa alla composizione.

Nome Metodo: + getPrezzo_acquisto():double
Descrizione Metodo: Viene visualizzato il prezzo d’acquisto relativo alla composizione.

Nome Metodo: + setPrezzo_acquisto(double prezzo_acquisto):void
Descrizione Metodo: Viene settato il prezzo d’acquisto relativo alla composizione.

Nome Metodo: + getPrezzo_acquisto():double
Descrizione Metodo: Viene visualizzato il prezzo d’acquisto relativo alla composizione.

Nome Metodo: + setPrezzo_acquisto(double prezzo_acquisto):void
Descrizione Metodo: Viene settato il prezzo d’acquisto relativo alla composizione.

Nome Metodo: + getNome_prodotto():String
Descrizione Metodo: Viene visualizzato il nome del prodotto relativo alla composizione.

Nome Metodo: + setNome_prodotto(String nome_prodotto):void
Descrizione Metodo: Viene settato il nome del prodotto relativo alla composizione.

EffetuazioneBean

Nome Classe: EffettuazioneBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi alle effettuazioni.
Metodi: + getEmail() + setEmail(String email) + getNumero() + setNumero(String numero)

Invariante:
context Effetuazione inv:
self.email = cliente.email;
 	
context Effetuazione inv:
self.numero = fattura.numero;

Nome Metodo: + getEmail():String
Descrizione Metodo: Viene visualizzata l’email relativa all’effettuazione.

Nome Metodo:+ setEmail(String email):void
Descrizione Metodo: Viene settata l’email relativa all’effettuazione.

Nome Metodo: + getNumero():int
Descrizione Metodo: Viene visualizzato il numero relativo all’effettuazione.

Nome Metodo: + setNumero(String numero):void
Descrizione Metodo: Viene settato il numero relativo all’effettuazione.

ClienteBean

Nome Classe: ClienteBean

Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi al cliente.

Metodi:

+ getEmail()
+ setEmail(String email)
+ getPassword()
+ setPassword(String password)
+ getNome()
+ setNome(String nome)
+ getCognome()
+ setCognome(String cognome)
+ getData_di_nascita()
+ setData_di_nascita(Date data_di_nascita)
+ getVia()
+ setVia(String via)
+ getCap()
+ setCap(String cap)
+ getCitta()
+ setCitta(String citta)
+ getNumero_civico()
+ setNumero_civico(String numero_civico)
+ getPunti()
+ setPunti(int punti)
+ getNumero_di_telefono()
+ setNumero_di_telefono(String numero_di_telefono)
+ getCarte_credito()
+ setCarte_credito(String numero, String tipologia)
+ deleteCarte_credito(String numero)

Invariante:

context Cliente inv:
self.allInstanes()→isUnique(email);

context Cliente inv:
self.email <> ""
self.email.ismatrix('/^\w+([.-]?\w+)@\w+([.-]?\w+)(.\w{2,3})+$/');

context Cliente inv:
self.password.size >=6 AND
self.password.ismatrix('/^[0-9]+$/');

context Cliente inv:
self.nome.ismatrix('/^[A-Za-z]+$/') AND

```
self.cognome.ismatrix('/^[A-Za-z]+$/') AND
self.città.ismatrix('/^[A-Za-z]+$/') AND
self.via.ismatrix('/^[A-Za-z]+$/') AND
self.cap.toInteger <> invalid AND
self.numeroCivico.toInteger <> invalid;

context Cliente inv:
self.punti >= 0;

context Cliente inv:
self.numeroDiTelefono.toInteger <> invalid AND
self.numeroDiTelefono.size = 10;
```

Nome Metodo: + getEmail():String
Descrizione Metodo: Viene visualizzata l'email relativa al cliente.

Nome Metodo: + setEmail(String email):void
Descrizione Metodo: Viene settata l'email relativa al cliente.

Nome Metodo: + getPassword():String
Descrizione Metodo: Viene visualizzata la password relativa al cliente.

Nome Metodo: + setPassword(String password):void
Descrizione Metodo: Viene settata la password relativa al cliente.

Nome Metodo: + getNome():String
Descrizione Metodo: Viene visualizzato il nome relativo al cliente.

Nome Metodo: + setNome(String nome):void
Descrizione Metodo: Viene settato il nome relativo al cliente.

Nome Metodo: + getCognome():String
Descrizione Metodo: Viene visualizzato il cognome relativo al cliente.

Nome Metodo:+ setCognome(String cognome):void
Descrizione Metodo: Viene settato il cognome relativo al cliente.

