Design Document

TravelDream

Cirigliano Andrea

Di Sabatino Di Diodoro Serena

Sommario

1. Introduction

1.1 Purpose

Lo scopo di questo documento è quello di descrivere l'architettura generale del sistema TravelDream.
Illustreremo dunque le decisioni prese in relazione alla struttura di tale applicazione giustificandole opportunamente e raffinandole, se necessario, in fase di implementazione.

1.2 Scope

L'architettura della piattaforma deve essere in grado di gestire le principali funzionalità del sistema, che possiamo dividere in 3 ambiti principali:

1. **Utente**
   L'applicazione si occupa della registrazione, autenticazione e disconnessione degli utenti.

2. **Pacchetto viaggio standard**
   L'applicazione permette la gestione dei pacchetti viaggio standard e dei suoi componenti.

3. **Pacchetto viaggio custom**
   L'applicazione permette la gestione dei pacchetti viaggio custom e dei suoi componenti.

1.3 Definitions, acronyms, abbreviations

- **TD**: TravelDream.
- **J2EE**: Java 2 Enterprise Edition.
- **EJB**: Enterprise Java Beans.
- **JSP**: JavaServer Pages.
- **RDBMS**: Relational DataBase Management System.
- **ER**: Entity Relationship.
- **MVC**: Model-View-Controller.
- **ECB**: Entity-Control-Boundary.
- **FR**: Functional Requirement (Requisito Funzionale).
- **RASD**: Requirements Analysis and Specification Document.
- **ID PACKAGE**: Codice ID relativo a un pacchetto viaggio custom.

- **Pacchetto Standard**: Pacchetto viaggio non personalizzato, con componenti predefiniti.
- **Customizzazione**: Personalizzazione di pacchetti viaggio da parte di un cliente.
- **Pacchetto Custom**: Pacchetto personalizzato dal cliente.
- **Componenti**: Prodotti offerti dai singoli pacchetti viaggio. Essi sono principalmente di 3 tipi: volo, hotel, escursione. In futuro si potranno aggiungere altri mezzi di trasporto.
- **Utente Registrato**: Cliente iscritto al sistema.
- **Utente Non Registrato**: Un qualsiasi utente non iscritto al sistema; può essere anche l'amico di un cliente che è stato invitato a visualizzare un pacchetto viaggio.
- **Amministratore**: Impiegato TravelDream che gestisce il sistema.
- **Condivisione** di un pacchetto personalizzato: Attività dell'utente registrato di inviare ad un utente non registrato l'ID PACKAGE di un suo pacchetto custom al fine di mostrargli il contenuto. Tale attività è esterna al sistema sviluppato.

1.4 References

Il design del sistema fa riferimento alle informazioni raccolte dal documento di specifica messo a disposizione dai docenti sulla piattaforma Beep e soprattutto al documento di analisi RASD precedentemente redatto.

1.5 Overview

Questo documento è stato suddiviso nelle seguenti sezioni:

1. **Introduction**

   Presentazione dello scopo del documento e definizione di termini ed elementi ricorrenti.

2. **Design Overview**

   Descrizione generale dell'architettura del sistema in relazione alle sue funzionalità e alle tecnologie che si intendono utilizzare.

3. **Database Design**

   Presentazione della struttura dettagliata del RDBMS: questa sezione contiene il modello concettuale e logico.

4. **Detailed System Design**

Presentazione dettagliata dell’architettura del sistema: questa sezione contiene i modelli di navigazione che descrivono l’interazione con gli utenti e la specifica dei vari componenti da implementare (pagine, entità, controllori).

2. Design Overview

2.1 Design Context

Questa sezione stabilisce i vincoli per il design del sistema specificando funzionalità e contesto tecnologico dell'applicazione.

2.1.1 Functionalities

Nel RASD avevamo identificato i seguenti requisiti funzionali suddivisi in base agli utenti che interagiscono con il sistema:

- **Utente Registrato**:

  [FR1] Autenticazione.
  [FR2] Visualizzazione dei pacchetti viaggio disponibili.
  [FR3] Ricerca dei pacchetti viaggio.
  [FR4] Visualizzazione dettagli dei pacchetti standard.
  [FR5] Acquisto dei pacchetti standard.
  [FR6] Customizzazione del pacchetto standard scegliendo almeno un componente tra quelli disponibili.
  [FR7] Salvataggio del pacchetto customizzato.
  [FR8] Visualizzazione dell' ID PACKAGE relativo al pacchetto custom.
  [FR9] Visualizzazione dei pacchetti salvati.
  [FR10] Visualizzazione dei dettagli dei pacchetti customizzati.
  [FR11] Eliminazione dei pacchetti customizzati.
  [FR12] Acquisto dei pacchetti personalizzati.
  [FR13] Log out.
  [FR28] Ricerca pacchetto custom tramite ID PACKAGE.

- **Utente Non Registrato**:

  [FR14] Registrazione.
  [FR15] Inserimento ID PACKAGE.
  [FR16] Visualizzazione dei dettagli del pacchetto customizzato.

- **Amministratore**:

  [FR17] Autenticazione.
  [FR18] Visualizzazione di pacchetti viaggio standard disponibili.
  [FR19] Creazione di pacchetti viaggio standard.

[FR20] Ricerca di pacchetti viaggio standard.
[FR21] Modifica di pacchetti viaggio standard.
[FR22] Eliminazione di pacchetti viaggio standard.
[FR23] Visualizzazione dei componenti dei pacchetti viaggio.
[FR24] Eliminazione dei componenti dei pacchetti viaggio.
[FR25] Modifica dei componenti dei pacchetti viaggio.
[FR26] Creazione dei componenti dei pacchetti viaggio.
[FR27] Log out.

2.1.2 System Technologies

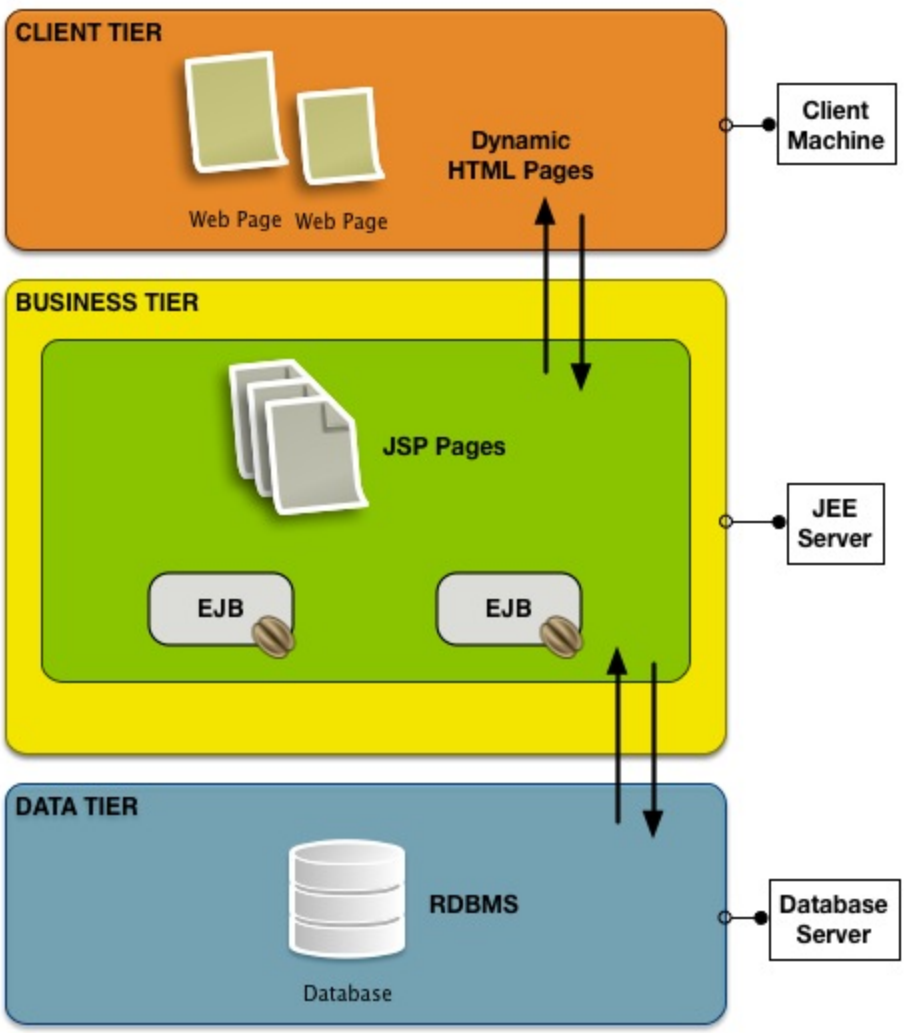
Dopo aver analizzato le principali funzionalità della nostra applicazione abbiamo deciso di utilizzare un'architettura di tipo client-server multilivello.
In particolare sfrutteremo la soluzione three-tier (livelli) tipica che prevede tre diversi moduli o strati dedicati rispettivamente alla interfaccia utente, alla logica funzionale (business logic) e alla gestione dei dati persistenti.
Ogni livello richiede delle tecnologie specifiche:

1. **CLIENT TIER**
   Il Browser Web a questo livello invia le richieste dell'utente al Server Web, il quale genera in risposta pagine HTML dinamiche. Esse potrebbero contenere codice JavaScript per facilitare l'interazione con gli utenti e non sovraccaricare il server con ulteriori richieste lato client.

