Progetto di Ingegneria del Software 2

A.A. 2013/2014

DESIGN DOCUMENT

for

TRAVELDREAM

Casamassima Nicola

Lazzati Riccardo

Marcu Bogdan Alexandru

Indice

1.INTRODUZIONE

1.1 Purpose

Lo scopo del documento di design è definire un'architettura del tutto generale per il sistema che andremo ad implementare.

E' la prima fase di definizione dell'architettura che sarà poi raffinata in fase d'implementazione.

1.2 References

La struttura del documento e le informazioni da inserirvi sono state dedotte dall'analisi delle specifiche fornite dai docenti.

1.3 Acronimi, Definizioni e Abbreviazioni

I seguenti acronimi saranno utilizzati nel documento e verranno qui definiti in modo esplicito:

- RASD: Requirements analysis and specification document.
- DBMS: Database Management System.
- EJB: Enterprise Java Beans.
- J2EE: Java 2 Enterprise Edition.

Definizioni:

- Utenti registrati: utenti iscritti al sistema che possono usufruire dell'intero insieme di funzionalità offerte.
- Utenti non registrati: utenti che visitano il sistema come ospiti.
- Impieati: utenti che gestiscono i componenti offerti alla clientela.

1.4 Overview

Il documento è suddiviso in cinque sezioni:

1. Introduzione

Una breve introduzione contenente lo scopo del documento, la definizione di alcuni termini e acronimi, e le references.

2. Descrizione Architettura

Questa sezione contiene la descrizione dettagliata del tipo di architettura che è stata ritenuta opportuna per lo sviluppo dell'applicazione. Qui vengono identificati i tier e i sottosistemi che verranno implementati.

3. Persistenza dei Dati

In questa sezione viene definita la struttura della base di dati, partendo dal modello concettuale che abbiamo ottenuto nella fase di analisi dei requisiti e arrivando alla struttura dettagliata delle varie tabelle del database.

4. Progettazione in Dettaglio

Sono qui forniti gli schemi dei modelli di navigazione ricavati dai diagrammi presentati nel RASD che hanno permesso di identificare i punti d'interazione fra sistema e utente. Sono inoltre contenuti diagrammi da essi derivati che identificano la struttura dei componenti da implementare (pagine, entità e controller).

5. Rettifiche al documento di RASD

Sono qui fornite le rettifiche al documento di analisi dei requisiti.

2 DESCRIZIONE ARCHITETTURA

2.1 Definizione architettura

Si è scelto di utilizzare un'architettura di tipo client-server, poiché gli utenti che vogliono fruire del servizio dovranno accedervi tramite browser, perciò il lato client l'applicazione dovrà contenere solo una logica di presentazione dei dati che via via vengono richiesti al server attraverso le interazioni con l'utente tramite l'interfaccia del client.

Il server che conterrà l'intera logica applicativa, dovrà gestire le richieste dei client, mantenere le sessioni degli utenti e gestire gli accessi all'unica base di dati.

Siamo quindi giunti a una suddivisione dell'architettura a tre livelli (tier) descritti di seguito:

2.1.2 Tier

Viene qui proposta la descrizione di ognuno dei 3 tier con l'indicazione delle tecnologie che prevediamo di utilizzare per realizzarli:

1. Client Tier

È fondamentalmente composto dalle pagine HTML con cui gli utenti interagiscono con il sistema remoto. Questo tier invia le richieste dell'utente al server tramite il browser web. Le pagine possono essere arricchite con script in linguaggio javascript per gestire la visualizzazione delle informazioni e il passaggio tra le varie schermate.

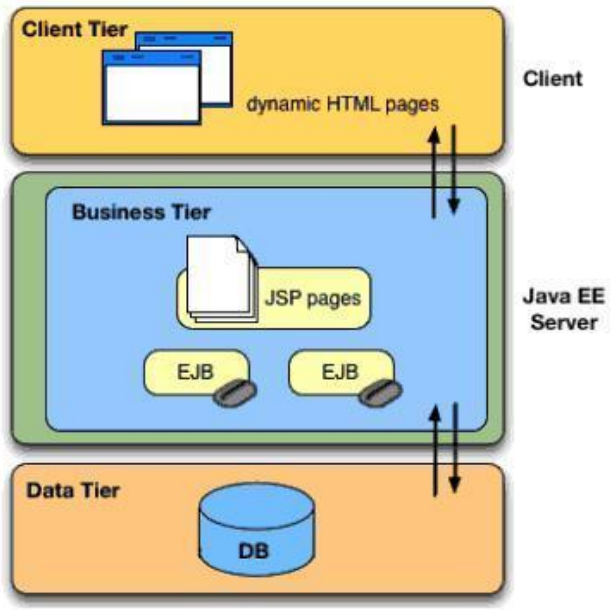
2. Business Tier

Questo tier contiene le pagine web dinamiche (pagine JSP) che il server interpreterà e invierà al client corredandole di opportune informazioni richieste. Contiene inoltre tutti i componenti in grado di gestire le sessioni degli utenti e l'intera logica dell'applicazione. E' inoltre incaricato di recuperare le informazioni dalla base di dati.

3. Data Tier

E' il livello che include le entità nel database e grazie al quale ci si interfaccia ad esse per accedere alle informazioni contenute nella base di dati.

Di seguito schematizzata l'architettura utilizzata:



2.1.3 Sottosistemi

Abbiamo qui elencato i principali sottosistemi da realizzare considerando quali funzionalità i vari attori del sistema devono poter compiere (le quali sono state definite nel documento di RASD).

Sottosistema Accesso

Il sottosistema di accesso permetterà agli utenti di accedere alle funzionalità del sistema. Il punto di accesso sarà unico e, dopo aver verificato i dati di accesso, attiverà il sottosistema corretto che permetterà all'utente di accedere alle funzionalità per lui previste (accesso role-based).

Sottosistema UtenteRegistrato

Questo sottosistema si occuperà di gestire tutti gli aspetti riguardanti gli utenti registrati, cioè conterrà i moduli necessari per fornire le funzionalità di:

o Registrazione

o Ricerca pacchetti

o Gestire e visualizzare propri pacchetti o pacchetti di amici.