Nome Metodo: + getData_di_nascita():String
Descrizione Metodo: Viene visualizzata la data di nascita relativa al cliente.

Nome Metodo: + setData_di_nascita(Date data_di_nascita):void
Descrizione Metodo: Viene settata la data di nascita relativa al cliente.

Nome Metodo: + getVia():String
Descrizione Metodo: Viene visualizzata la via relativa al cliente.

Nome Metodo: + setVia(String via):void
Descrizione Metodo: Viene settata la via relativa al cliente.

Nome Metodo: + getCap():String
Descrizione Metodo: Viene visualizzato il Cap relativo al cliente.

Nome Metodo:+ setCap(String cap):void
Descrizione Metodo: Viene settato il Cap relativo al cliente.

Nome Metodo: + getCitta():String
Descrizione Metodo: Viene visualizzata la città relativa al cliente.

Nome Metodo:+ setCitta(String citta):void
Descrizione Metodo: Viene settata la città relativa al cliente.

Nome Metodo: + getNumero_civico():int
Descrizione Metodo: Viene visualizzato il numero civico relativo al cliente.

Nome Metodo:+ setNumero_civico(String numero_civico):void
Descrizione Metodo: Viene settato il numero civico relativo al cliente.

Nome Metodo: + getPunti():int
Descrizione Metodo: Viene visualizzato il numero di punti relativo al cliente.

Nome Metodo:+ setPunti(int punti):void
Descrizione Metodo: Viene settato il numero di punti relativo al cliente.

Nome Metodo: + getNumero_di_telefono():String
Descrizione Metodo: Viene visualizzato il numero di telefono relativo al cliente.

Nome Metodo:+ setNumero_di_telefono(String numero_di_telefono):void
Descrizione Metodo: Viene settato il numero di punti relativo al cliente.

Nome Metodo: + getCarte_credito():HashMap<String,String>
Descrizione Metodo: Vengono visualizzate le carte di credito relative al cliente

Nome Metodo: + setCarte_credito(String numero, String tipologia):HashMap<String,String>
Descrizione Metodo: Viene aggiunta una carta di credito al cliente.

Nome Metodo: + deleteCarte_credito(String numero):void
Descrizione Metodo: Viene eliminata una carta di credito relativa al cliente

AdminBean

Nome Classe: AdminBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi all'admin.
Metodi: + getEmail() + setEmail(String email) + getPassword() + setPassword(String password) + getNome() + setNome(String nome) + getCognome() + setCognome(String cognome) + getData_di_nascita() + setData_di_nascita(Date data_di_nascita) + getVia() + setVia(String via) + getCap() + setCap(String cap) + getCitta() + setCitta(String citta) + getNumero_civico() + setNumero_civico(String numero_civico)

Invariante: context Admin inv:
self.allInstanes()→isUnique(email);
 	
context Admin inv:
self.email <> ""
self.email.ismatrix('/^\w+([.-]?\w+)@\w+([.-]?\w+)(.\w{2,3})+$/');
 	
context Admin inv:
self.password.size >=6 AND
self.password.ismatrix('/^[0-9]+$/');

```
context Admin inv:
self.nome.ismatrix(‘/^[A-Za-z]+$/’)  AND
self.cognome.ismatrix(‘/^[A-Za-z]+$/’)  AND
self.città.ismatrix(‘/^[A-Za-z]+$/’) AND
self.via.ismatrix(‘/^[A-Za-z]+$/’) AND
self.cap.toInteger <> invalid AND
self.numeroCivico.toInteger <> invalid;

context Admin inv:
self.numeroDiTelefono.toInteger <> invalid   AND
self.numeroDiTelefono.size = 10;
```

Nome Metodo: + getEmail():String
Descrizione Metodo: Viene visualizzata l’email relativa all’admin.

Nome Metodo:+ setEmail(String email):void
Descrizione Metodo: Viene settata l’email relativa all’admin.

Nome Metodo: + getPassword():String
Descrizione Metodo: Viene visualizzata la password relativa all’admin.

Nome Metodo:+ setPassword(String password):void
Descrizione Metodo: Viene settata la password relativa all’admin.

Nome Metodo: + getNome():String
Descrizione Metodo: Viene visualizzato il nome relativo all’admin.

Nome Metodo: + setNome(String nome):void
Descrizione Metodo: Viene settato il nome relativo all’admin.

Nome Metodo: + getCognome():String
Descrizione Metodo: Viene visualizzato il cognome relativo all’admin.