2. **BUSINESS TIER**
   Questo livello contiene la logica del sistema: l'Application Server J2EE, tramite i componenti software EJB, interpreta le richieste inviate dagli utenti e genera le pagine dinamiche JSP. Le informazioni necessarie per la creazione di tali pagine vengono recuperate dal database del livello sottostante.

3. **DATA TIER**
   Questo livello contiene l'RDBMS MySQL Server 5.6 che, con una semplice interfaccia, permette di accedere ai dati immagazzinati nella base dei dati dell'applicazione TravelDream.

La seguente immagine rappresenta l'architettura generale della nostra applicazione:



Img 1.1 Architettura Three-Tier

2.1.3 Subsystems

Sulla base delle funzionalità dell'applicazione che abbiamo richiamato nel paragrafo 2.1.1 possiamo individuare i seguenti sottosistemi:

1. **SOTTOSISTEMA ACCESSO**
   Tale sottosistema si occupa della gestione e controllo degli accessi: garantisce che le diverse tipologie di utenti registrati (clienti o amministratori) vengano indirizzati alle loro rispettive interfacce e usufruiscano solo dei servizi previsti per loro.
   L'accesso è una funzionalità critica del nostro sistema poichè da esso dipende la corretta interazione con gli utenti quindi utilizzeremo i sistemi di accesso sicuro messi a disposizione da J2EE.

2. **SOTTOSISTEMA UTENTE**[1]
   Tale sottosistema gestisce le principali funzionalità di un utente, registrato e non. In particolare si occupa dei seguenti servizi:
   - Registrazione.
   - Richieste relative ai pacchetti standard (ricerca, visualizzazione, acquisto).
   - Richieste relative ai pacchetti custom (creazione, salvataggio, visualizzazione, ricerca tramite ID PACKAGE, acquisto, eliminazione).

3. **SOTTOSISTEMA AMMINISTRATORE**
   Tale sottosistema gestisce le principali funzionalità dell'amministratore TD. In particolare si occupa dei seguenti servizi:
   - Richieste relative alla gestione dei pacchetti standard (creazione, ricerca, visualizzazione, modifica, eliminazione).
   - Richieste relative alla gestione dei componenti dei pacchetti (creazione, visualizzazione, modifica, eliminazione).

4. **SOTTOSISTEMA DATI**
   Tale sottosistema si occupa della gestione delle richieste al database.

2.2 System Specification

Il nostro sistema è sviluppato in Java utilizzando l'ambiente di sviluppo Eclipse (Juno 4.2) e le tecnologie J2EE.
La seguente tabella mostra le tecnologie usate in fase di implementazione per ogni livello:

Tier	Technology
Client Tier	XHTML - JavaScript
Business Tier	JSP - EJB - Glassfish Server 4.0
Data Tier	MySQL 5.6.14

Tab 1.1 Tecnologie Utilizzate

[1] Non abbiamo inserito un Sottosistema Utente Non Registrato perchè la visualizzazione dei dettagli di un pacchetto custom tramite l'inserimento dell'ID PACKAGE è un servizio di cui può usufruire chiunque: utenti registrati e non.

3. Database Design

In questa sezione viene mostrato il design del Data Tier, pensato per ospitare tutti i dati necessari all'applicazione TravelDream. La struttura viene mostrata inizialmente attraverso un modello di tipo concettuale, con un livello di astrazione tale da potersi concentrare sugli elementi del mondo reale e indipendentemente dagli aspetti realizzativi. Successivamente viene analizzato lo schema logico che introduce una parte della semantica dell'applicazione.

3.1 Conceptual Schema

Il design concettuale viene svolto con il modello Entity - Relationship. Le principali classi di entità sono gli utenti, i pacchetti viaggio, e i componenti.

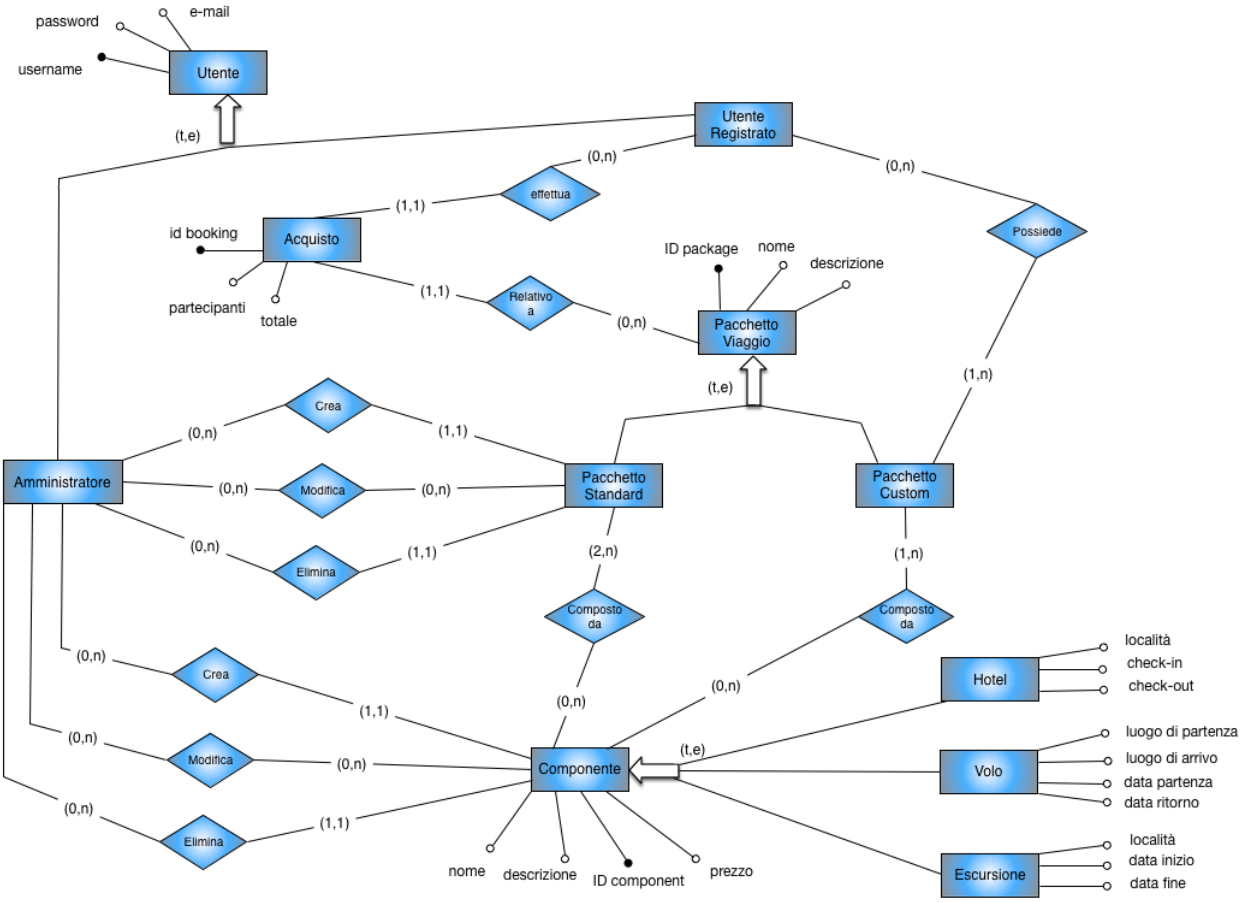
- Le entità utente registrato e amministratore vengono modellizzate come sottoinsiemi di utente, tramite la relazione di generalizzazione totale ed esclusiva. Non è prevista un'entità per l'attore utente non registrato, che pur interagendo con il sistema, non richiede che alcuna informazione ad esso relativa venga mantenuta nel sistema.

- Le entità pacchetto standard e pacchetto custom sono specializzazioni della entità più generale pacchetto viaggio. Ogni pacchetto viaggio, indipendentemente dal tipo, è identificato da un ID package univoco. Nel caso in cui il pacchetto sia di tipo custom l'ID package è quello di cui si fa riferimento nel paragrafo 1.1.3 "Definitions, acronyms, abbreviations". L'ID package di un pacchetto standard, invece, è un'informazione non visibile agli utenti (registrati e non) e in nessun modo utilizzabile nelle funzioni di ricerca tramite ID.

- Le entità volo, hotel ed escursione, sono state modellizzate come specializzazione dell'entità componente. La distinzione è necessaria in quanto diversi componenti necessitano di diversi tipi di attributi; inoltre viene garantita la possibilità di aggiungere facilmente altri tipi di componenti.

Ogni entità acquisto è legata ad uno e un solo utente registrato tramite l'associazione effettua; inoltre ciascun acquisto si riferisce ad uno e un solo pacchetto viaggio.
L'associazione possiede lega ogni utente registrato ai pacchetti viaggio personalizzati che ha salvato e che può consultare nella sezione "I miei pacchetti".
Ogni pacchetto viaggio è associato ad un insieme di componenti che lo costituiscono attraverso l'associazione composto da.

