Politecnico di Milano

Progetto di Ingegneria del Software 2

DD - Design Document

Autori:
Claudia Foglieni
Giovanni Matteo Fumarola
Massimo Maggi

Professori:
Elisabetta Di Nitto
Raffaela Mirandola

1 gennaio 2012

Indice

1 Panoramica del sistema

Il sistema è stato creato per servire un servizio a due tipologie di utenti principali: studenti e professori. Per rendere facile l'accesso anche a persone poco esperte si è deciso di sviluppare una piattaforma web, per fornire un'interfaccia semplice e pulita. In questo modo ogni utente avrà accesso alle sue funzioni in modo rapido ed efficace.

Inoltre per salvaguardare i dati inseriti si è deciso di implementare un sistema di sicurezza, controllando chi può avere accesso all'applicazione e le operazioni consentite. Solo utenti registrati (e nel caso di un professore anche abilitati) posso effettuare il login con le loro credenziali. La lista degli utenti registrati è inclusa, insieme ad altre informazioni personali, nel database della piattaforma. Come implimentazione si è utilizzato unmodulo JAAS (Java Authentication ed Authorization Service), usando quindi il sistema di sicurezza integrato in J2EE. Oltre a ciò per impedire accessi a funzioni non abilitate per utenti registrati, si è deciso di utilizzare il Role-Based Access Control. E' previsto quindi che l'utilizzo ai Session Bean e ai singoli metodi deve essere consentito solo agli utenti appartenenti a determinati ruoli, specificati con annotazioni nel codice. In base al tipo di profilo registrato sarà possibile usufruire solo delle opzioni valide per il proprio ruolo.

2 Rettifiche al RASD

Nel corso della stesura del documenti di Design si è deciso di approfondire alcuni punti specifici:

- **calcolo della penalità**, per attribuire la penalità agli elaborati consegnati si condera come data quella relativa alla prima consegna. Se gli studenti invieranno una successiva versione dello stesso documento, la prima data di consegna sarà quella utilizzata per stabilire se è necessario attribuire una penalità o meno.
- **valutazione di un elaborato**, ogni professore potrà visionare tutte le versioni di un documento invitae per una data consegna, ma potrà valutare solo l'ultimo elaborato inviato.

3 Descrizione dell'architettura

Per sviluppare la piattaforma sono stati utilizzati vari strumenti:

- **WaveMaker**, per creare l'interfaccia grafica;
- **Eclipse Indigo**, per creare la logica di business;
- **Mysql Server**, per gestire la base di dati.

Come struttura generale il progetto è stato creato come applicazione di WaveMaker.
In questo modo il progetto finale risulta composto da varia pagine html, ognuna delle quali fornisce un servizio o delle informazioni utili all'utente. In ogni pagina è stato fatto uso di JavaScript, per poter recuperare e mostrare i dati all'utente. Tramite Javascript è possibile richiamare dei Java Service, componenti particolari utilizzati da WaveMaker.
Un Java Service è un termine utilizzato in WaveMaker per indicare una classe java che

implementa metodi che forniscono dati ai widget lato client oppure effettuano operazioni per conto di essi.
Tramite i Java Service è possibile inviare richieste al business tier, che gestisce i dati e le operazioni necessarie per fornire le informazioni richieste dall'utente. Quando necessario, il server accede al database per recuperare i dati salvati o per inserirne di nuovi.

La piattaforma risulta quindi un'applicazione multi-tiered, composta da quattro livelli:

- **client tier**, è composta dall'applicazione lato client, gestita dai singoli utenti e quindi utilizzata su computer diversi. Questo livello invia richieste al server tramite un browser web. Le pagine html fornite includono javascript per gestire tutti i componenti e il passaggio tra le varie schermate.

- **web tier**, include i componenti che gestiscono l'interazione tra client e business tier. E' costituito da servlet, che rispondono a richieste Ajax, generando dinamicamente i dati necessari al client. Inoltre raccolgono gli input forniti dagli utenti, inviando i messaggi ai componenti appropriati nel business tier. Le servlet servono anche a mantenere le sessioni dati degli utenti.

- **business tier**, è costituito dai Entreprise JavaBeans che gestiscono la logica del sistema. Si è deciso di implementare per ogni funzionalità un session beans, suddividendo poi al suo interno quali operazioni possono essere invocate dai vari tipi di utente. Inoltre al suo interno contiene gli Entity Beans, ricavati dall'analisi del nostro database.

- **data tier**, include il database, accessibile dai componenti presenti nel business tier.