Nome Metodo: + setCognome(String cognome):void
Descrizione Metodo: Viene settato il cognome relativo all’admin.

Nome Metodo: + getData_di_nascita():String
Descrizione Metodo: Viene visualizzata la data di nascita relativa all’admin.

Nome Metodo: + setData_di_nascita(Date data_di_nascita):void
Descrizione Metodo: Viene settata la data di nascita relativa all’admin.

Nome Metodo: + getVia():String
Descrizione Metodo: Viene visualizzata la via relativa all’admin.

Nome Metodo: + setVia(String via):void
Descrizione Metodo: Viene settata la via relativa all’admin.

Nome Metodo: + getCap():String
Descrizione Metodo: Viene visualizzato il Cap relativo all’admin.

Nome Metodo: + setCap(String cap):void
Descrizione Metodo: Viene settato il Cap relativo all’admin.

Nome Metodo: + getCitta():String
Descrizione Metodo: Viene visualizzata la città relativa all’admin.

Nome Metodo: + setCitta(String citta):void
Descrizione Metodo: Viene settata la città relativa all’admin.

Nome Metodo: + getNumero_civico():int
Descrizione Metodo: Viene visualizzato il numero civico relativo all’admin.

Nome Metodo: + setNumero_civico(String numero_civico):void
Descrizione Metodo: Viene settato il numero civico relativo all’admin.

AziendaBean

Nome Classe: AziendaBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi all’azienda.
Metodi: + getEmail() + setEmail(String email) + getPassword() + setPassword(String password) + getNome() + setNome(String nome) + getData_di_nascita() + setData_di_nascita(Date data_di_nascita) + getVia() + setVia(String via) + getCap()

+ setCap(String cap) + getCitta() + setCitta(String citta) + getNumero_civico() + setNumero_civico(String numero_civico)

Invariante: context Azienda inv:
self.allInstanes()→isUnique(email);
 	
context Azienda inv:
self.email <> ""
self.email.ismatrix('/^\w+([.-]?\w+)@\w+([.-]?\w+)(.\w{2,3})+$/');
 	
context Azienda inv:
self.password.size >=6 AND
self.password.ismatrix('/^[0-9]+$/');
 	
context Azienda inv:
self.nome.ismatrix('/^[A-Za-z]+$/') AND
self.cognome.ismatrix('/^[A-Za-z]+$/') AND
self.città.ismatrix('/^[A-Za-z]+$/') AND
self.via.ismatrix('/^[A-Za-z]+$/') AND
self.cap.toInteger <> invalid AND
self.numeroCivico.toInteger <> invalid;
 	
context Azienda inv:
self.numeroDiTelefono.toInteger <> invalid AND
self.numeroDiTelefono.size = 10;

Nome Metodo: + getEmail():String
Descrizione Metodo: Viene visualizzata l'email relativa all'azienda.

Nome Metodo: + setEmail(String email):void
Descrizione Metodo: Viene settata l'email relativa all'azienda.

Nome Metodo: + getPassword():String
Descrizione Metodo: Viene visualizzata la password relativa all'azienda.

Nome Metodo: + setPassword(String password):void
Descrizione Metodo: Viene settata la password relativa all'azienda.

Nome Metodo: + getNome():String
Descrizione Metodo: Viene visualizzato il nome relativo all'azienda.

Nome Metodo: + setNome(String nome):void
Descrizione Metodo: Viene settato il nome relativo all’azienda.

Nome Metodo: + getData_di_nascita():String
Descrizione Metodo: Viene visualizzata la data di nascita relativa all’azienda.

Nome Metodo:+ setData_di_nascita(Date data_di_nascita):void
Descrizione Metodo: Viene settata la data di nascita relativa all’azienda.

Nome Metodo: + getVia():String
Descrizione Metodo: Viene visualizzata la via relativa all’azienda.

Nome Metodo: + setVia(String via):void
Descrizione Metodo: Viene settata la via relativa all’azienda.

Nome Metodo: + getCap():String
Descrizione Metodo: Viene visualizzato il Cap relativo all’azienda.

Nome Metodo: + setCap(String cap):void
Descrizione Metodo: Viene settato il Cap relativo all’azienda.

Nome Metodo: + getCitta():String
Descrizione Metodo: Viene visualizzata la città relativa all’azienda.

Nome Metodo:+ setCitta(String citta):void
Descrizione Metodo: Viene settata la città relativa all’azienda.