I pacchetti custom devono essere composti da almeno uno (qualsiasi) dei componenti.
I pacchetti standard sono composti da almeno due componenti, un volo ad un hotel. Quest'ultimo vincolo (specificato nel paragrafo 2.5 "Assumptions And Dependencies" del RASD) non è stato interamente espresso nello schema ER per non appesantire troppo il modello.



Img 2.1 Diagramma ER

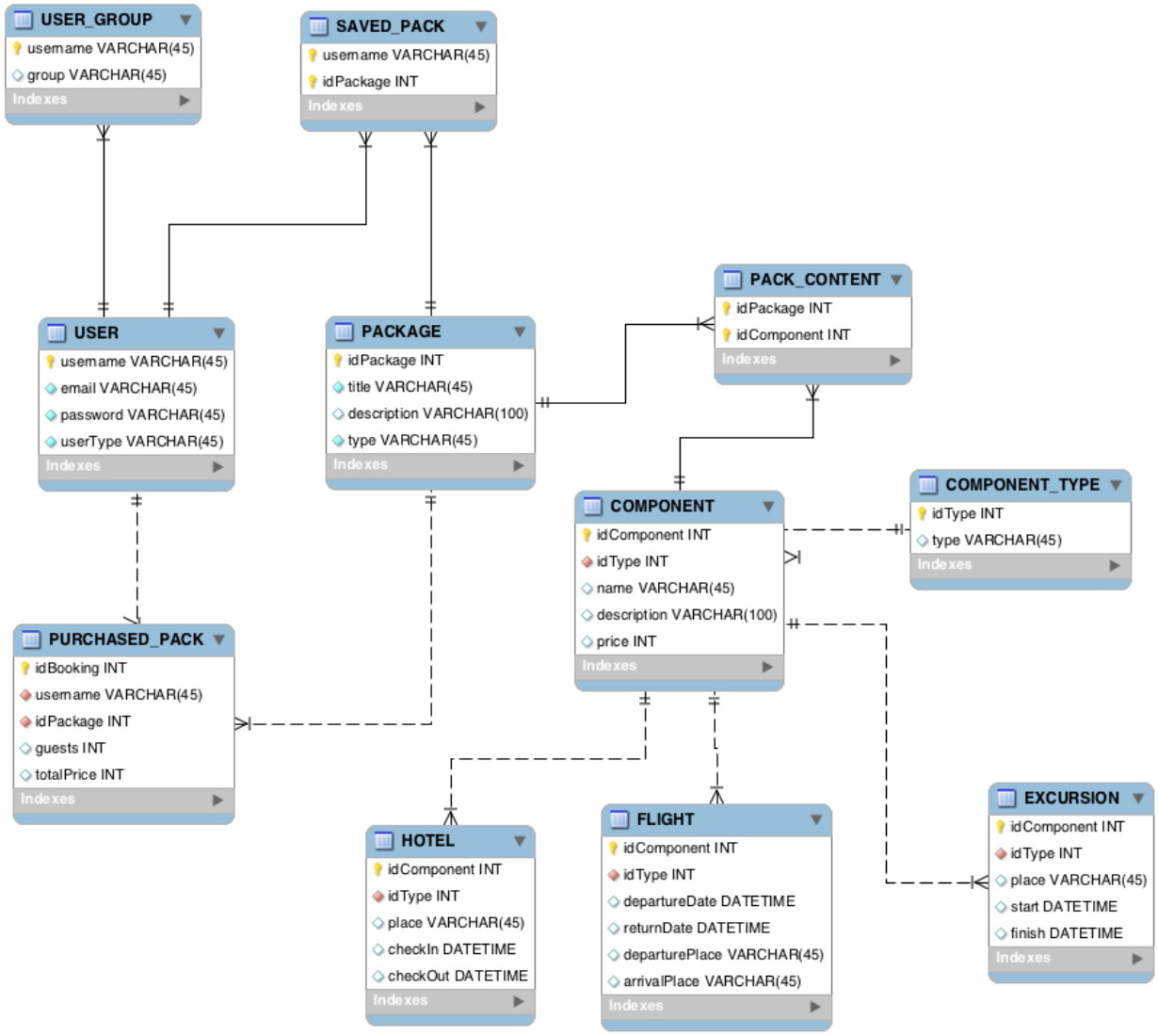
3.2 Logical Schema

Il modello entità - associazione della sezione precedente è stato utilizzato per costruire uno schema logico della base di dati. Sono state individuate delle chiavi primarie per tutte le tabelle (<primary key>), chiavi esterne per alcune (foreign key); le gerarchie e le associazioni sono state implementate o eliminate.

USER (<username>, e-mail, password)
USER GROUP (<username>, group)
PACKAGE (<ID package>, titolo, descrizione, tipo pacchetto)
SAVED PACK (<username, ID package>)
PURCHASED PACK (<ID booking>, username, ID package, partecipanti, prezzo totale)
COMPONENT (<ID component>, ID type, nome, descrizione, prezzo)
PACK CONTENT (<ID package, ID component>)
COMPONENT TYPE (<ID type>, tipo componente)
FLIGHT (<ID component>, ID type, data partenza, data ritorno)
HOTEL (<ID component>, ID type, località, check in, check out)
EXCURSION (<ID component>, ID type, località, data inizio, data fine)

Nella tabella **USER** la chiave primaria è il campo username in quanto unico e non modificabile. Il campo password verrà salvato solo dopo essere stato codificato da una funzione di hash.
La tabella **USER GROUP** viene utilizzata per conoscere a quale gruppo appartiene un utente registrato. Il campo group presenterà i valori "user" oppure "admin" a seconda del tipo di utente. Il campo username è chiave in questa tabella in modo che ogni utente possa appartenere ad un solo gruppo di utenti.
Nella tabella **PACKAGE** il campo tipo pacchetto indica se l'elemento è un pacchetto standard o un pacchetto customizzato. Anche in questo caso si è preferito utilizzare una singola tabella per i due tipi di oggetto, essendo composti allo stesso modo. L'attributo ID package è un numero intero autoincrementale che funge da chiave primaria. Nel caso in cui il pacchetto sia di tipo custom (ed in nessun altro caso), il corrispondente ID package potrà essere utilizzato per le funzioni di condivisione e ricerca tramite ID.
Nella tabella **SAVED PACK**, a parità di username, troviamo tutti gli ID dei pacchetti che quell'utente ha deciso di salvare e che verranno estratti accedendo alla sezione "I MIEI PACCHETTI".
Nella tabella **PURCHASED PACK** ogni tupla riferisce ad un acquisto di uno e un solo pacchetto viaggio. Per ogni acquisto viene salvato lo username dell'utente, il numero di partecipanti e il prezzo totale.
La tabella **COMPONENT** contiene tutti i componenti creati dagli amministratori con cui possono essere composti i pacchetti. Nel caso dei componenti si è scelto di mantenere la gerarchia individuata nel modello ER in cui le diverse tipologie volo,

hotel ed escursione discendono dall'entità padre componente.
Per implementare l'ereditarietà evitando la ripetizione di informazioni è stata creata la tabella **COMPONENT TYPE** che contiene le diverse tipologie, ciascuna associata al proprio numero identificativo (1: Volo; 2: Hotel; 3: Escursione). In futuro vi si potranno inserire nuovi tipi. La tabella component presenta per ogni singolo componente il suo tipo e un insieme di attributi comuni a tutte le tipologie (nome, descrizione, prezzo). Nella tabella di ogni singolo tipo invece risiedono gli attributi specifici (ad esempio, per i voli, data di partenza e ritorno).