Sottosistema Impiegato

Questo sottosistema si occupa della gestione delle funzionalità specifiche per gli amministratori, cioè permette di gestire:

o Creazione, modifica e cancellazione elementi base

o Creazione pacchetti

Sottosistema dati

Questo sottosistema si occuperà di gestire tutte le richieste al database.

2.2 Software

Per l'implementazione prevediamo di usare Java e le tecnologie J2EE. Utilizzeremo Eclipse Kepler come ambiente di sviluppo, MySQL per la base di dati e GlassFish Server come implementazione dell'application server.

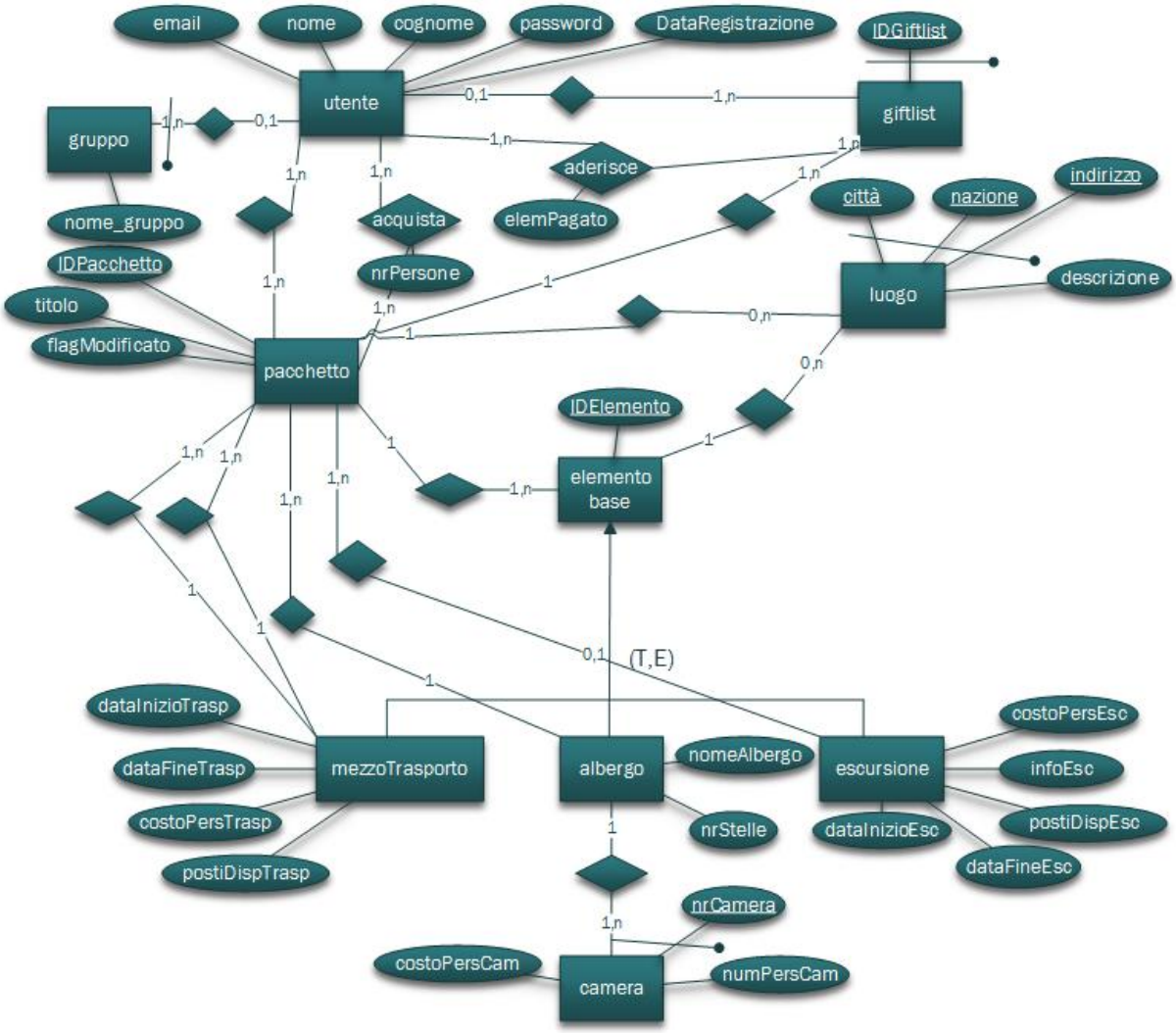
Per il livello di presentazione prevediamo di creare pagine web dinamiche sfruttando JSP dal lato server e in giusta misura Javascript, che girando su lato client, potrebbe tornare utile per alleggerire il carico di lavoro del server.

3. PERSISTENZA DEI DATI

3.1 Lo schema concettuale

Il diagramma ER che segue è utile ai fini di definire quale informazioni siano necessarie da registrare, come ad esempio dati indispensabili riguardanti l'utente o i pacchetti viaggio. Il punto di partenza preso in considerazione è stato il Class Diagram presentato nel documento di analisi dei requisiti, modificato in vista

della creazione del database che conterrà i dati indispensabili del sistema. Perciò, abbiamo ottenuto il seguente diagramma ER:



Note:

- la notazione (T,E) nelle gerarchie indica una relazione totale ed esclusiva;
- l'attributo flagModificato è appunto un flag per segnalare la modifica di un pacchetto da parte dell'utente;
- l'attributo infoEsc dell'entità "escursione" contiene una breve descrizione, in poche parole, dell'escursione; es: gita sul Bosforo, visita al Vaticano
- l'attributo titolo dell'entità "pacchetto" contiene la breve descrizione, simile alla tipologia dell'escursione vista sopra, del pacchetto; es: imperdibile viaggio a Londra, viaggio a Parigi ecc
- l'attributo nome_gruppo dell'entità "gruppo" può avere solamente i valori "user" e "admin";
- la chiave primaria dell'entità "luogo" è una chiave composta da 3 attributi: città, nazione, indirizzo
- l'attributo "descrizione" dell'entità "luogo" può avere solamente i valori "escursione", "aeroporto" e "albergo"