Nome Metodo: + getNumero_civico():int
Descrizione Metodo: Viene visualizzato il numero civico relativo all’azienda.

Nome Metodo:+ setNumero_civico(String numero_civico):void
Descrizione Metodo: Viene settato il numero civico relativo all’azienda.

RecensioneBean

Nome Classe: RecensioneBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi alla recensione.
Metodi: + getTitolo() + setTitolo(String titolo) + getEmail() + setEmail(String email) + getDescrizione() + setDescrizione(String descrizione)

Invariante:
context Recensione inv:
self.descizione.size > 0;
 	
context Recensione inv:
self.email = cliente.email;
 	
context Recensione inv:
self.email = ricetta.descrizione;

Nome Metodo: + getTitolo():String
Descrizione Metodo: Viene visualizzato il titolo della ricetta relativo alla recensione.

Nome Metodo: + setTitolo(String titolo):void
Descrizione Metodo: Viene settato il titolo della ricetta relativo alla recensione.

Nome Metodo: + getEmail():String
Descrizione Metodo: Viene visualizzata l’email relativa alla recensione.

Nome Metodo: + setEmail(String email):void
Descrizione Metodo: Viene settata l’email relativa alla recensione.

Nome Metodo: + getDescrizione():String
Descrizione Metodo: Viene visualizzata la descrizione relativa alla recensione.

Nome Metodo:+ setDescrizione(String descrizione):void
Descrizione Metodo: Viene settata la descrizione relativa alla recensione.

RicettaBean

Nome Classe: RicettaBean
Descrizione Classe: Classe il cui compito è quello di rappresentare i dati relativi alla ricetta.
Metodi: + getTitolo() + setTitolo(String titolo) + getDescrizione() + setDescrizione(String descrizione) + getCategoria() + setCategoria(String categoria) + getProvenienza() + setProvenienza(String provenienza) + getImmagine() + setImmagine(Blob immagine)
Invariante: context Ricetta inv: self.allInstanes() → isUnique(titolo); context Ricetta inv: self.categoria =”Primo” OR self.categoria =”Secondo” OR self.categoria =”Dolce”; context Ricetta inv: self.provenienza =”Campania” OR self.provenienza =”Puglia” OR self.provenienza =”Sicilia” OR self.provenienza =”Basilicata” OR self.provenienza =”Calabria” OR self.provenienza =”Abruzzo” OR self.provenienza = ”Molise”;

Nome Metodo: + getTitolo():String
Descrizione Metodo: Viene visualizzato il titolo relativo alla ricetta.

Nome Metodo:+ setTitolo(String titolo):void
Descrizione Metodo: Viene settato il titolo relativo alla ricetta.

Nome Metodo: + getDescrizione():String
Descrizione Metodo: Viene visualizzata la descrizione relativa alla ricetta.

Nome Metodo:+ setDescrizione(String descrizione):void
Descrizione Metodo: Viene settata la descrizione relativa alla ricetta.

Nome Metodo: + getCategoria():String
Descrizione Metodo: Viene visualizzata la categoria relativa alla ricetta.

Nome Metodo: + setCategoria(String categoria):void
Descrizione Metodo: Viene settata la categoria relativa alla ricetta.

Nome Metodo: + getProvenienza():String
Descrizione Metodo: Viene visualizzata la provenienza relativa alla ricetta.