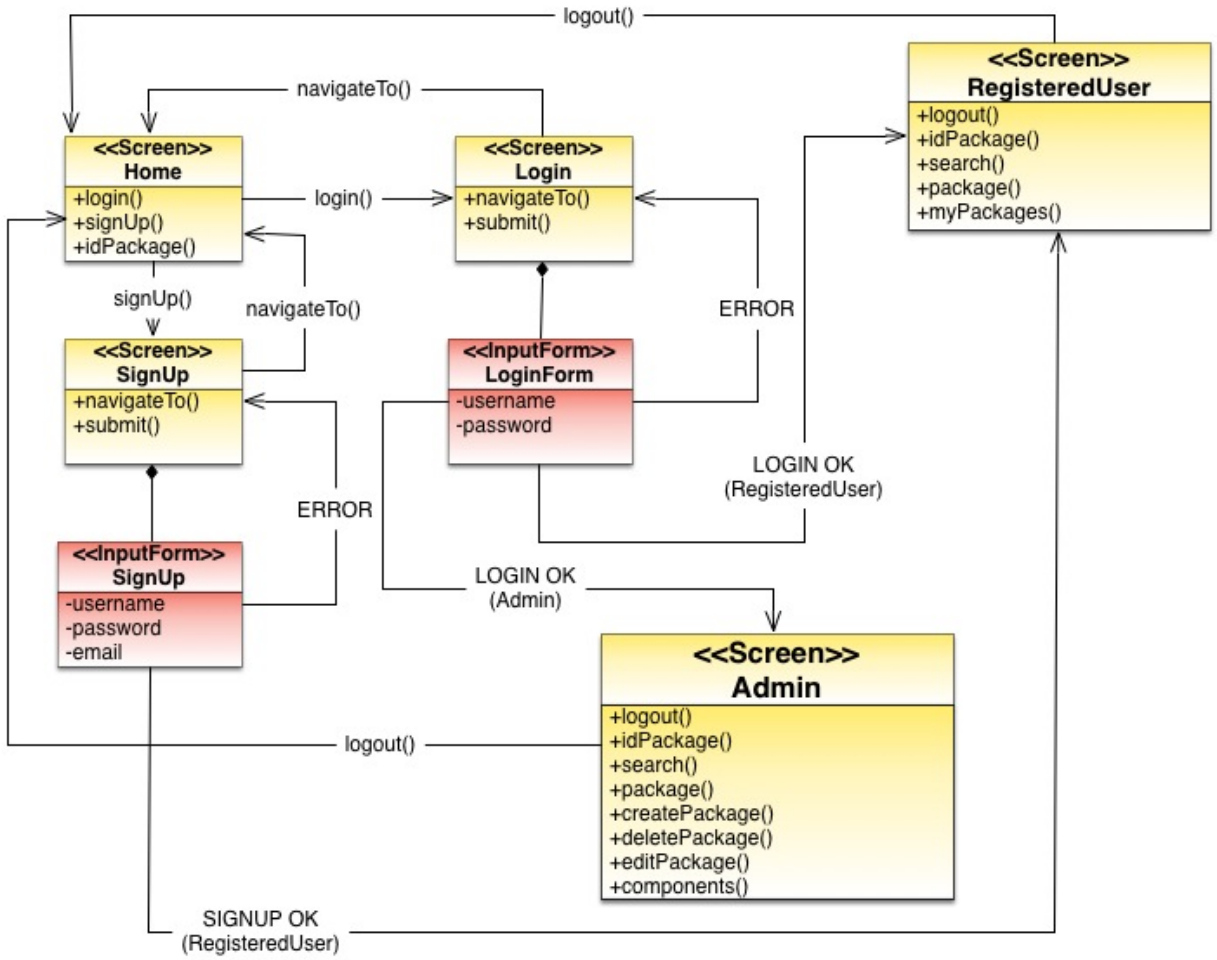


Img 2.2 Schema Logico

4. Detailed System Design

4.1 Navigation Model

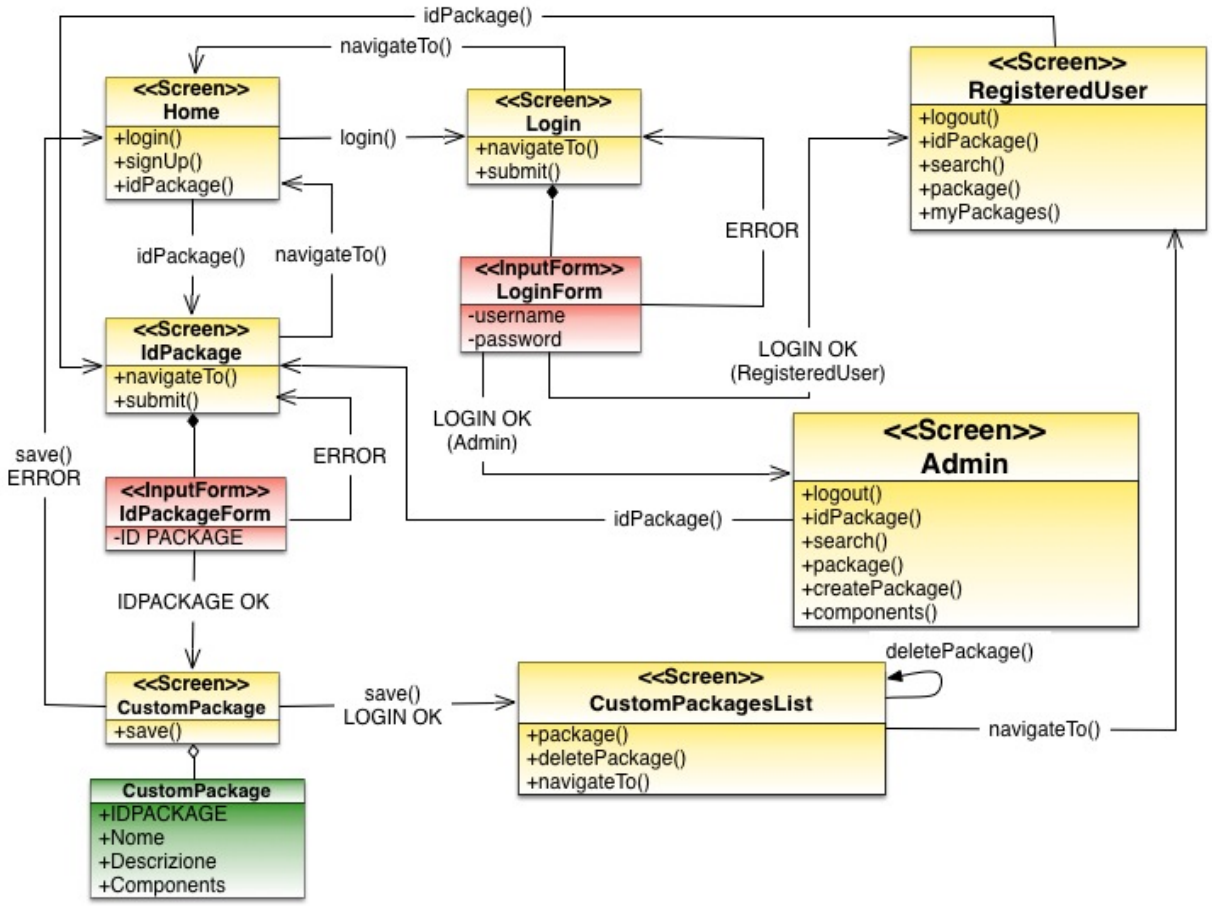
In questa sezione presentiamo il modello di navigazione ovvero la descrizione dell'interazione tra le varie pagine (<<Screen>>) dell'applicazione.
Per motivi di spazio abbiamo suddiviso il diagramma generale in più diagrammi che esplorano diverse funzionalità del sistema.
Il primo modello (Img 3.1) rappresenta l'accesso degli utenti registrati (amministratori e clienti) e la registrazione degli utenti non registrati a TD:



Img 3.1 Navigation Model: Accesso e Registrazione

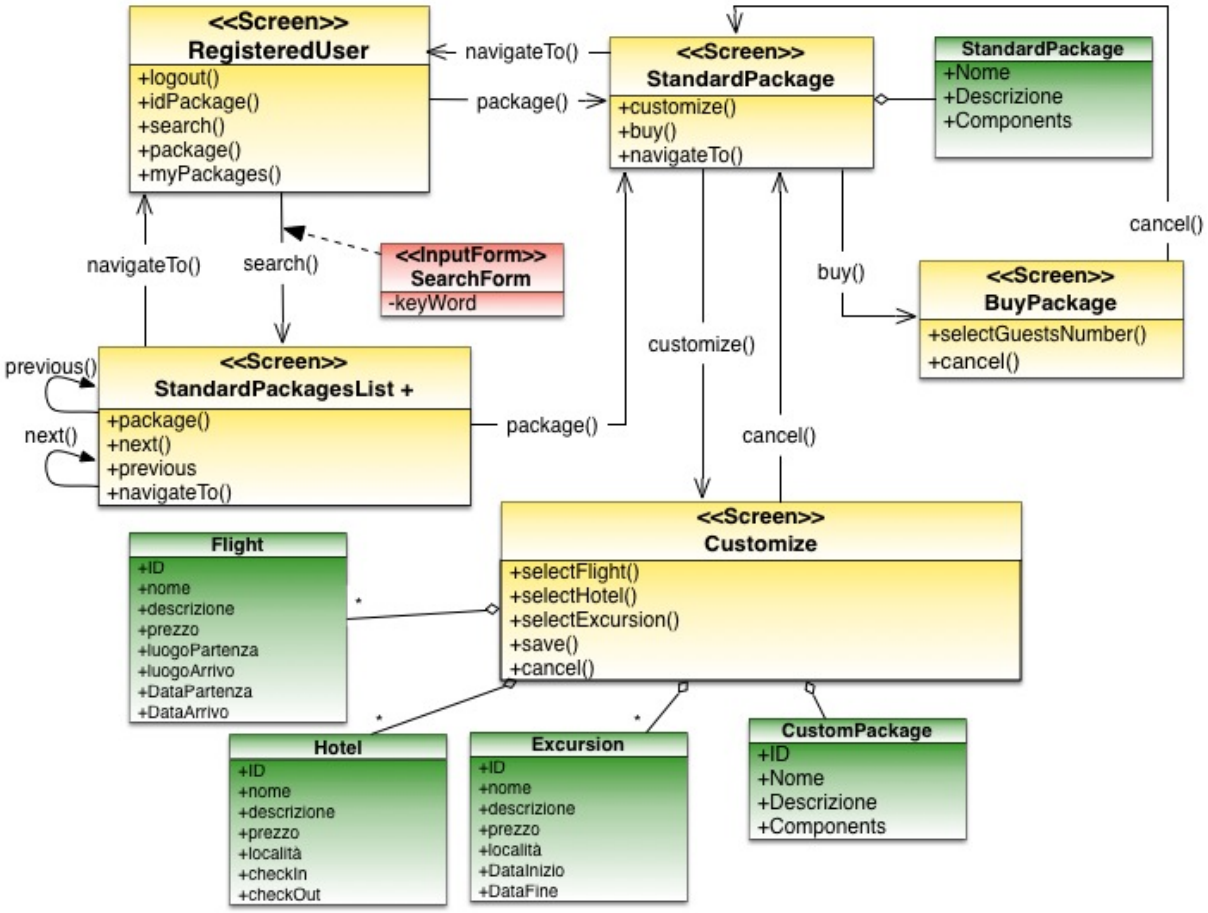
Nel seguente modello (Img 3.2) invece si descrive meglio come funziona l'opzione di condivisione del pacchetto customizzato: chiunque, se in possesso dell' ID PACKAGE relativo ad un pacchetto custom, può visualizzarne i dettagli. Se vuole

salvarlo o acquistarlo deve però essere un utente registrato. Per questo motivo nella pagina di visualizzazione dei dettagli del pacchetto custom (<<Screen>> CustomPackage) tutti sono in grado di visualizzare l'opzione di salvataggio (save()) ma possono effettuarlo solo gli utenti registrati ed autenticati nel sistema, gli altri vengono rimandati alla Home. Gli utenti registrati vengono rimandati alla sezione "I Miei Pacchetti" (<<Screen>> CustomPackagesList) in cui possono visualizzare tutti i pacchetti salvati (funzionalità spiegata meglio nell' Img 3.4).