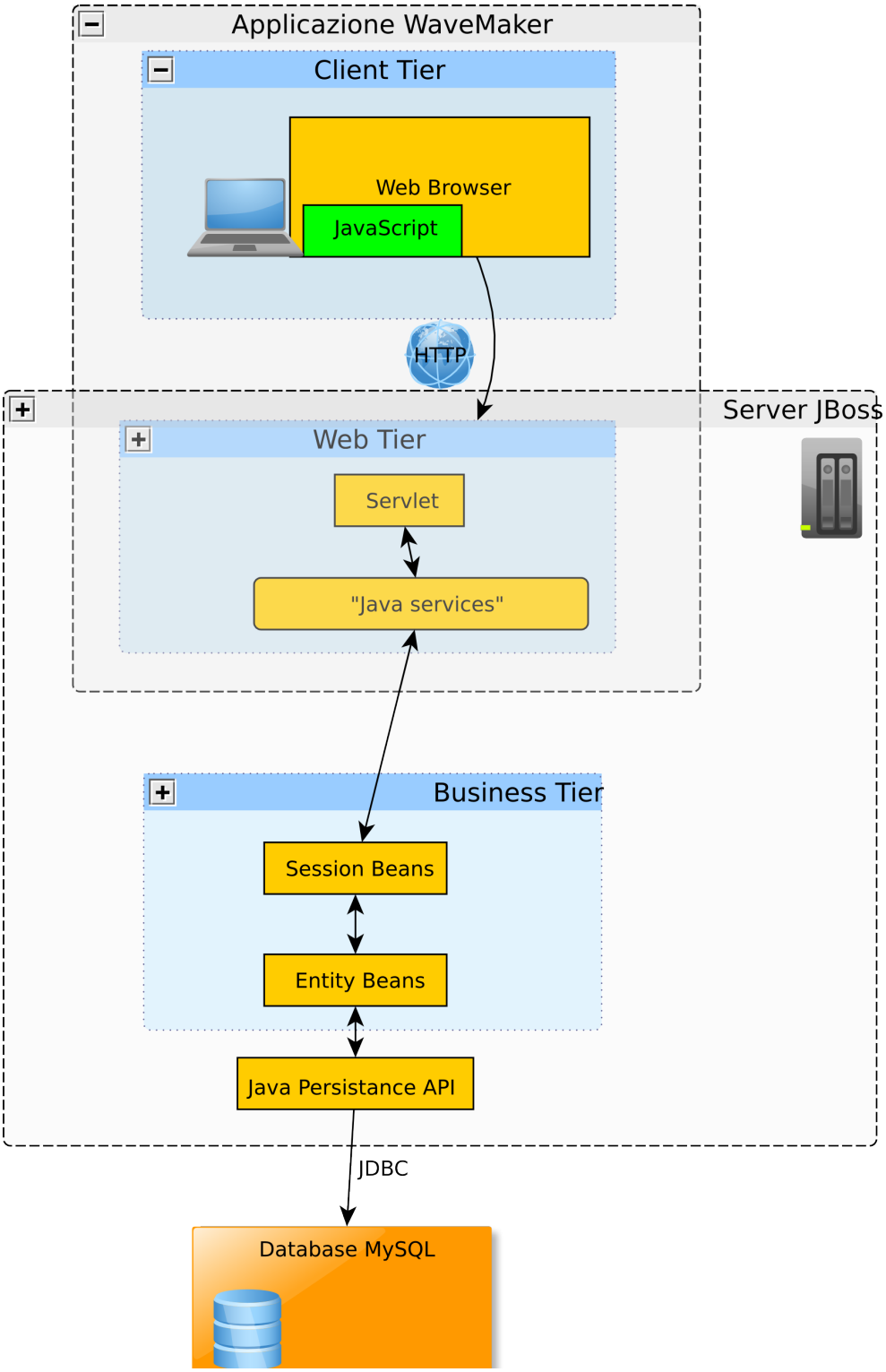

Figura 1: Architettura e schema di deployment del sistema

3.1 Decomposizione in Sottosistemi

Per la realizzazione del sistema sono necessari i seguenti sottosistemi:

- login;
- archivio informazioni;
- tre diverse funzionalità, ognuna per un determinato ruolo utilizzabile nel sistema.

La parte software è direttamente accessibile da qualsiasi tipo di utente perché consisterà nella semplice navigazione utilizzando un web browser.
Analizzando i sottosistemi individuati per la realizzazione del sistema si ottiene:

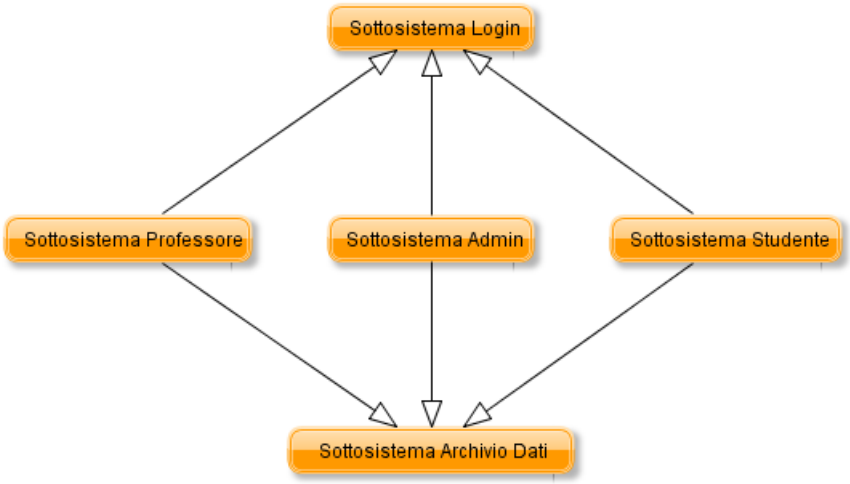

Figura 2: Decomposizione del sistema in sottosistemi.

- **Login**: sottosistema che si occupa della gestione dell'accesso al sistema, in base alla tipologia dell'utente seleziona il sottosistema di utilizzo;
- **Archivio**: sottosistema che si occupa della gestione degli accessi e degli utilizzi del DBMS del sistema;
- **Sottosistema Studente**: si occupa delle funzionalità dello studente, tra le quali operazioni troviamo l'iscrizione personale e del gruppo di lavoro al sistema, la parte esecutiva della gestione progetti dalla scelta alla consegna finale;
- **Sottosistema Professore**: si occupa delle funzionalità del professore, tra le quali troviamo la gestione dei progetti, dalla presentazione alla valutazione del lavoro finale;
- **Sottosistema Admin**: si occupa delle funzionalità dell'amministratore del sistema, tra le quali troviamo la gestione degli utenti.

4 Gestione dei Dati Persistenti

Per la gestione dei dati persistenti è stata realizzata una base di dati relazionale protetta da password. Come tutti i sistemi informativi la protezione da password è essenziale per la sicurezza e la privacy dei dati memorizzati.

4.1 Progettazione concettuale

la progettazione concettuale serve a fornire una rappresentazione semplificata della realtà per rappresentarla nel sistema. I dati vanno individuati in maniera tale da rispondere allo scopo per il quale il modello viene creato e per le successive fasi di design. Serviranno per il recupero di informazioni fondamentali alla realizzazione.

Come prima parte della progettazione concettuale è necessario analizzare i dati necessari per l'identificazione dell'utente nel sistema, alcuni di questi dati saranno necessari per la fase di accesso.
La seconda parte consiste nella analisi dei dati fondamentali per l'utilizzo vero e proprio del sistema.

4.1.1 Entità

Per la gestione dell'utenza è possibile individuare:

- **Persona**: é l'entità padre, generalizza qualsiasi tipo di utente con un unico formato. Le sue sotto-entità sono Studente, Professore e Admin. Le diverse categorie di utenza differiscono al massimo di 2 campi. In persona individuiamo:

  1. dei campi standard per un utente (indifferente che sia uno studente o uno professore), i quali sono: Nome, Cognome, Email;
  2. dei campi essenziali per l'accesso al sistema, i quali sono Username e Password;
  3. per rendere il sistema più innovativo, a ogni utente è associato una immagine di profilo.

  Analizzando ogni entità figlia di persona si ottiene:

  1. **Admin**: Possiede tutti i campi di persona, utilizzata per l'utente che è amministratore del sistema;
  2. **Studente**: Possiede tutti i campi di persona con l'aggiunta di 2 informazioni indispensabili, la propria matricola (necessaria nel mondo accademico) e il proprio di team di lavoro. Utilizzata per l'utente che è uno studente universitario;
  3. **Professore**: Possiede tutti i campi di persona, utilizzata per l'utente che è un docente. Inoltre ogni professore presenta un campo enabled, che permette di stabilire se l'utente è già stato abilitato dall'amministratore di sistema.