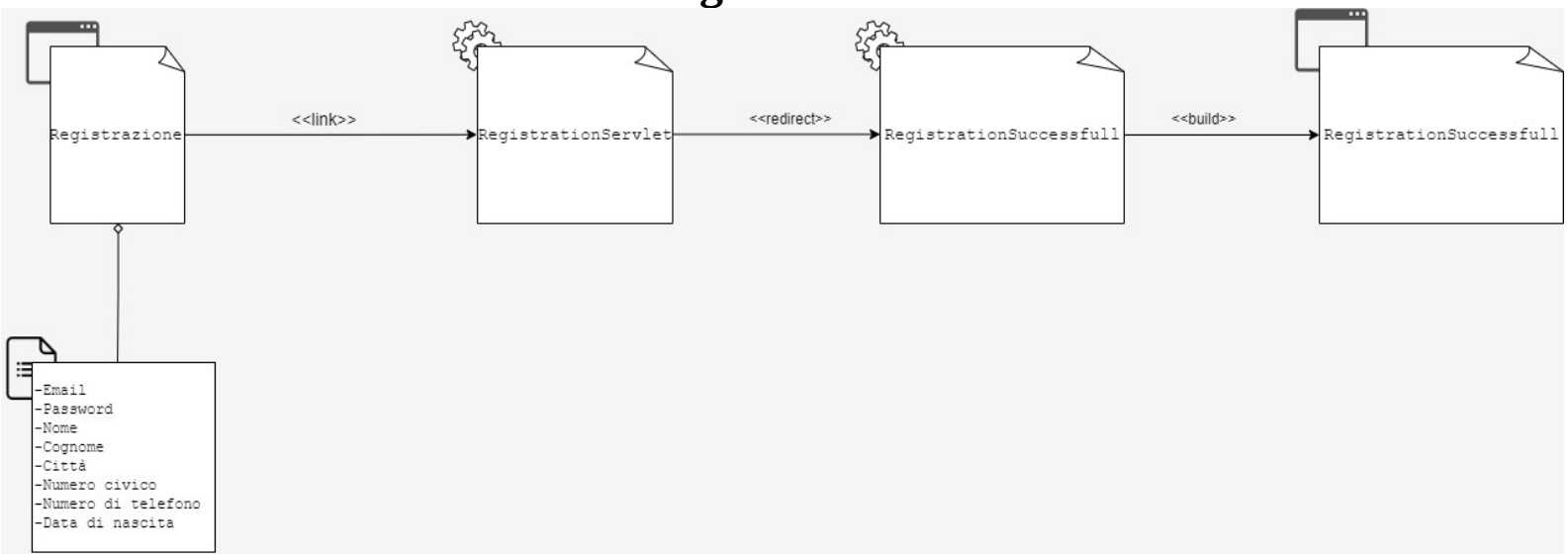
Nome Metodo: + setCategoria(String categoria):void
Descrizione Metodo: Viene settata la provenienza relativa alla ricetta.

Nome Metodo: + getImmagine():Blob
Descrizione Metodo: Viene visualizzata l’immagine relativa alla ricetta.

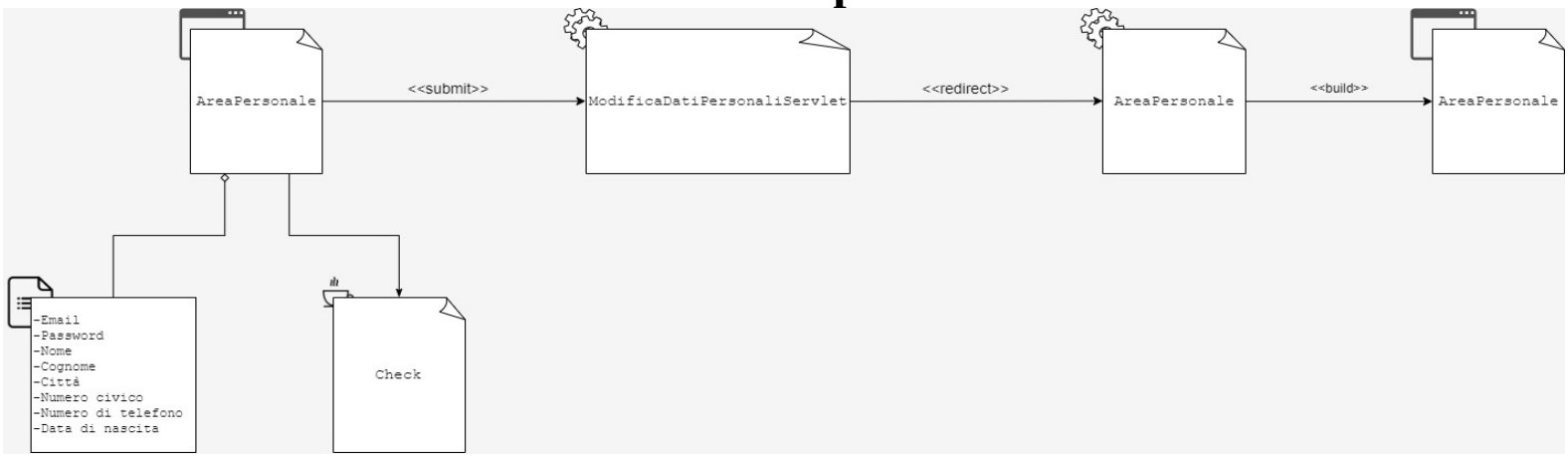
Nome Metodo: + setImmagine(Blob immagine):void
Descrizione Metodo: Viene settata l’immagine relativa alla ricetta.

Diagrammi UML per il Web

Registrazione



Modifica dati personali



Acquisto