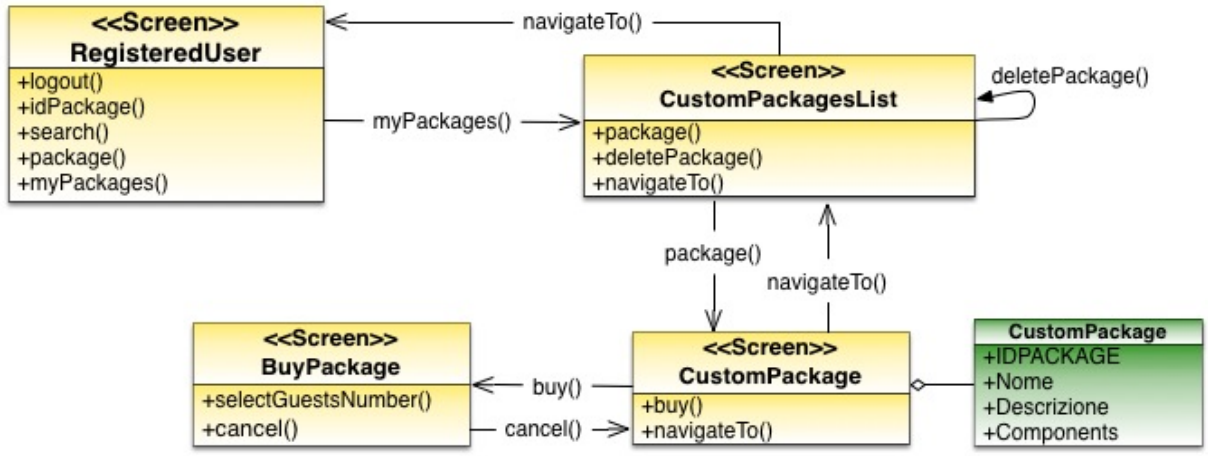


Img 3.2 Navigation Model: Condivisione Pacchetto Custom

I prossimi modelli esplorano invece le principali funzionalità dell'utente registrato: il primo (Img 3.3) riguarda la visualizzazione, ricerca e customizzazione dei pacchetti viaggio standard; il secondo (Img 3.4) invece descrive le pagine relative ai pacchetti custom salvati nella sezione I Miei Pacchetti.
La pagina di acquisto dei pacchetti per ora non è stata implementata nel dettaglio.