3.2 Lo schema logico

Dallo schema ER otteniamo il seguente schema logico:

Utente(Email, password, nome, cognome, dataRegistrazione);

Gruppo(nome_gruppo, email);

Giflist(idGiftlist, idUser, idPack);

AderisceGiflist(idUtente, idGift, idElementoPagato);

ElementoBase(idViaggio, idAlbergo, idEscursione);

Paccheto(idPacchetto, idLuogoPack, idMezzoAndata, idMezzoRitorno, titolo, numPers, flagModificato);

AcquistaPacchetto(idUtente, idPacch, numPers);

PartecipaPacchetto(idUtente, idPacch);

MezzoTrasporto(idMezzoTrasporto, idLuogoPartenza, idLuogoArrivo, dataInizioTras, dataFineTras, costoPers, postiDispTras);

Luogo(nazione, città, indirizzo, descrizione);

Albergo(idAlbergo, idLuogoAlb, nome, nrStelle);

Camera(idCamera, idAlb, nrCamera, numPersCam, costoPersCam);

Escursione(idEscursione, idLuogoEsc, dataInizioEsc, dataFineEsc, postiDispEsc, infoEsc);

EscursioniPacchetto(idEsc, idPacch);

CameraPacchetto(idCam, idPacch);

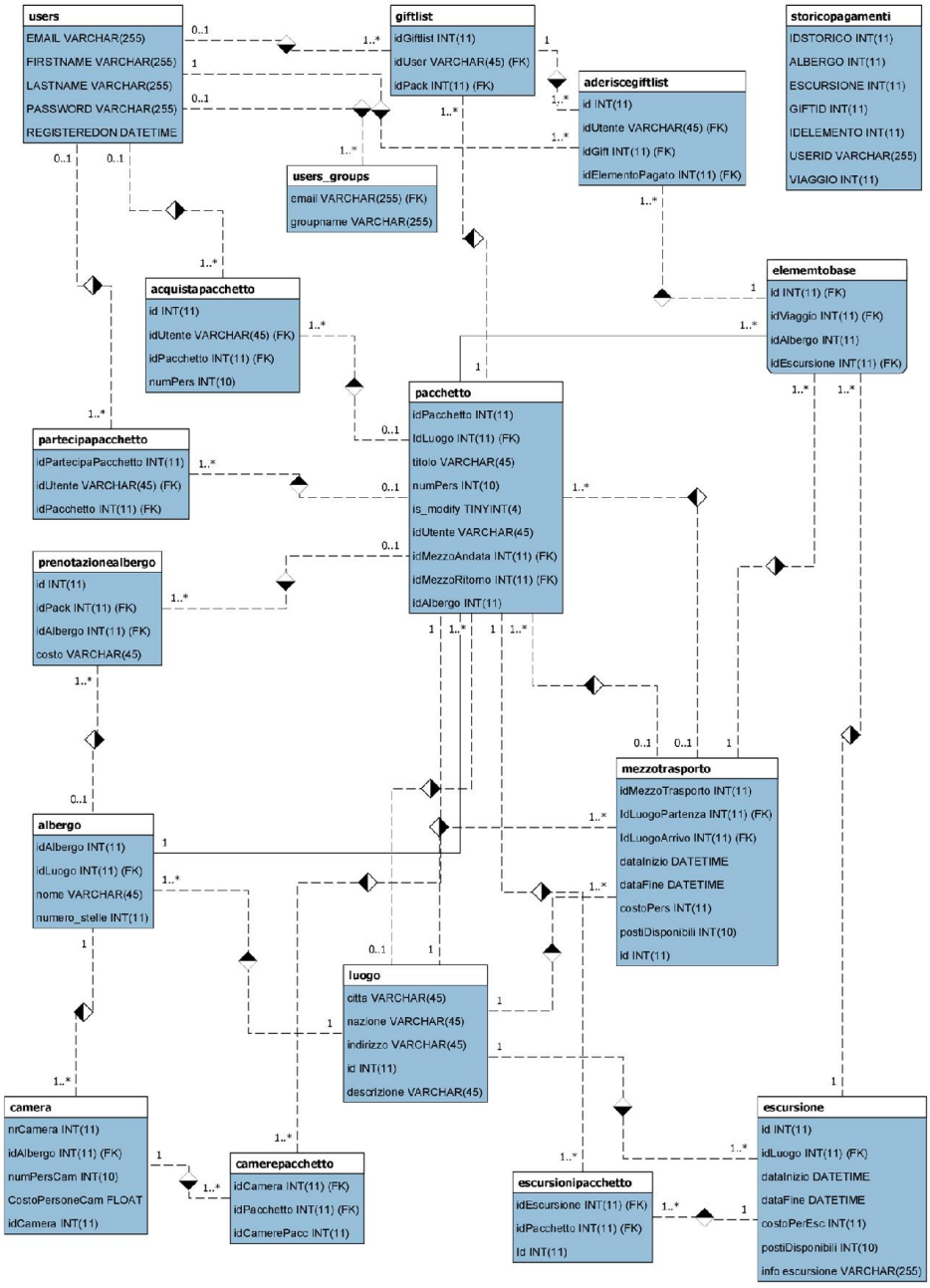
PrenotazioneAlbergo(idAlb, idPacch);

StoricoPagamenti(idStorico, ALBERGO, ESCURSIONE, GIFTID, IDELEMENTO, USERID, VIAGGIO);

Note:

- i campi sottolineati rappresentano le chiavi primarie, i campi in blu rappresentano le chiavi esterne;
- si è deciso di inserire anche una tabella StoricoPagamenti per tenere traccia dei movimenti di denaro;

Riportiamo l'immagine del modello della nostra base di dati:

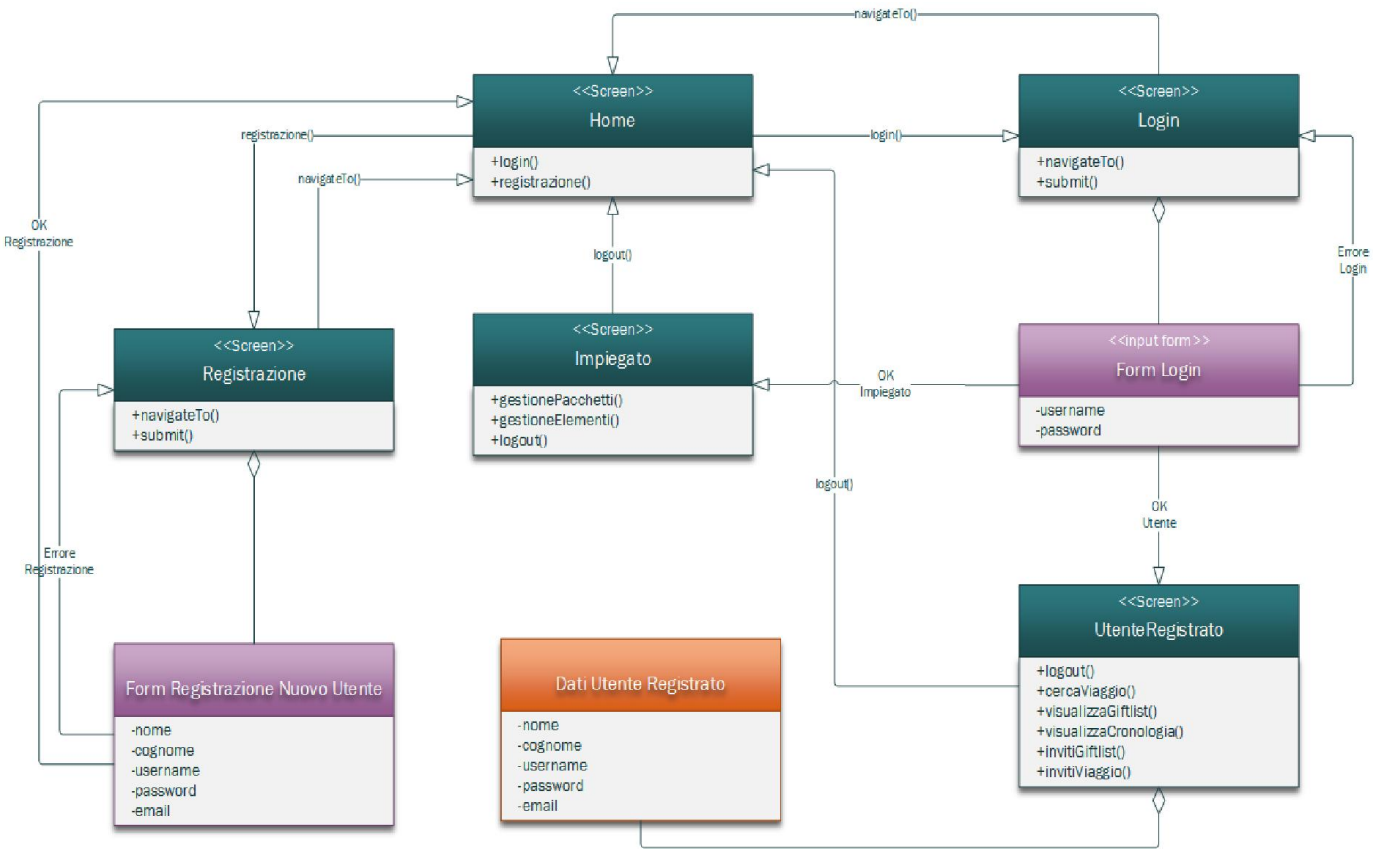

users
EMAIL VARCHAR(255)
FIRSTNAME VARCHAR(255)
LASTNAME VARCHAR(255)
PASSWORD VARCHAR(255)
REGISTEREDON DATETIME
giftlist
idGiftlist INT(11)
idUser VARCHAR(45) (FK)
idPack INT(11) (FK)
storicopagamenti
IDSTORICO INT(11)
ALBERGO INT(11)
ESCURSIONE INT(11)
GIFTID INT(11)
IDELEMENTO INT(11)
USERID VARCHAR(255)
VIAGGIO INT(11)
aderiscegiftlist
id INT(11)
idUtente VARCHAR(45) (FK)
idGift INT(11) (FK)
idElementoPagato INT(11) (FK)
users_groups
email VARCHAR(255) (FK)
groupname VARCHAR(255)
acquistapacchetto
id INT(11)
idUtente VARCHAR(45) (FK)
idPacchetto INT(11) (FK)
numPers INT(10)
elememtobase
Id INT(11) (FK)
idViaggio INT(11) (FK)
idAlbergo INT(11)
idEscursione INT(11) (FK)
pacchetto
idPacchetto INT(11)
IdLuogo INT(11) (FK)
titolo VARCHAR(45)
numPers INT(10)
is_modify TINYINT(4)
idUtente VARCHAR(45)
idMezzoAndata INT(11) (FK)
idMezzoRitorno INT(11) (FK)
IdAlbergo INT(11)
partecipapacchetto
idPartecipaPacchetto INT(11)
idUtente VARCHAR(45) (FK)
idPacchetto INT(11) (FK)
prenotazionealbergo
id INT(11)
idPack INT(11) (FK)
idAlbergo INT(11) (FK)
costo VARCHAR(45)
mezzotrasporto
idMezzoTrasporto INT(11)
IdLuogoPartenza INT(11) (FK)
IdLuogoArrivo INT(11) (FK)
dataInizio DATETIME
dataFine DATETIME
costoPers INT(11)
postiDisponibili INT(10)
id INT(11)
albergo
idAlbergo INT(11)
idLuogo INT(11) (FK)
nome VARCHAR(45)
numero_stelle INT(11)
luogo
citta VARCHAR(45)
nazione VARCHAR(45)
indirizzo VARCHAR(45)
id INT(11)
descrizione VARCHAR(45)
camera
nrCamera INT(11)
idAlbergo INT(11) (FK)
numPersCam INT(10)
CostoPersoneCam FLOAT
idCamera INT(11)
camerepacchetto
idCamera INT(11) (FK)
idPacchetto INT(11) (FK)
idCamerePacc INT(11)
escursionipacchetto
idEscursione INT(11) (FK)
idPacchetto INT(11) (FK)
Id INT(11)
escursione
Id INT(11)
idLuogo INT(11) (FK)
dataInizio DATETIME
dataFine DATETIME
costoPerEsc INT(11)
postiDisponibili INT(10)
info escursione VARCHAR(255)