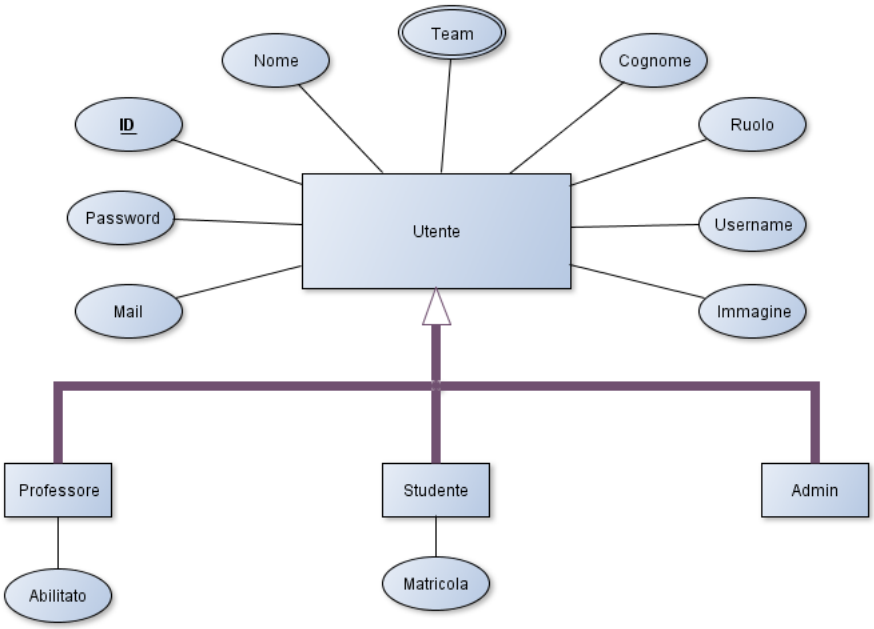

Figura 3: Attributi tabella utente.

Per la gestione dei progetti è possibile individuare:

- **Team**: Ha solo 1 campo informazione, che corrisponde al nome del gruppo.
- **Progetto**: Ha 2 campi informazione, il titolo del progetto e una descrizione generale.
- **Richiesta**: Un progetto è composto da varie richieste (in alcuni casi anche una sola). Ha i seguenti campi informazione: Nome, Giorno di creazione, Giorno di Deadline, una descrizione, valore di penalità per giorno di ritardo e il relativo peso all'interno del progetto.
- **Elaborato**: Ha alcuni campi informazione tra i quali troviamo: Il nome del file per la richiesta, la penalità ricevuto in base al potenziale ritardo, la valutazione del docente, il giorno e l'orario di consegna. Questa entità ha un campo informazione che può essere confuso per la propria tipologia. Si tratta del file vero e proprio consegnato, anche questo va nell'archivio dati.

In tutte le entità c'è l'aggiunta di un campo univoco definito ID, fondamentale per l'utilizzo in modo proprio di un database relazionale.

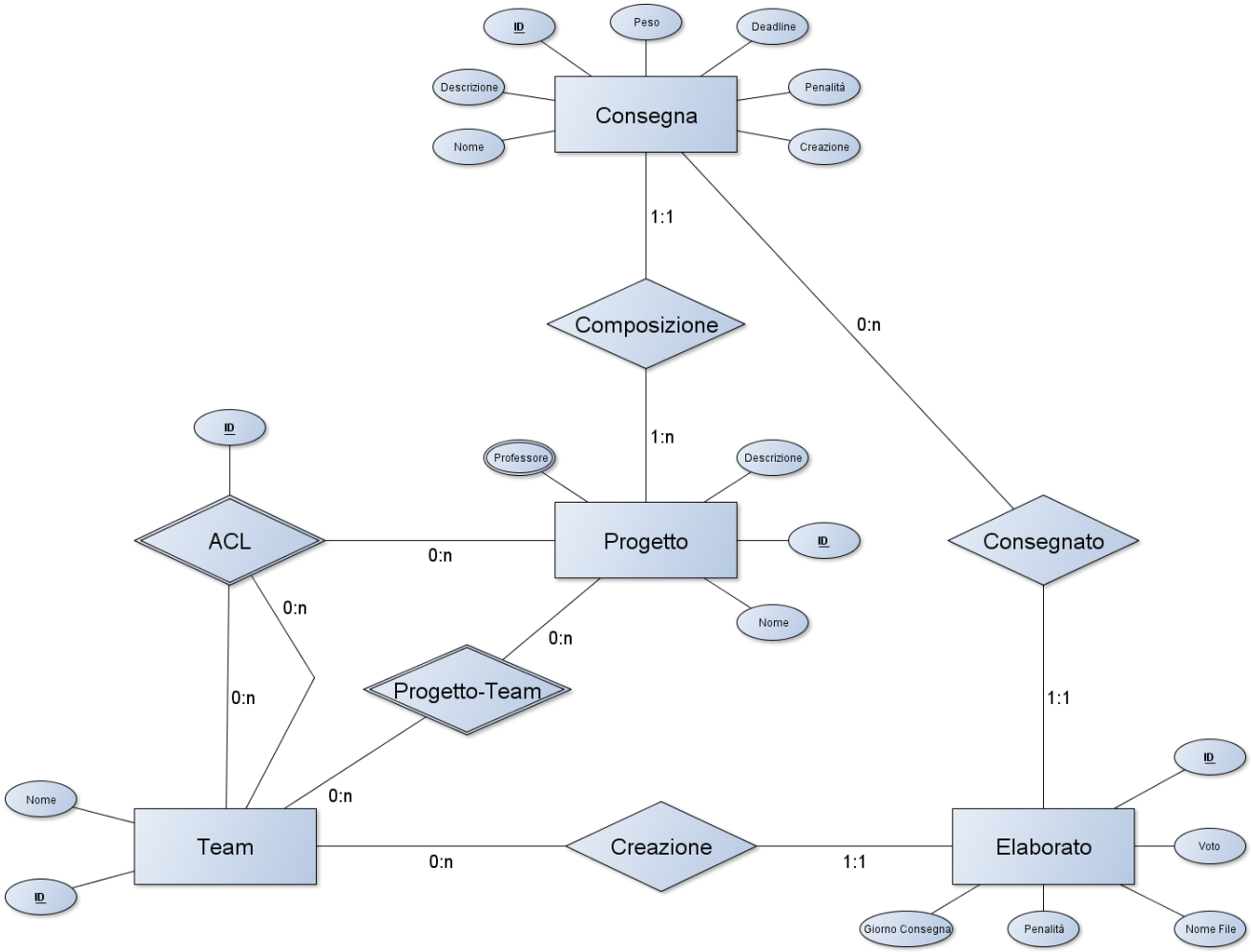

Figura 4: Attributi tabelle nel database.

4.1.2 Associazioni

- **Consegna**: Lega l'entità Elaborato con Richiesta. Non possiede attributi propri.
  Dal lato elaborato abbiamo una cardinalità 1 a 1, Ogni elaborato corrisponde a una sola Richiesta nella realizzazione del progetto.
  Dal lato richiesta abbiamo una cardinalità 0 a n, una richiesta può avere nessun elaborato ancora consegnato.

- **Creazione**: lega l'entità Elaborato con l'entità team. Non possiede attributi propri.
  Dal lato elaborato abbiamo una cardinalità 1 a 1, ogni elaborato è stato realizzato da un unico gruppo di lavoro.
  Dal lato Team abbiamo una cardinalità 0 a n, un gruppo può non aver consegnato ancora nessun elaborato.