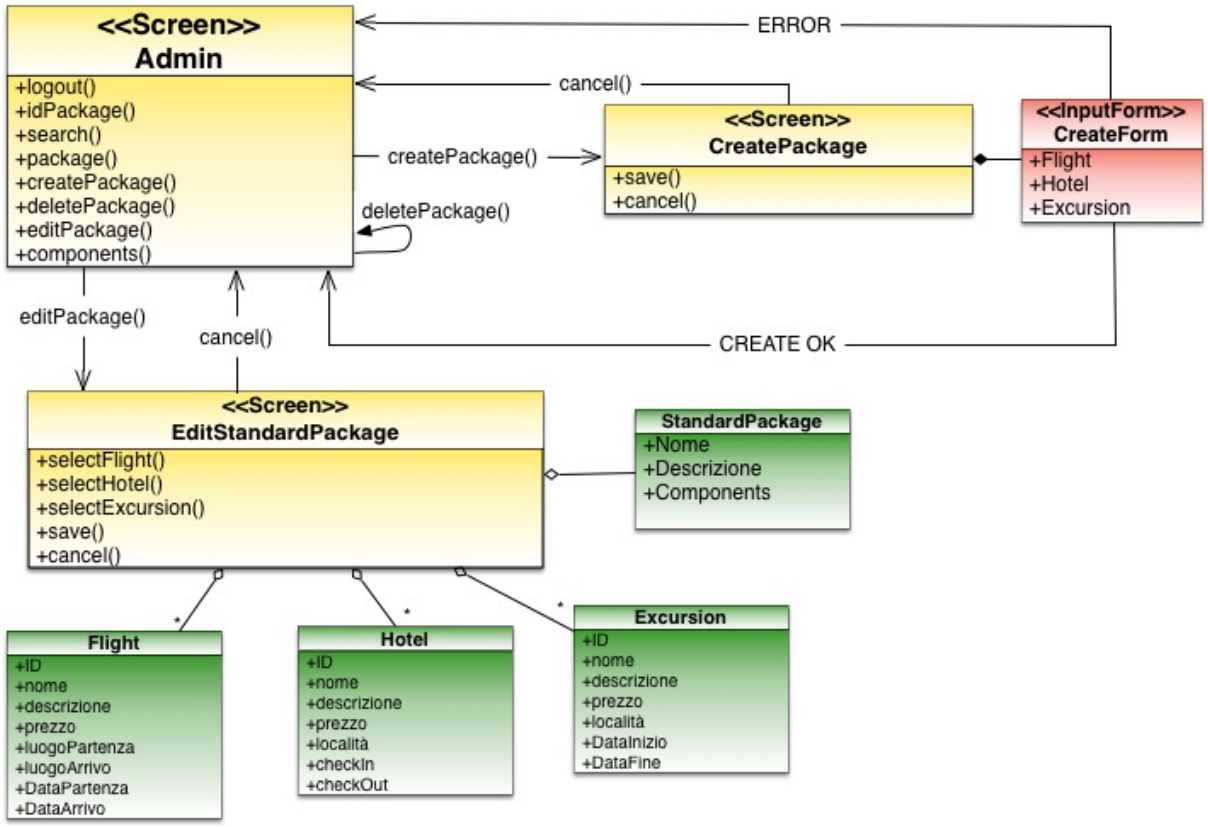


Img 3.3 Navigation Model: Utente Registrato e Pacchetti Standard

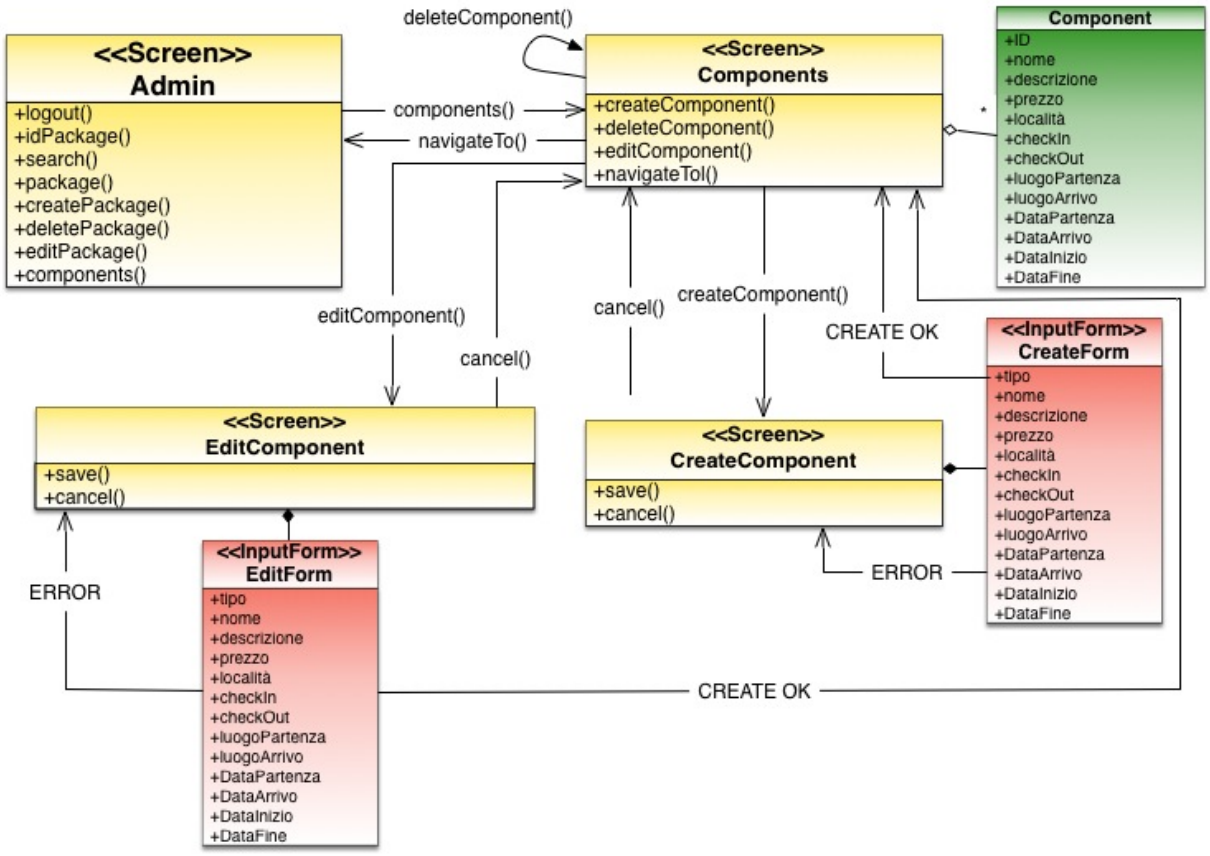


Img 3.4 Navigation Model: Utente Registrato e Pacchetti Custom

Infine i seguenti modelli riguardano i servizi dedicati agli amministratori TD: il primo (Img 3.5) riguarda la gestione dei pacchetti standard, ll secondo (Img 3.6) riguarda la gestione dei componenti.