4 PROGETTAZIONE IN DETTAGLIO

4.1 Modello di navigazione

Nel seguente modello mostriamo come avviene la navigazione attraverso la spiegazione di come gli <<screen>> interagiscono tra loro.

Il modello è suddiviso in più parti secondo una logica che prevede di raggruppare le navigazioni inerenti uno stesso modulo nello stesso schema.



4.1.1 Modello generale

Dallo <<screen>> Home possiamo decidere se fare login o iscriverci tramite i metodi login() e registrazione().

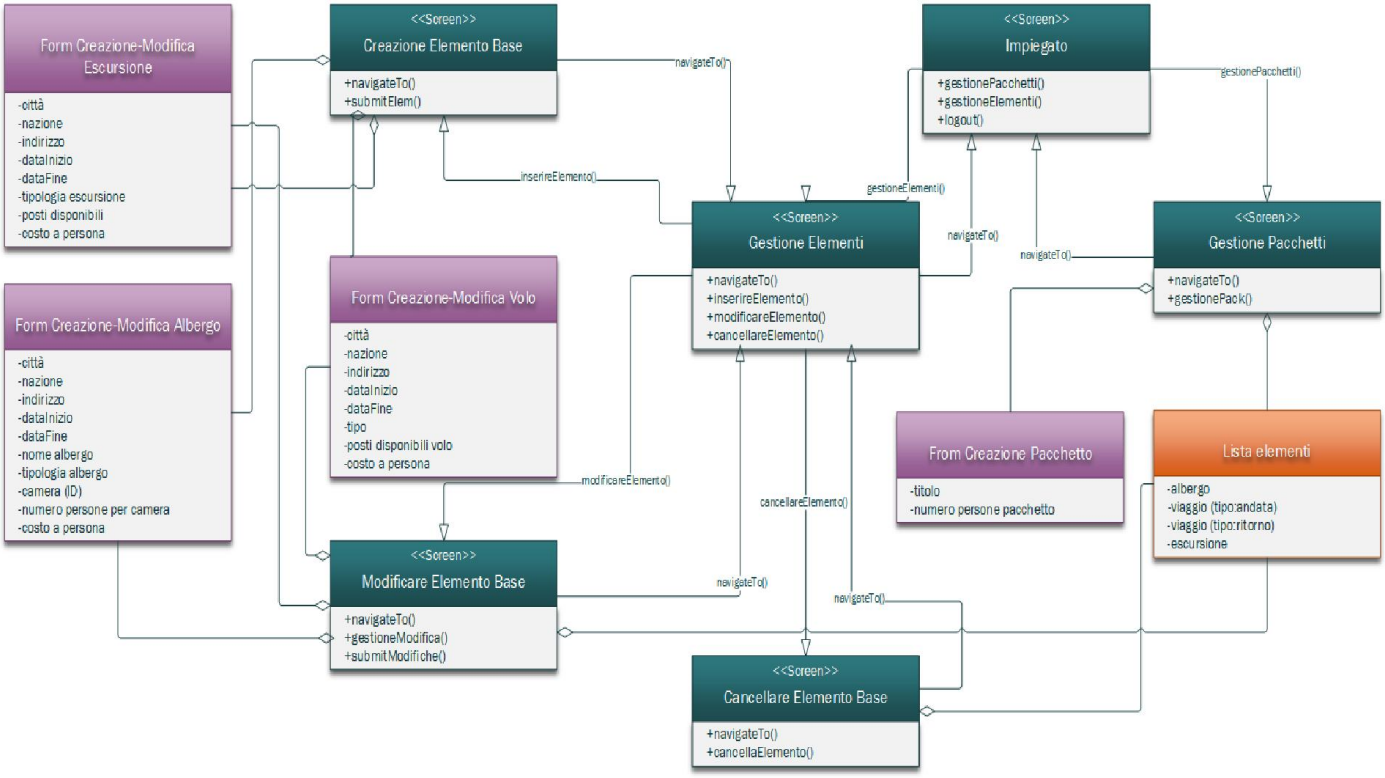
Una volta effettuato il login il sistema determinerà se siamo impiegati o semplici utenti portandoci rispettivamente agli <<screen>> Impiegato e UtenteRegistrato.

La registrazione ovviamente riguarderà solamente l'utente normale dato che, come noto, l'impiegato non ha modo di registrarsi tramite l'interfaccia del sito.

Di seguito vengono sviluppate le navigazioni di un Impiegato e di un UtenteRegistrato.