- **Composizione**: lega l'entità Progetto con Richiesta, non possiede attributi propri.
  Dal lato progetto abbiamo una cardinalità 1 a n, ogni progetto è composto da almeno una richiesta.
  Dal lato richiesta abbiamo una cardinalità 1 a 1, ogni richiesta serve per la realizzazione (anche totale) di un progetto.

- **Presentazione**: lega l'entità Professore con progetto, non possiede attributi propri.
  Dal lato professore abbiamo una cardinalità 0 a n, un professore può non aver presentato nessun progetto.
  Dal lato progetto abbiamo una cardinalità 1 a 1, ogni progetto è presentato da un unico professore. Nel caso reale di un progetto proposto da più professori, il sistema salverà nell'archivio il professore che ha creato il progetto sul sistema.

- **Progetto-Team**: come afferma il nome lega l'entità progetto al team, non possiede attributi propri.
  Dal lato progetto abbiamo una cardinalità 0 a n, un progetto può non essere ancora svolto da nessuno.
  Dal lato team abbiamo una cardinalità 0 a n, un gruppo di lavoro può non aver selezionato nessun progetto da svolgere.

- **Invito**: lega l'entità team con studente, non possiede attributi propri.
  Dal lato team abbiamo una cardinalità 0 a 2, uno studente del team può invitare nessuno studente come nel caso massimo 2.
  Dal lato studente abbiamo una cardinalità 0 a n, uno studente non può essere invitato da nessun gruppo.

- **Appartenenza**: lega l'entità team con studente, non possiede attributi propri.
  Dal lato team abbiamo una cardinalità 1 a 3, un gruppo è composto da al massimo 3 studenti.
  Dal lato studente abbiamo una cardinalità 0 a 1, uno studente può non far parte di nessun gruppo.

- **ACL**: lega l'entità progetto con team, non possiede attributi propri.
  È una associazione atipica, perché connette 2 diversi team (ci sono 2 frecce distinte partenti/arrivanti a Team).
  Dal lato progetto abbiamo una cardinalità 0 a n, un professore può non aver impostato nessuna condivisione tra progetti.
  Dal lato team abbiamo una cardinalità 0 a n, un gruppo può non aver nessuna condivisione attiva.

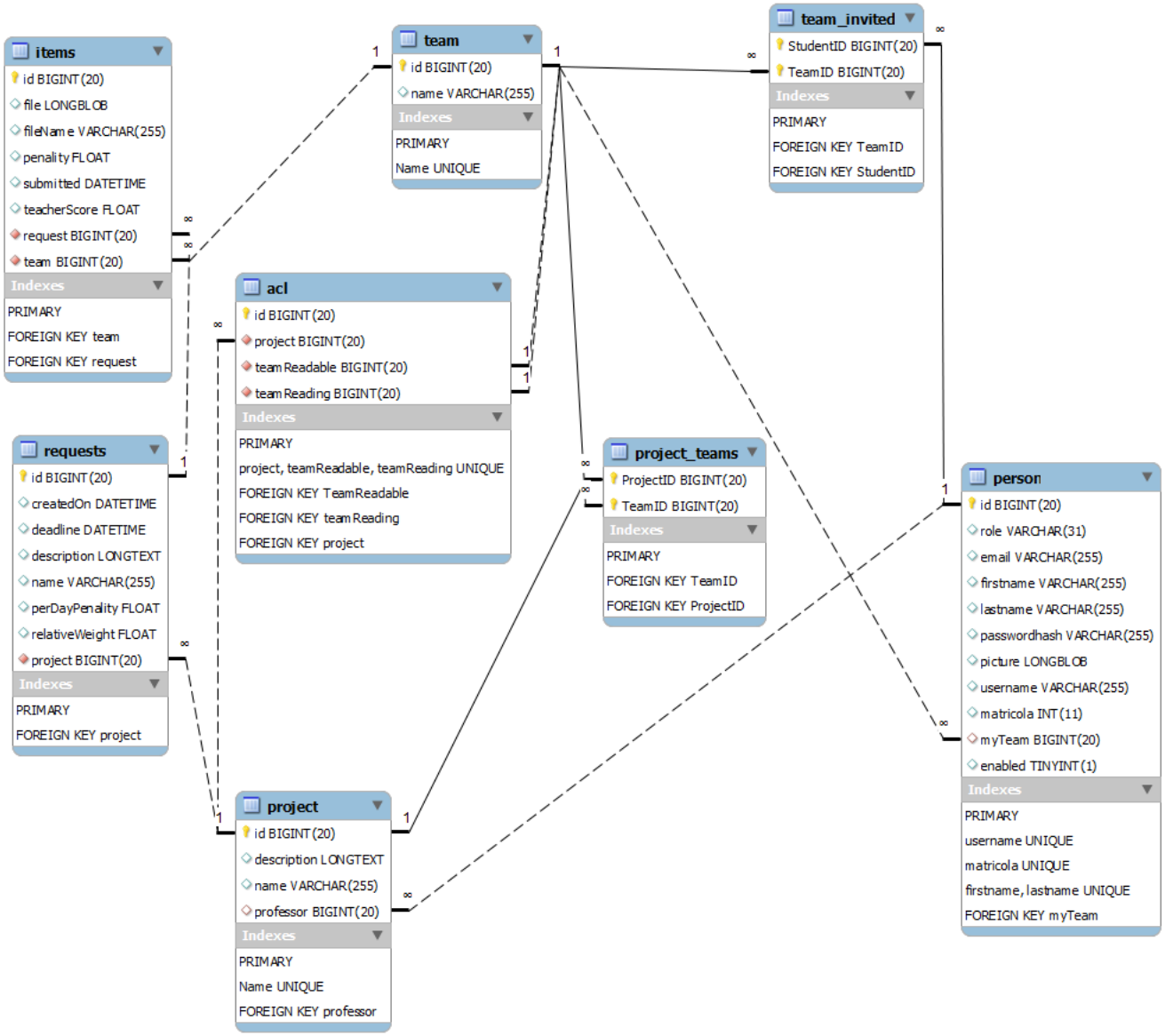

Figura 5: Relazioni tra le tabelle nel database.

4.2 Progettazione logica

La fase di progettazione logica si pone come obiettivo quello di trasformare lo schema concettuale, realizzato nella fase precedente,in uno schema logico che contenga la descrizione logica della struttura dati (tabelle,tracciati record,chiavi,ecc) per il DB per il sistema MPH.
In questa fase si applicano i concetti di ristrutturazione dello schema Entità-Relazione e successivamente i criteri per la traduzione verso il modello logico vero e proprio.

4.2.1 Ristrutturazione dello schema Entità-Relazione

1. Le entità figlie Studente, Admin e Professore vengono accorpate all'entità padre Persona a cui verrà aggiunto l'attributo Ruolo, che definisce (come afferma la parola) il ruolo che loro hanno nella vita reale e la parte del sistema di interesse. Nel caso di studente i due attributi aggiuntivi (quali matricola e Team) vengono anche utilizzati nei ruoli Admin e Professore ma hanno valore Null.