Img 3.5 Navigation Model: Admin e Pacchetti Standard

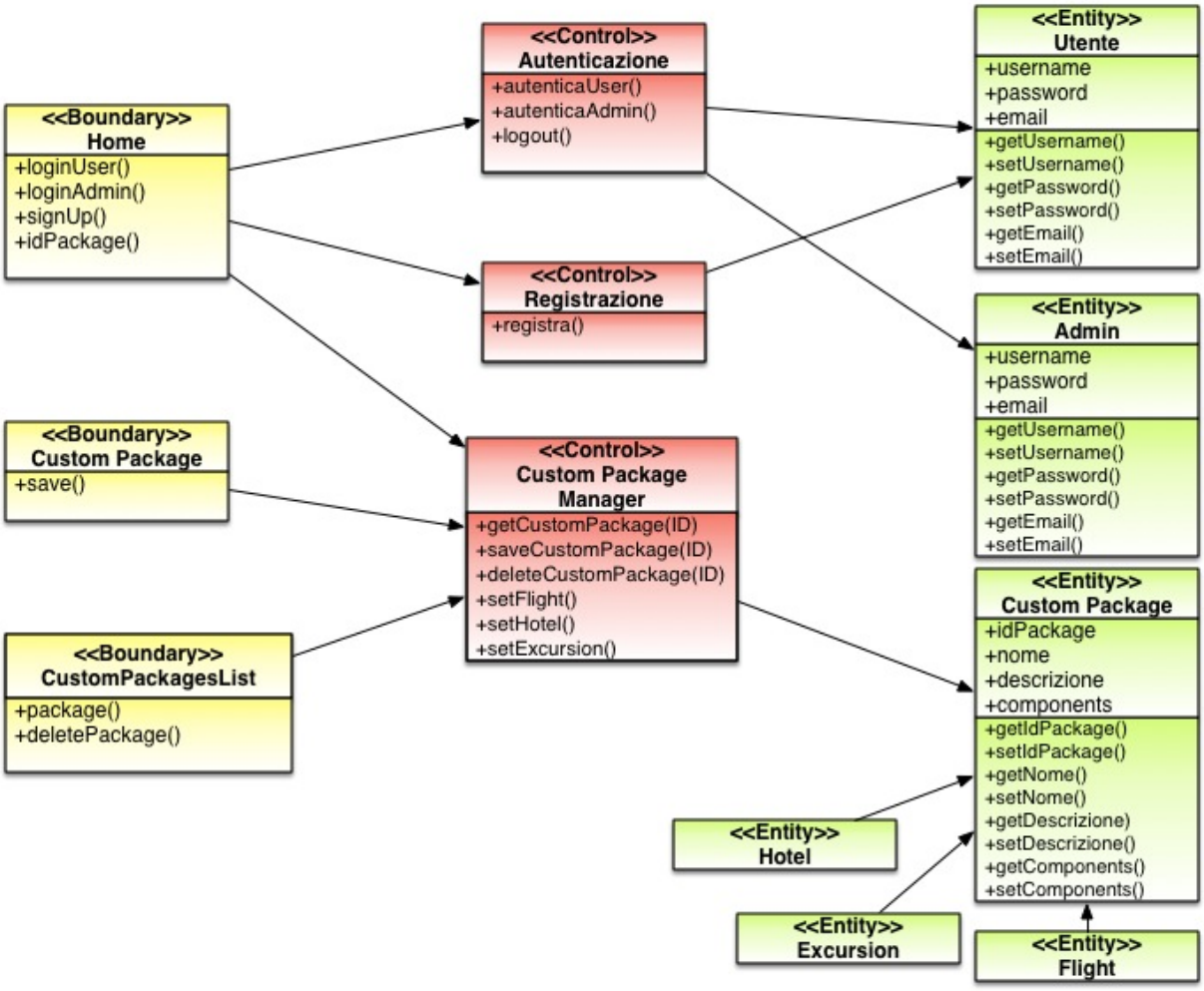


Img 3.6 Navigation Model: Admin e Gestione Componenti

4.2 Components Design

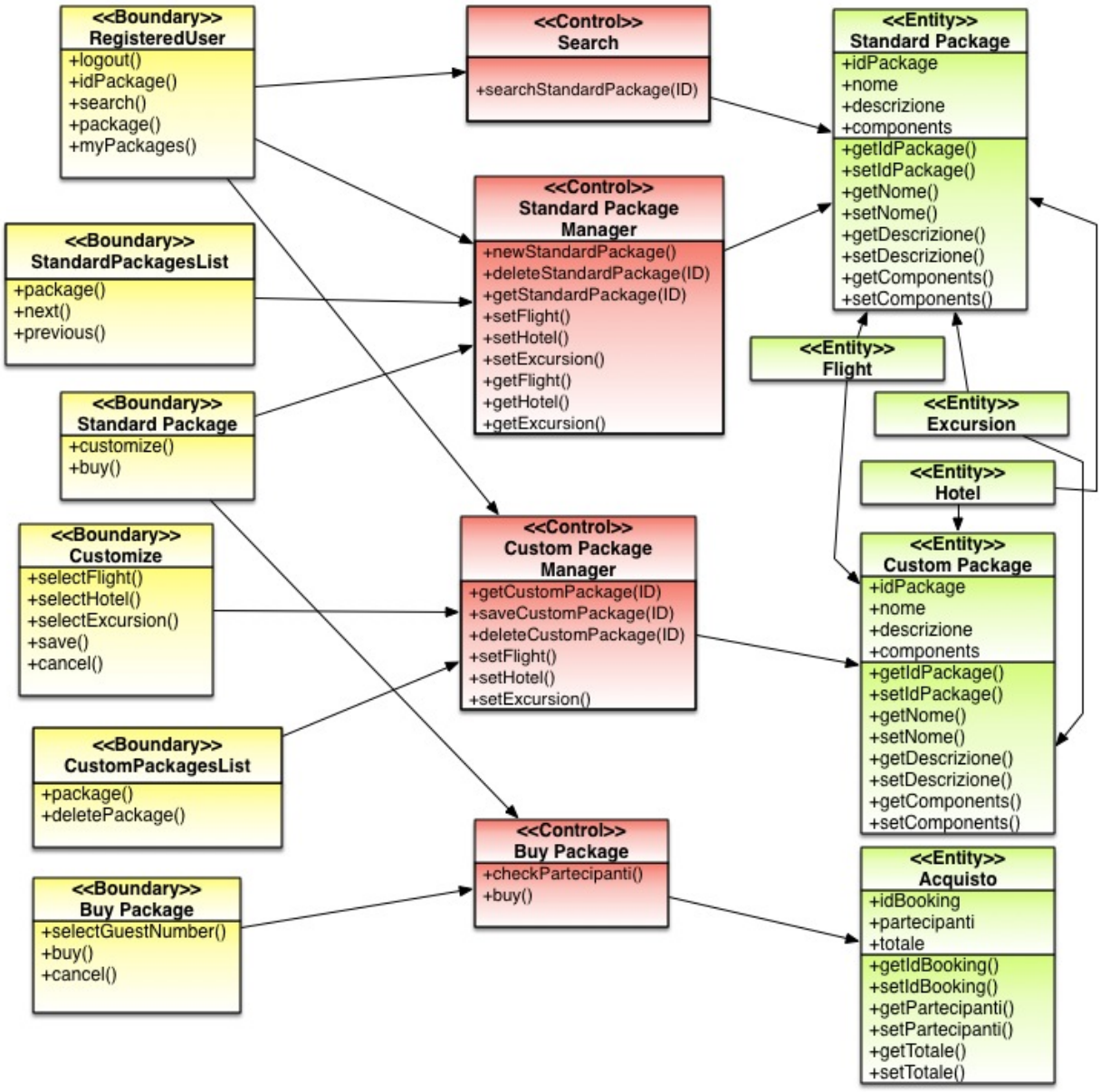
4.2.1 ECB Diagrams

Per il design dei componenti ci siamo basati sul pattern architetturale Model-View-Controller in modo da garantire che la presentazione dei dati sia indipendente dalla logica dell'applicazione e permettere una più semplice interazione con il sistema da parte dell'utente.
Di seguito chiariremo i dettagli di tale suddivisione utilizzando la classica semplificazione UML: il pattern ECB (Entity-Control-Boundary).
Gli oggetti di tipo <<Boundary>> sono quelli che si interfacciano con gli attori del sistema, ovvero gli <<Screen>> che abbiamo individuato nei modelli di navigazione del paragrafo 4.1; essi possono interagire solo con oggetti di tipo <<Control>>.
Le <<Entity>> invece rappresentano i dati del sistema, ovvero le entità mostrate nel diagramma ER del paragrafo 3.1; esse possono interagire o con altre <<Entity>> o con oggetti di tipo <<Control>>.
Infine i <<Control>> sono i cosiddetti controller che permettono l'interazione tra i due tipi di oggetti precedentemente descritti; possono dunque interagire con entrambi ma rappresentano la logica dell'applicazione dunque non devono e non possono essere raggiungibili dagli attori del sistema.
Come per il modello di navigazione per motivi di spazio abbiamo suddiviso il diagramma generale in più diagrammi che esplorano diverse funzionalità del sistema.
Qui di seguito (Img 4.1) vengono descritti i servizi messi a disposizione subito nella pagina iniziale dell'applicazione: registrazione, autenticazione dell'utente o dell' amministratore TD e visualizzazione dei dettagli di un pacchetto personalizzato attraverso l'inserimento di un ID PACKAGE valido.



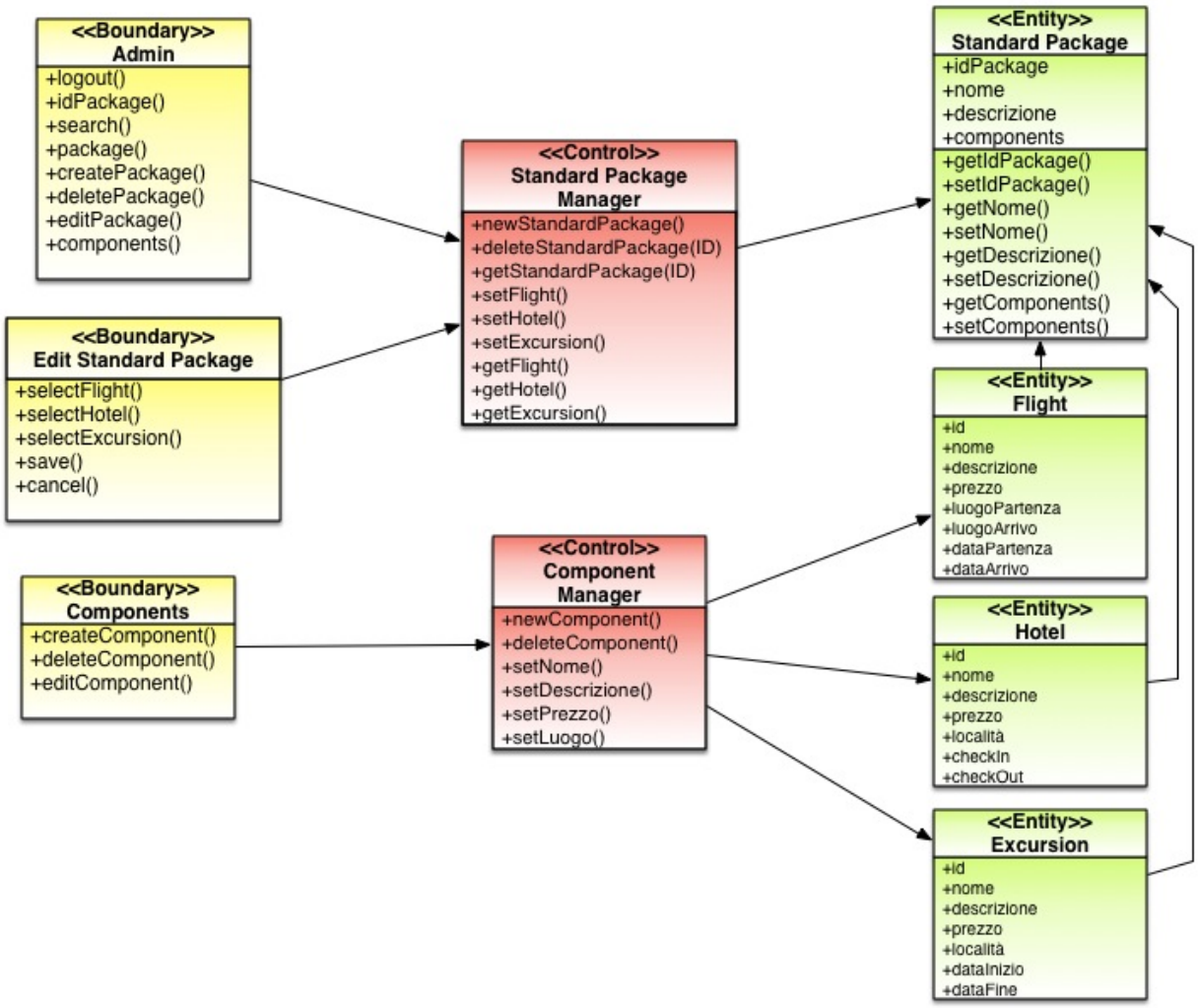
Img 4.1 ECB Diagram: Accesso, Registrazione e Condivisione Pacchetto Custom
(Le entità Volo, Hotel ed Escursione non sono state descritte nei dettagli)

Il prossimo diagramma (Img 4.2) invece rappresenta le funzionalità di cui può usufruire l'utente registrato dopo essersi autenticato: visualizzazione, ricerca e customizzazione dei pacchetti viaggio standard così come la gestione dei pacchetti custom da lui salvati.



Img 4.2 ECB Diagram: Funzionalità Utente Registrato
(Le entità Volo, Hotel ed Escursione non sono state descritte nei dettagli)

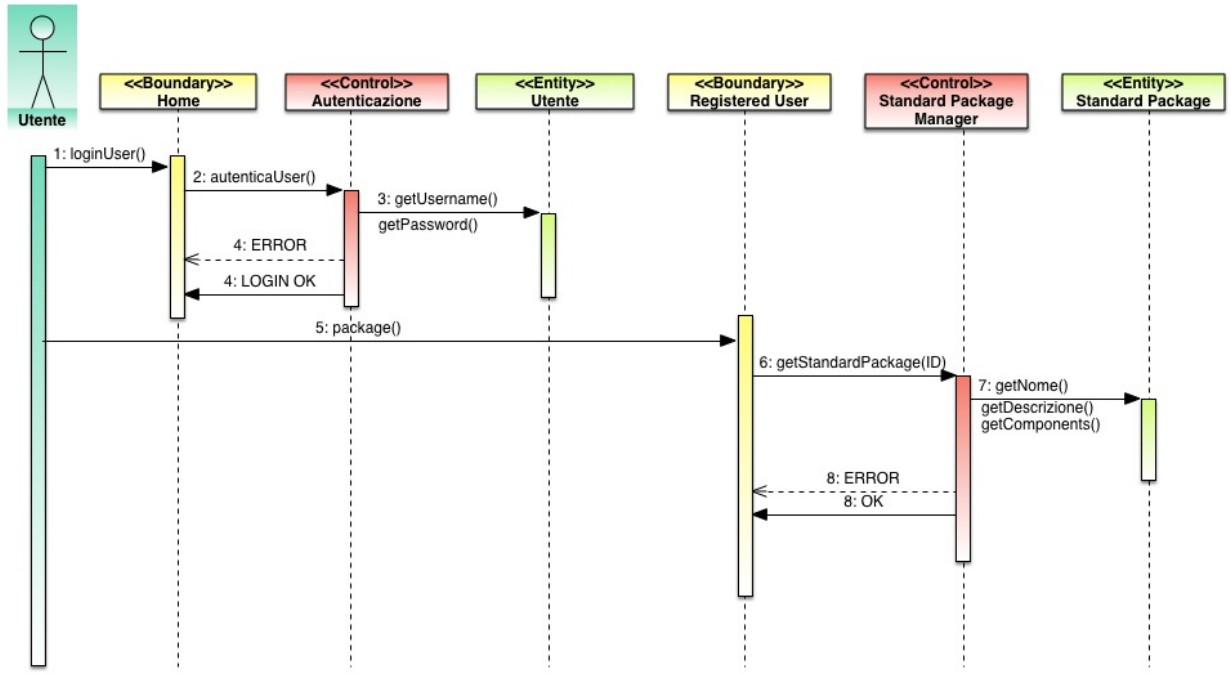
Infine l'ultimo diagramma (Img 4.3) descrive i principali servizi a cui può accedere l'amministratore TD dopo l'autenticazione al sistema: gestione dei pacchetti standard e dei vari componenti.