4.1.2 Modello Impiegato

Dallo screen Impegato sono raggiungibili i seguenti screen:

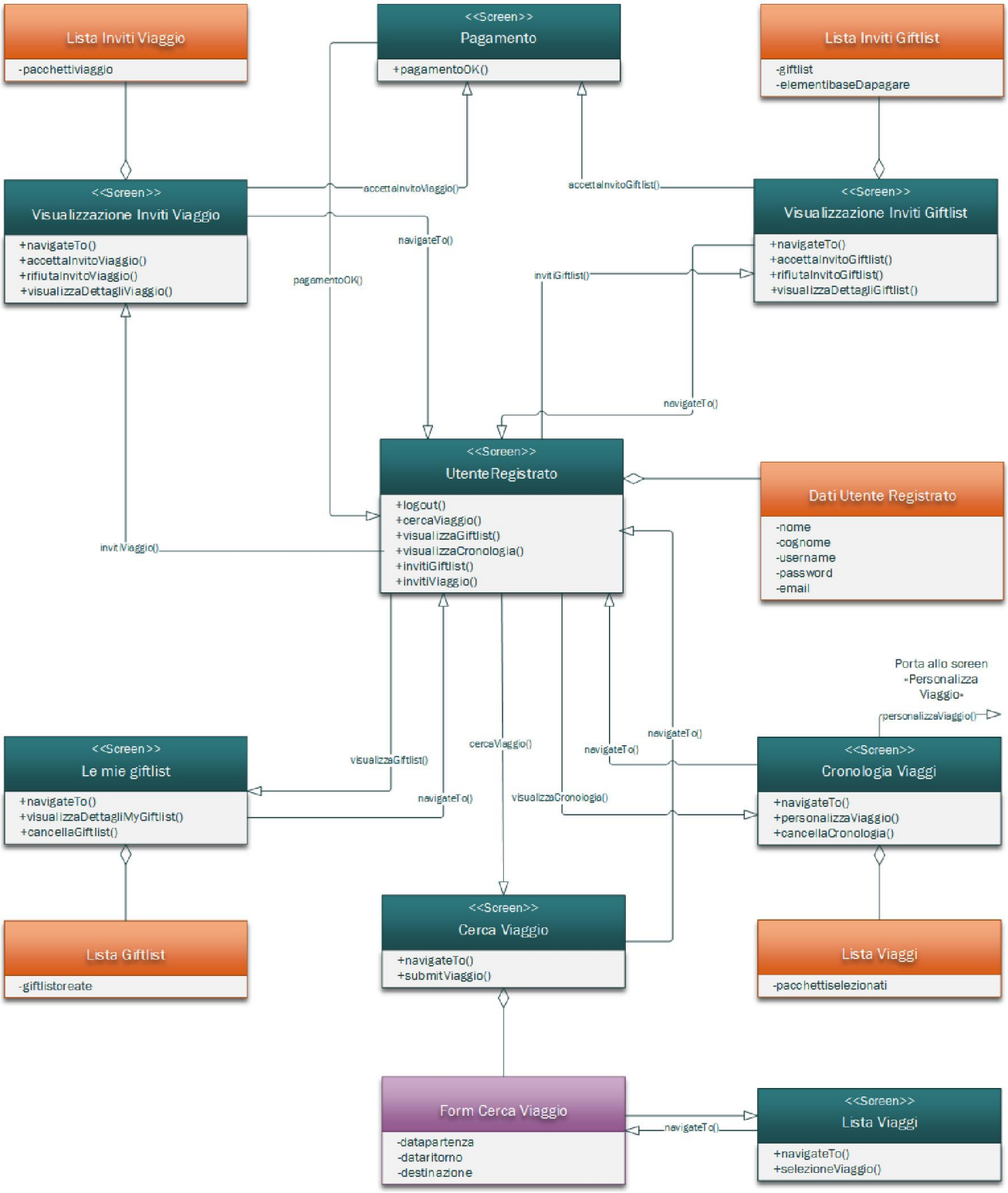


I metodi navigateTo() permettono la navigazione da uno screen all'altro se questa non è già garantita da un metodo.

L'impiegato che accede alla gestione elementi, prima di visualizzare la Lista elementi deve scegliere l'operazione da fare (modifica o cancella Elemento).

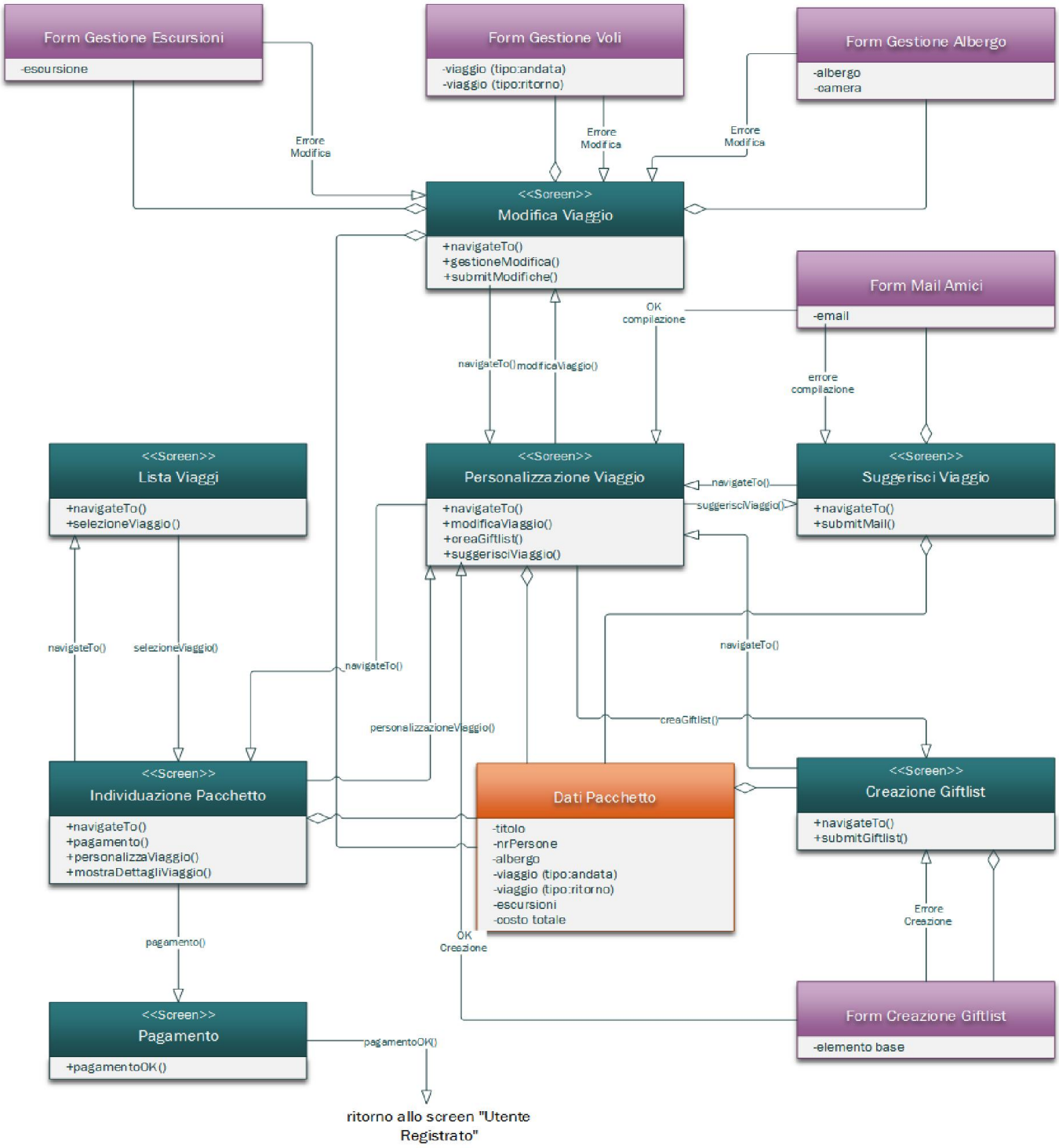
Nella gestione pacchetti invece, vi è la possibilità di crearne uno tramite gestionePack().