2. Avendo già inserito un ID in ogni entità nella fase precedente non serve introdurne altri per la identificazione univoca.

4.2.2 Traduzione verso il modello relazionale

1. l'associazione ACL essendo raggiunta da 3 cardinalità 0 a n per i criteri di realizzazione della progettazione logica diventa una tabella. Per ridurre la complessità inseriamo un ID univoco.

2. l'associazione project-team, avendo 2 cardinalità 0 a n, per i criteri di realizzazione della progettazione logica diventa una tabella. Consideriamo come chiave primaria l'unione dei 2 attributi team e progetto.

3. l'associazione invito, avendo le 2 cardinalità pari a 0 a n (con n maggiore di 2), per i criteri di realizzazione della progettazione logica diventa un tabella. Consideriamo come chiave primaria l'unione dei 2 attributi team e studente.

4. in tutte le altre associazioni che hanno una cardinalità pari a 1 a 1 (o 0 a 1) e 1 a n (o 0 a n) abbiamo l'inserimento di un attributo (tipicamente la chiave esterna) nella tabella che ha il collegamento 1 a 1.

La traduzione fisica della base di dati del sistema MPH risulta avere le seguenti relazioni:

- Persona(<u>ID</u>, Nome, Cognome, Email, Username, Password, Foto profilo, Matricola, Ruolo, Gruppo di lavoro, Attivo);

- Progetto(<u>ID</u>, Nome, Persona professore, Descrizione);

- Team (<u>ID</u>, Nome del gruppo);

- Elaborato(<u>ID</u>, File, Nome del file, Punteggio, Penalità, Richiesta, Team, Data di Rilascio);

- Richiesta(<u>ID</u>, Titolo, Progetto, Descrizione, Penalità per Giorno, Peso nel progetto, Deadline, Data di creazione);

- ACL (ID, Progetto, Team1, Team2);
- Invito(Studente, Team);
- Progetto-Team(Progetto, Team);

5 Design

5.1 Modelli di navigazione

In questa parte dello schema di navigazione dello studente non abbiamo fatto ipotesi, quali abbia o meno un gruppo o è iscritto ad almeno un progetto.
Procedendo da qualsiasi «Screen», dopo aver effettuato la login, possiamo andare verso lo schema 2 attraverso la funzione gestione().

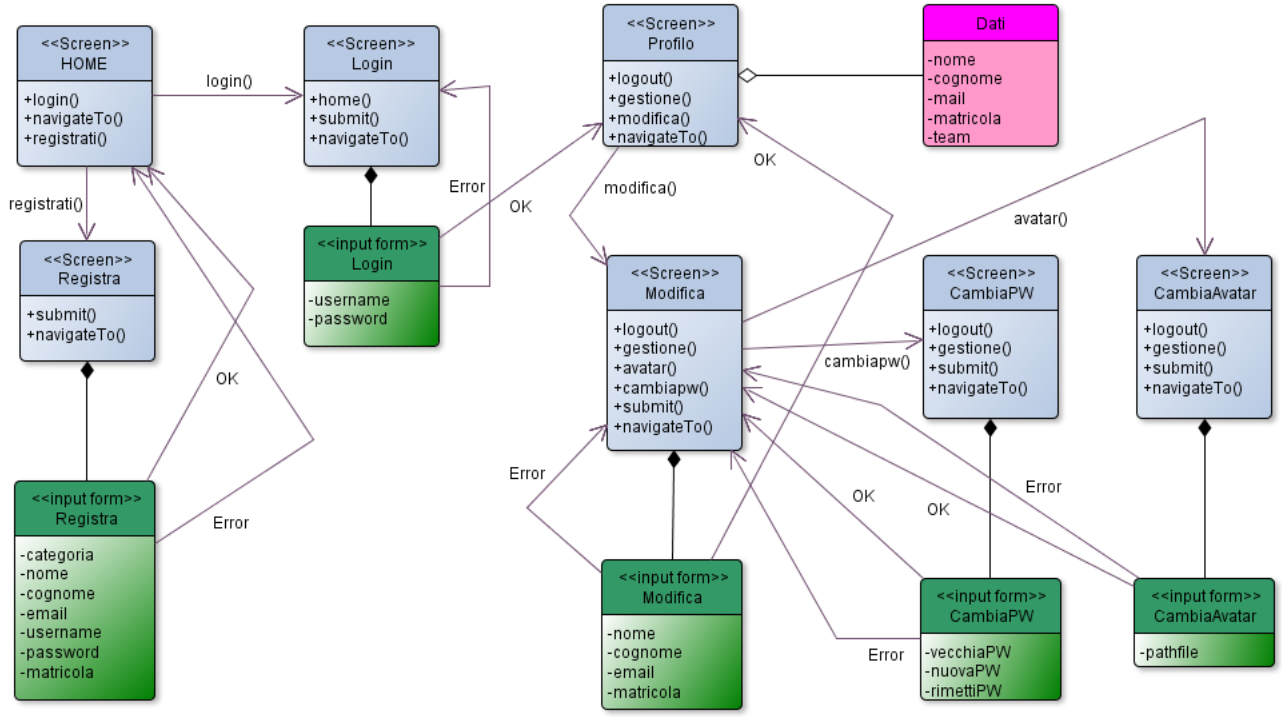

Figura 6: Funzionalità iniziali per uno studente.

In questo schema abbiamo differenziato 3 casi per lo studente, che coincidono con le tre possibili situazioni in cui un utente studente può trovarsi:

- E' iscritto ma non appartiene a nessun gruppo; (caso 1)
- Appartiene a un gruppo ma non è iscritto a nessun progetto; (caso 2)
- Appartiene a un gruppo ed è iscritto ad almeno un progetto. (caso 3)

Lo «Screen»Home Studente (in alto) rappresenta la schermata che appare all'utente quando è nel caso 1. Da questo punto si può procedere accettando (nel caso di aver ricevuto un invito) l'aggregazione a un gruppo o creando un gruppo. Eseguendo una delle operazioni (ipotizzando che vada a buon fine) arriviamo allo «Screen»Home Studente (posizionata centralmente) che individua il caso 2.
Se il gruppo invitante ha già un progetto, lo studente, accettando la richiesta, può accedere nella «Screen»Home Studente rappresentata nello schema successivo.
Dal caso 2 si può arrivare al caso 3 iscrivendosi a un progetto. Dopo aver effettuato tutta la procedura di iscrizione dall'«input form»Iscrivi si arriva al caso 3 (ipotizzando che l'operazione vada a buon fine).