Img 4.3 ECB Diagram: Funzionalità Admin TD
(Le entità Volo, Hotel ed Escursione non sono state descritte nei dettagli)

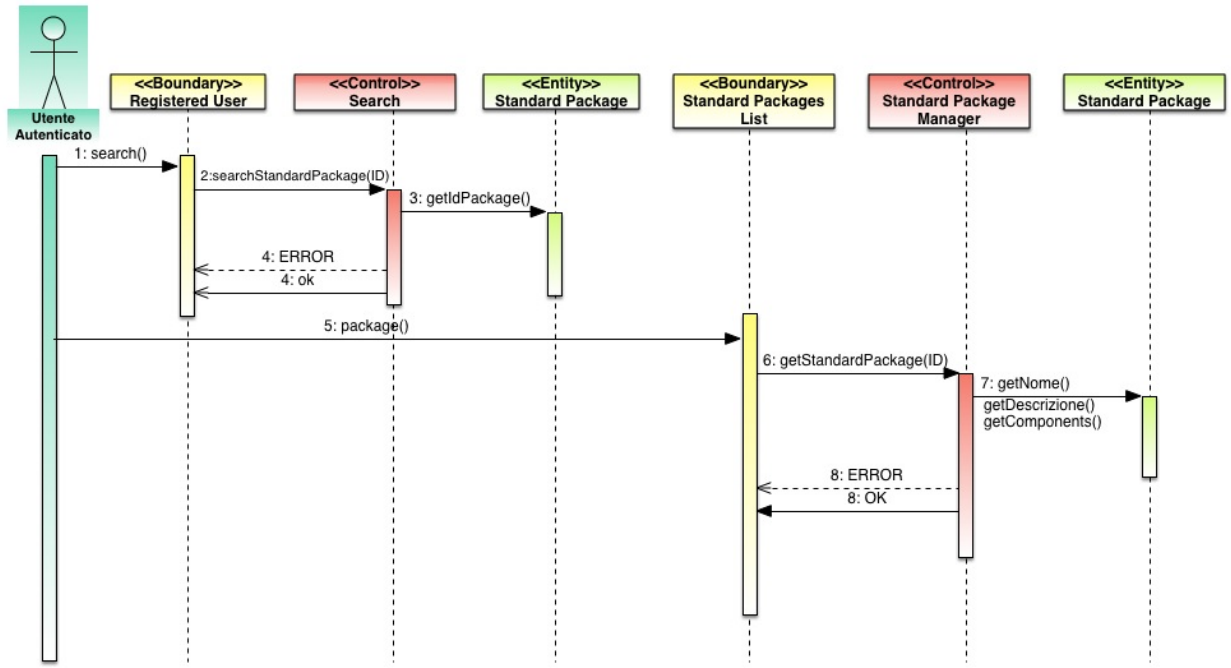
4.2.2 Sequence Diagrams

Per comprendere meglio come potrà essere seguito il pattern MCV in fase di implementazione presentiamo di seguito dei diagrammi di sequenza che descrivono come avviene l'interazione tra le <<Entity>>, i <<Control>> e i <<Boundary>> durante l'esecuzione delle principali funzionalità del sistema.
Il primo (Img 5.1) riguarda l'autenticazione di un utente registrato e la seguente visualizzazione di un pacchetto viaggio standard.



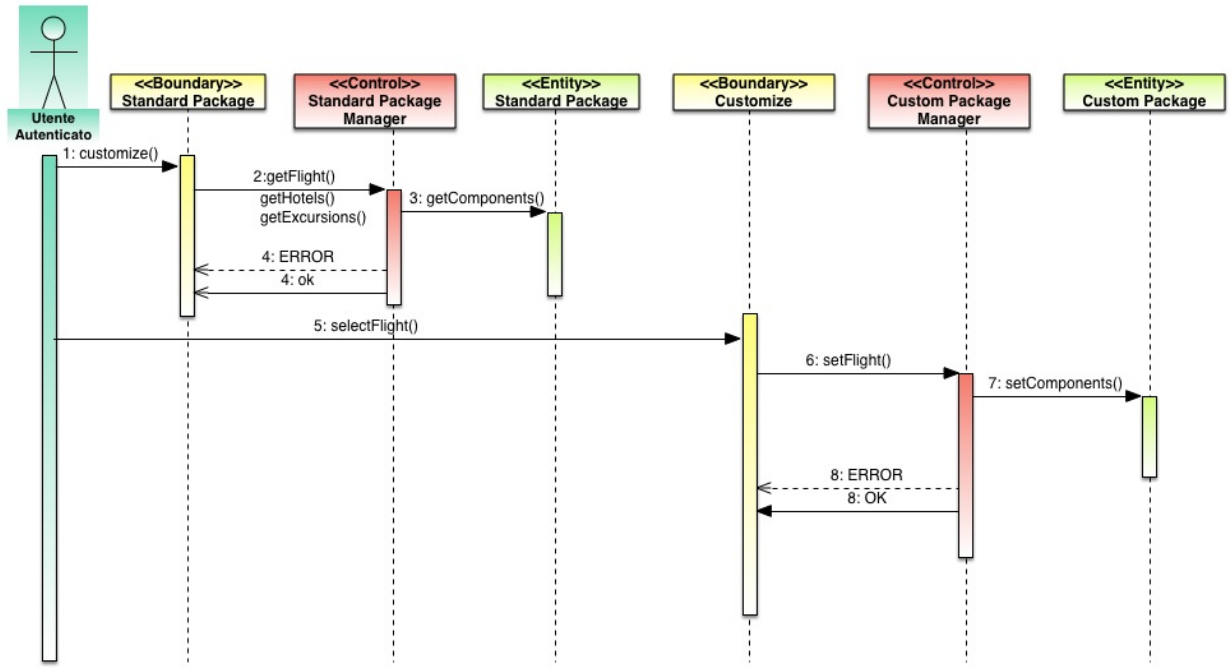
Img 5.1 Sequence Diagram: Accesso Utente e Visualizzazione Pacchetto Standard

Il prossimo diagramma (Img 5.2) invece riguarda la funzionalità dell'utente registrato di poter ricercare un pacchetto standard digitando una parola chiave nella form messa a disposizione e visualizzarne i dettagli.



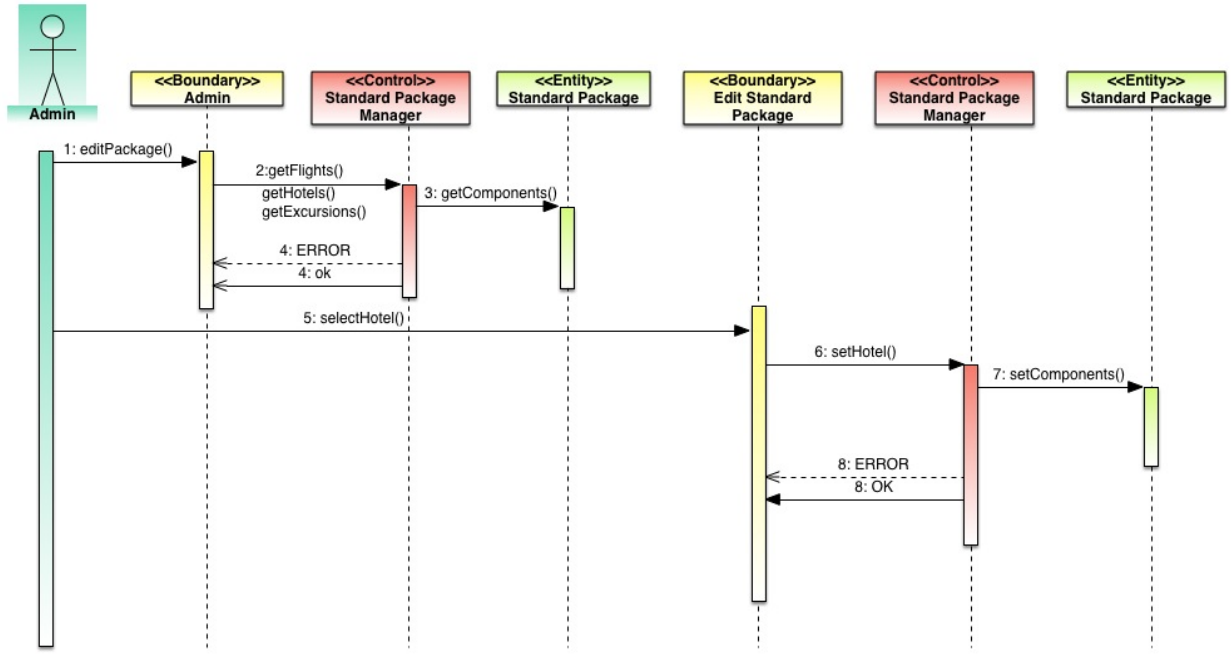
Img 5.2 Sequence Diagram: Ricerca e Visualizzazione Pacchetto Standard

Un'ultima funzionalità relativa all'utente registrato che esploreremo attraverso i sequence diagram è la customizzazione di un pacchetto standard, ad esempio selezionando solo il volo (Img 5.3).



Img 5.3 Sequence Diagram: Customizzazione Pacchetto Standard

Infine nell'ultimo diagramma (Img 5.4) abbiamo descritto la sessione che riguarda l'amministratore TD e la gestione di un pacchetto standard; in particolare illustriamo la modifica di uno dei suoi componenti (nell'esempio l'hotel).



Img 5.4 Sequence Diagram: Admin e Modifica Pacchetto Standard