4.1.3 Modello Utente Registrato



Dallo screen CronologiaViaggi è possibile, tramite il metodo personalizzaViaggio(), accedere ai seguenti screen:

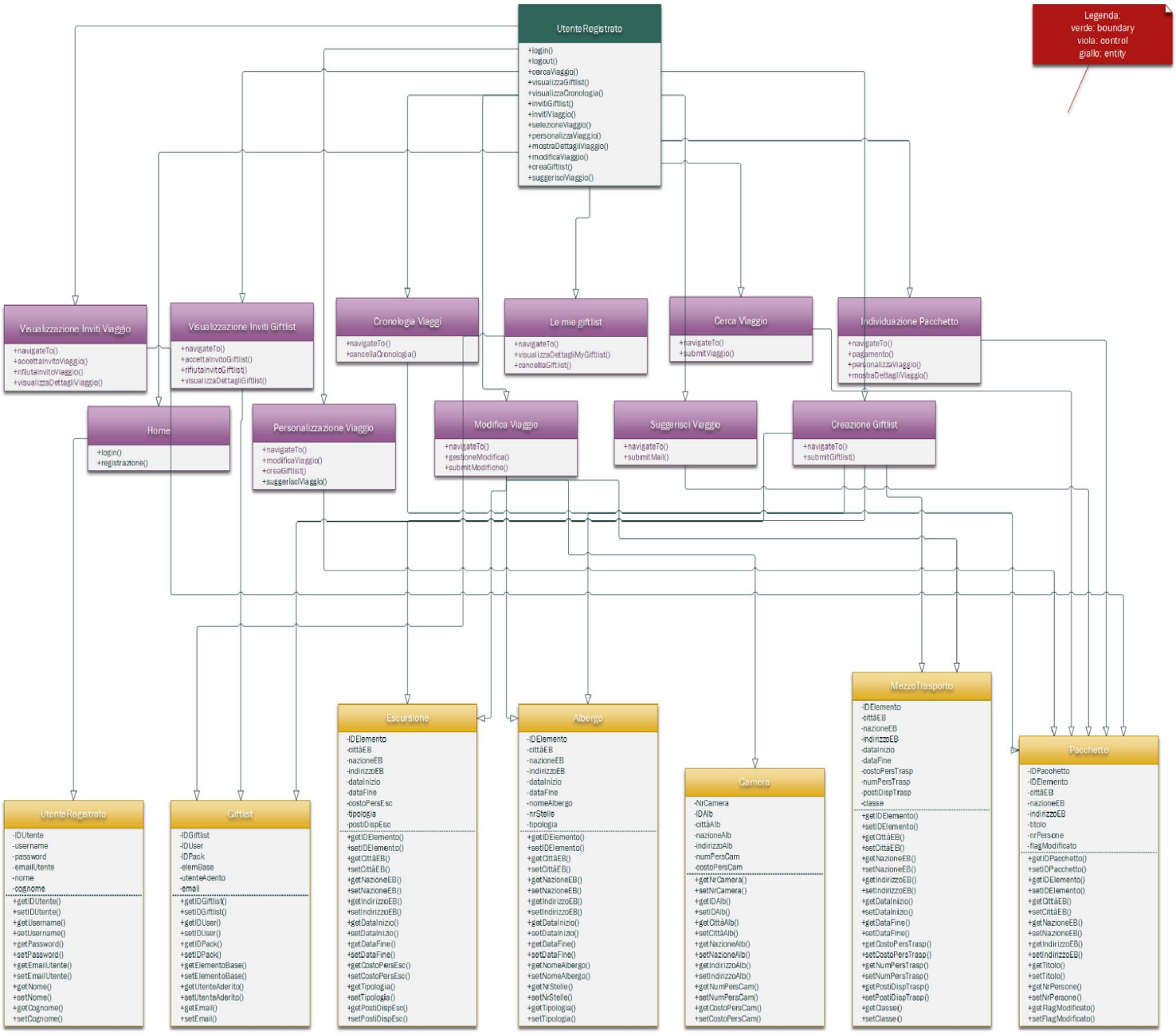
4.1.4 Modello personalizzazione viaggio



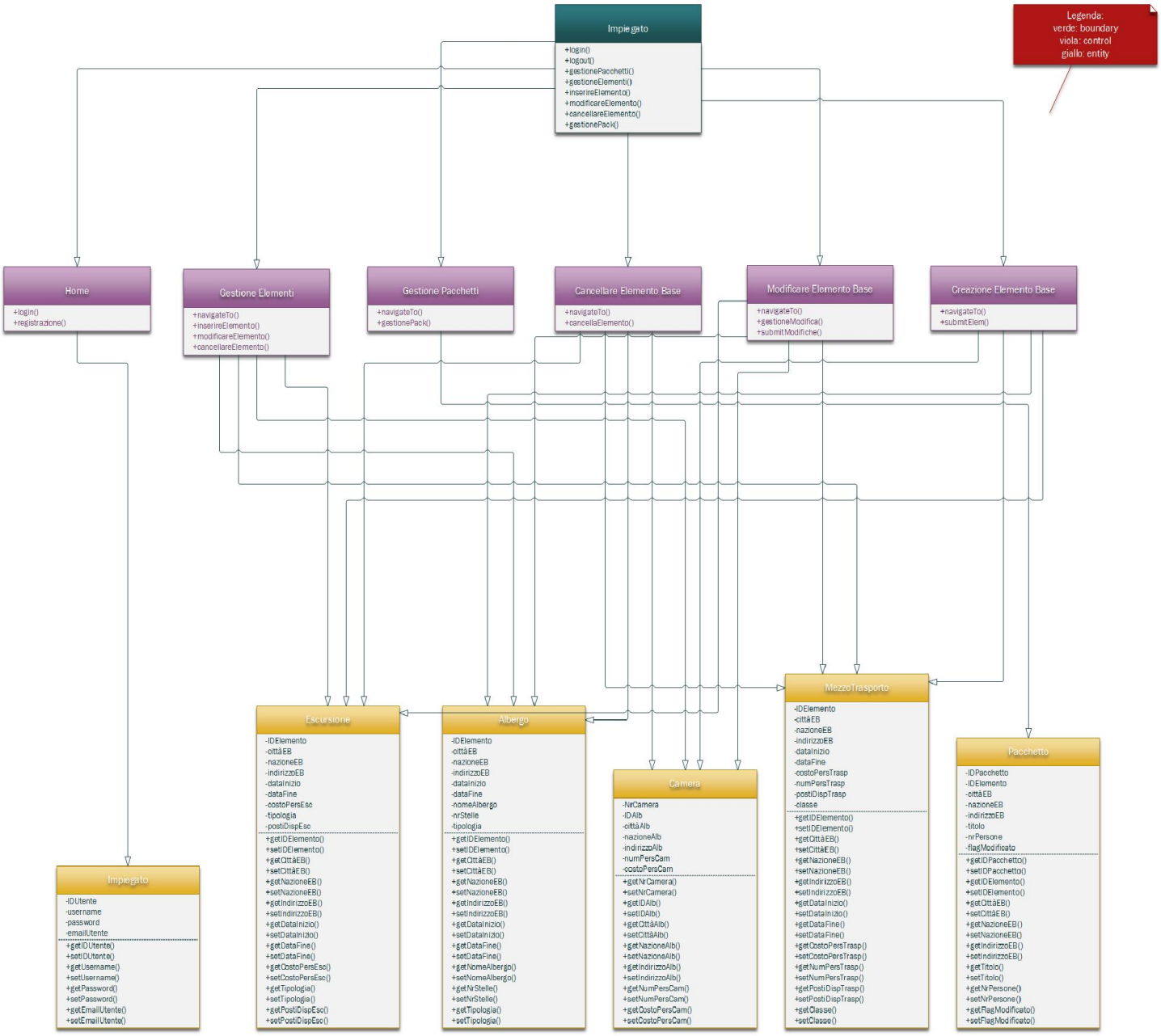
4.2 Diagrammi di analisi

Qui vengono mostrati i diagrammi di analisi relativi alle due tipologie di utenze presenti nel sito.

4.2.1 Scenari utente registrato



4.2.2 Scenari impiegato



5 INTEGRAZIONI AL RASD

5.1 Amministrazione pacchetto creato dall'impiegato

Come già specificato nel RASD l'impiegato può creare un nuovo pacchetto partendo dagli elementi base presenti nel data base.

In questa sezione si vuole specificare che tale pacchetto, una volta creato, non può essere modificato dall'impiegato ma solo dall'utente che eventualmente lo acquisterà.

Nel caso di inserimento errato di dati del pacchetto, si può procedere alla creazione dello stesso ex novo, selezionando tale pacchetto e ripetendo la normale procedura di creazione di questo.

Tutto ciò è fattibile a partire dallo screen GestionePacchetto nel quale viene mostrato l'elenco dei pacchetti.

5.2 Eliminazione della sezione "Dati Personali"

Nella sezione User Interface del RASD abbiamo creato alcuni mock ups per mostrare a grandi linee l'interfaccia dell'utente registrato. A quel punto avevamo deciso di gestire la modifica dei dati personali da parte dell'utente registrato, però