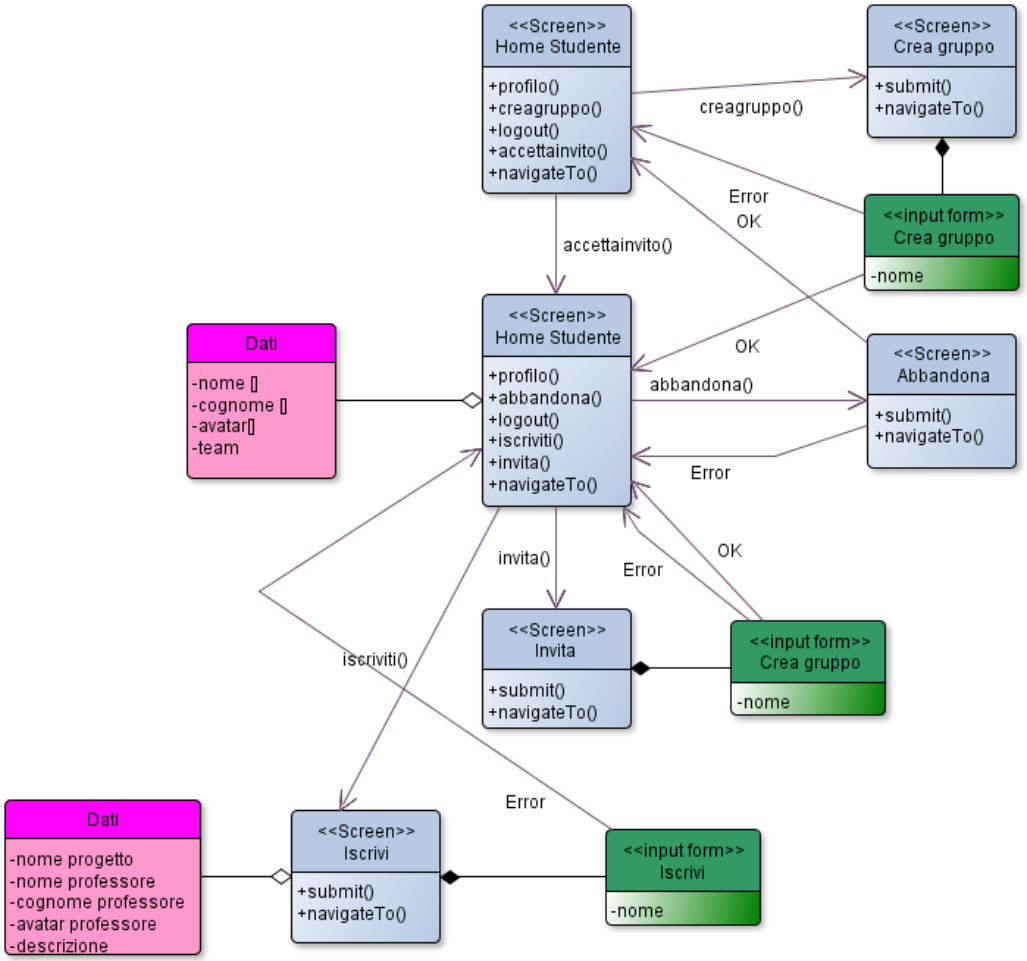

Figura 7: Funzionalità di gestione gruppi per uno studente.

Nell'ultimo schema realtivo all'utente studente è rappresentata tutta la struttura del programma per la gestione delle consegne dei progetti. Mancano alcune frecce (per non rendere il grafico incomprensibile) la cui funzione è di facile intuizione.

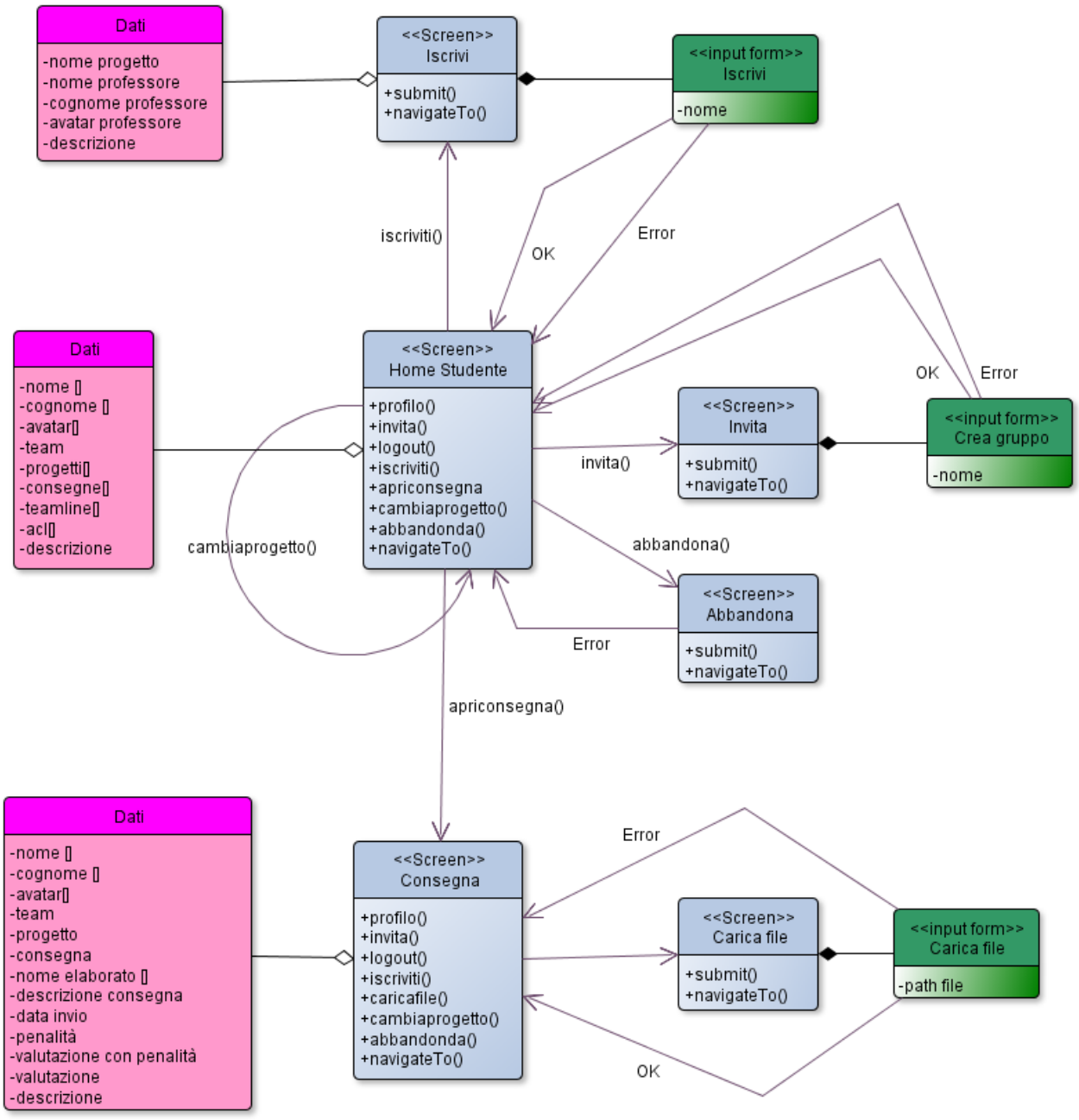

Figura 8: Funzionalità di gestione progetto per uno studente.

Sia per lo studente che per il professore è possibile accedere alla registrazione e alla modifica del profilo, rappresentata da uno schema identico (ad eccezione del campo team e matricola) pubblicato nella sezione dello studente (Si faccia riferimento allo schema di navigazione numero 6).

In questo schema abbiamo differenziato due casi per un professore abilitato, che coincidono con le due possibili situazioni in cui un utente del suo genere può trovarsi:

- E' iscritto ma non ha presentato nessun progetto; (caso 1)
- E' iscritto e ha presentato almeno un progetto. (caso 2)

Lo «Screen»Gestione (posizionato a sinistra) individua il caso 1. Presentando un progetto, il docente arriva allo «Screen»Gestione (posizionato centralmente) che rappresenta il caso 2.
Da questo punto è possibile fare tutte le operazioni utili al docente. Per motivi di spazio abbiamo inserito alcune funzioni nello schema 3.

Mancano alcune frecce (per non rendere il grafico incomprensibile) la cui funzione è di facile intuizione.

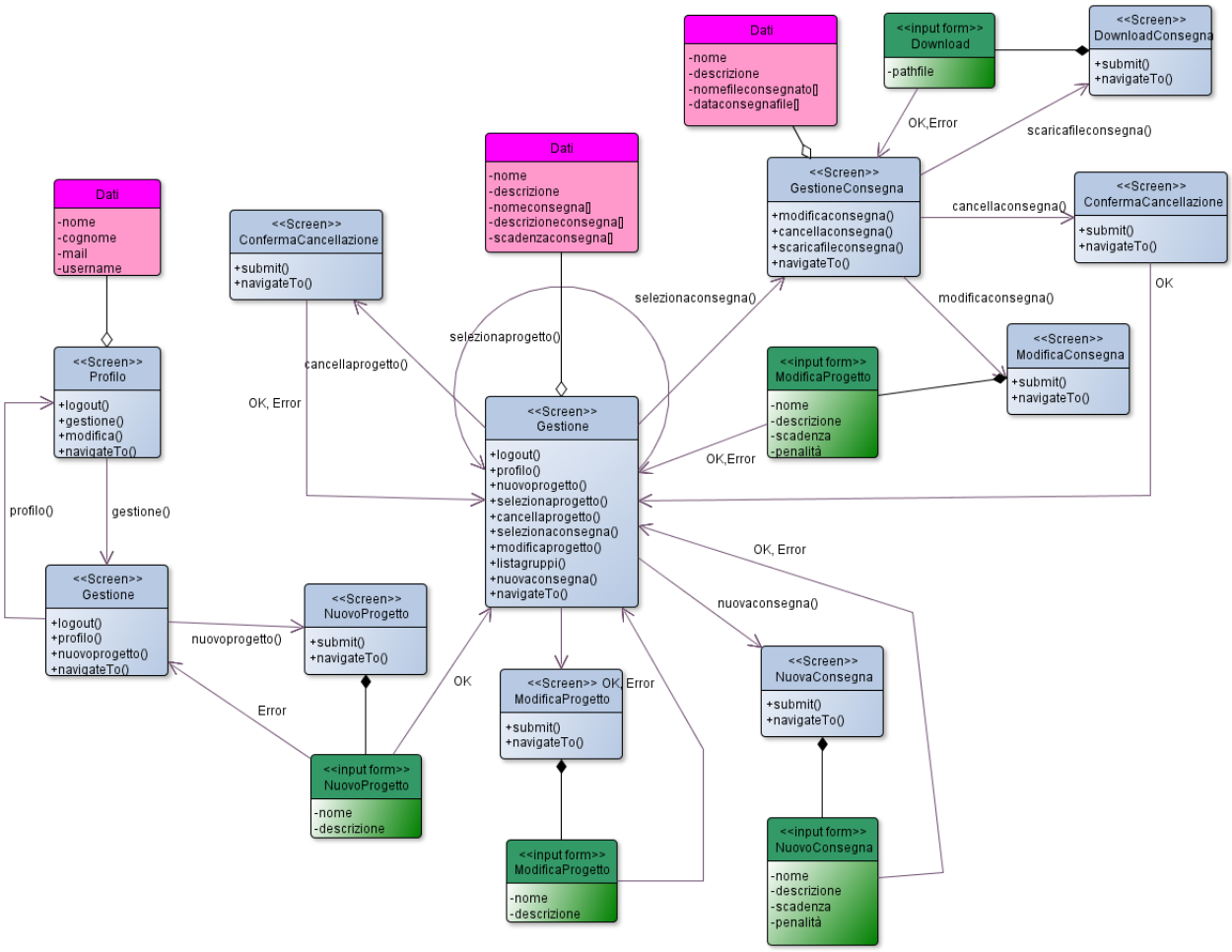

Figura 9: Funzionalità iniziali per un professore.