ai fini dell'implementazione questa funzionalità si allontana dal vero scopo del sistema, per cui abbiamo deciso di non inserirla nel progetto: i dati personali saranno memorizzati al momento della registrazione e non potranno più essere modificati.

Infatti, come si è visto nel modello di navigazione dell'utente registrato, non si ha uno screen "Dati Personali", bensì solamente le funzionalità relative ai vari viaggi e/o giftlist.

5.3 La possibilità di personalizzare un viaggio a partire dalla cronologia

Questa funzionalità equivale alla modifica in un secondo momento del pacchetto scelto. Con modifica si intendente la sua personalizzazione, cioè aggiunta o cancellazione di elementi e/o modifica di elementi; creazione di giftlist a partire dal pacchetto selezione, così come mandare inviti ad amici.

Proprio per questo, abbiamo aggiunto un collegamento tra lo screen "Cronologia viaggi" e "Personalizza Viaggio" nel modello di navigazione dell'utente registrato.

Per quanto riguarda i diagrammi di sequenza e di attività di questo caso d'uso, cambia pochissimo rispetto ai diagrammi già presentati nel RASD, quindi brevemente una descrizione chiarirà il cammino: la condizione d'ingresso è trovarsi nella sezione cronologia viaggi; l'utente seleziona di personalizzare un viaggio scelto; il sistema gli ritorna la schermata della personalizzazione. A questo punto il funzionamento è identico a quelli presentati.