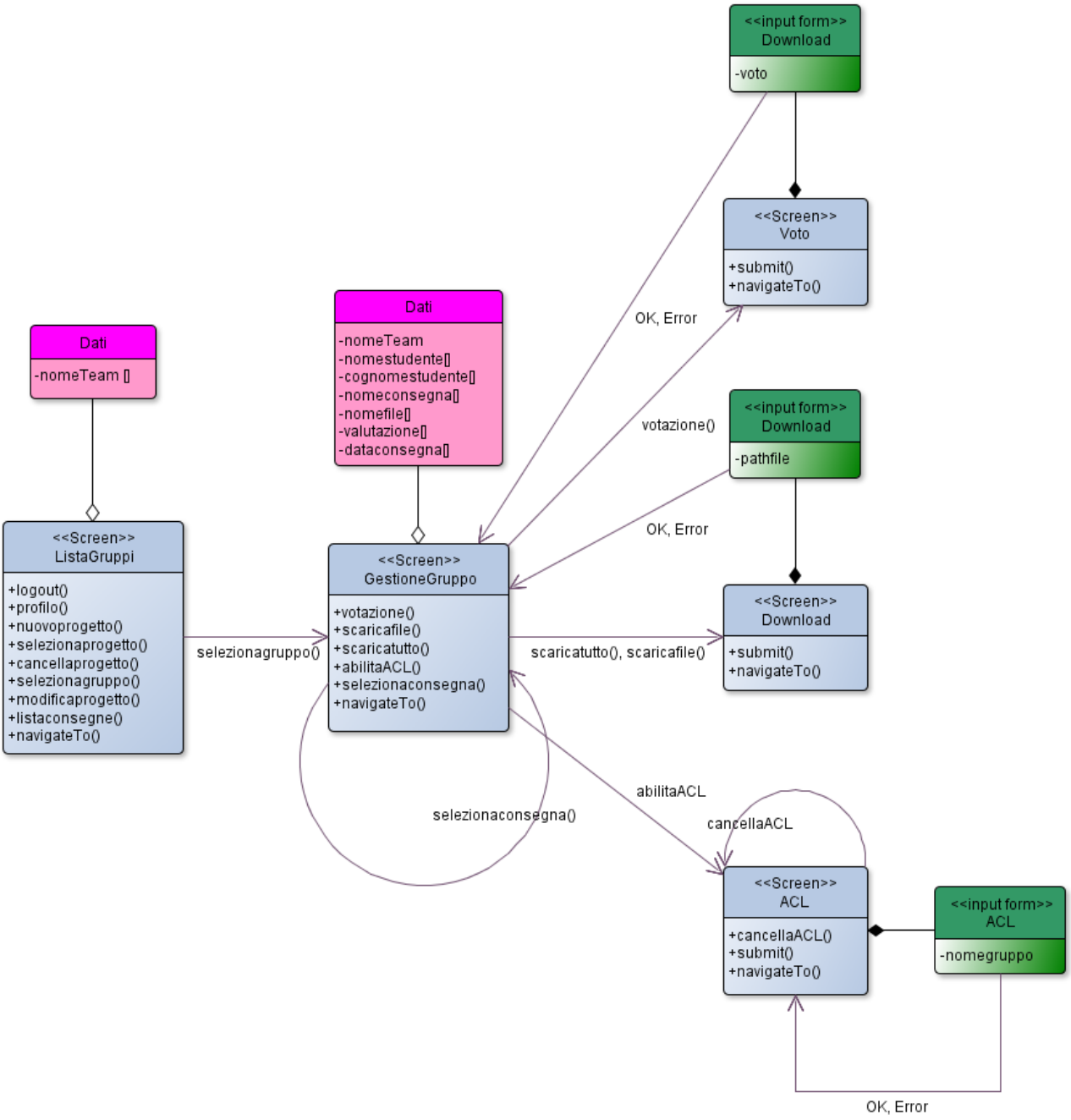

Figura 10: Funzionalità di gestione gruppi iscritti ad un progetto per un professore.

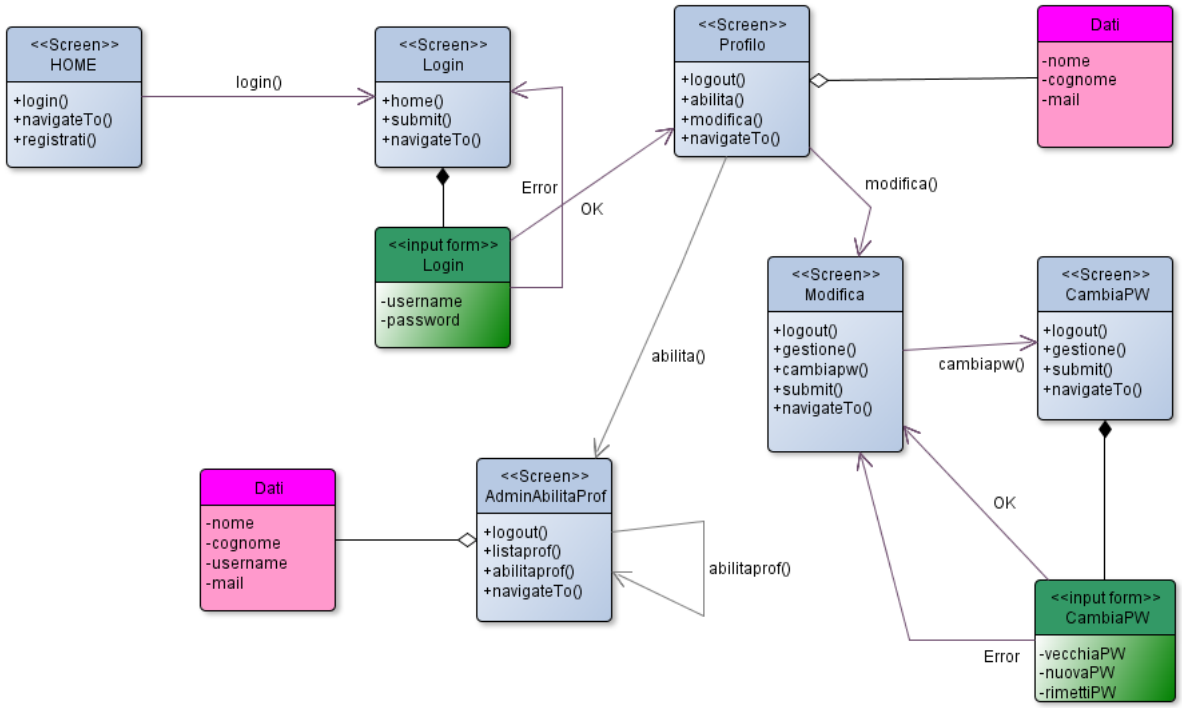

Figura 11: Funzionalità iniziali per un amministratore.

5.2 Diagrammi di analisi

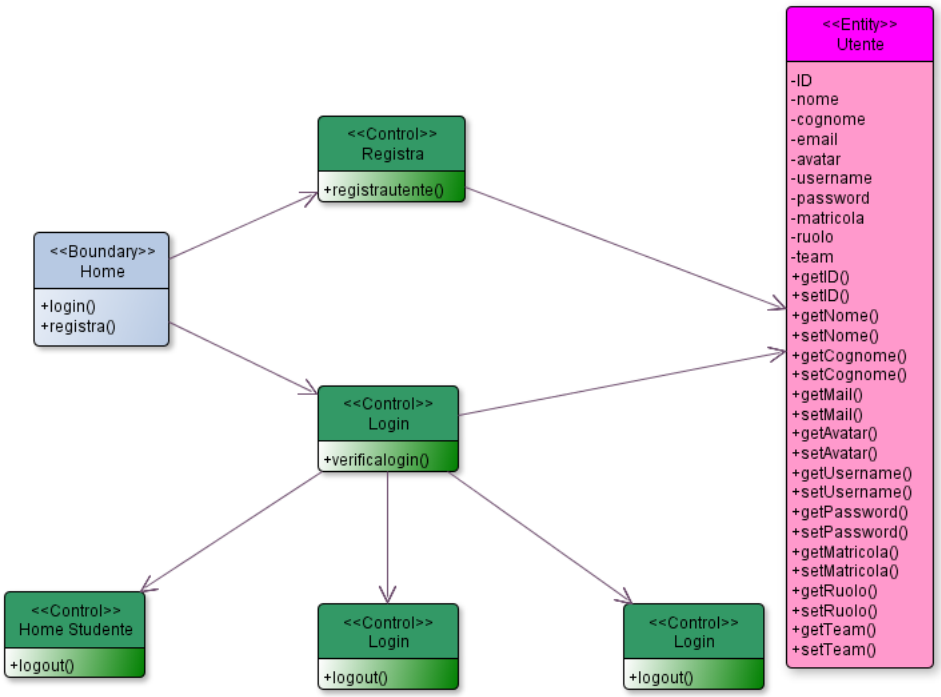

Figura 12: Composizione della funzionalità di login.

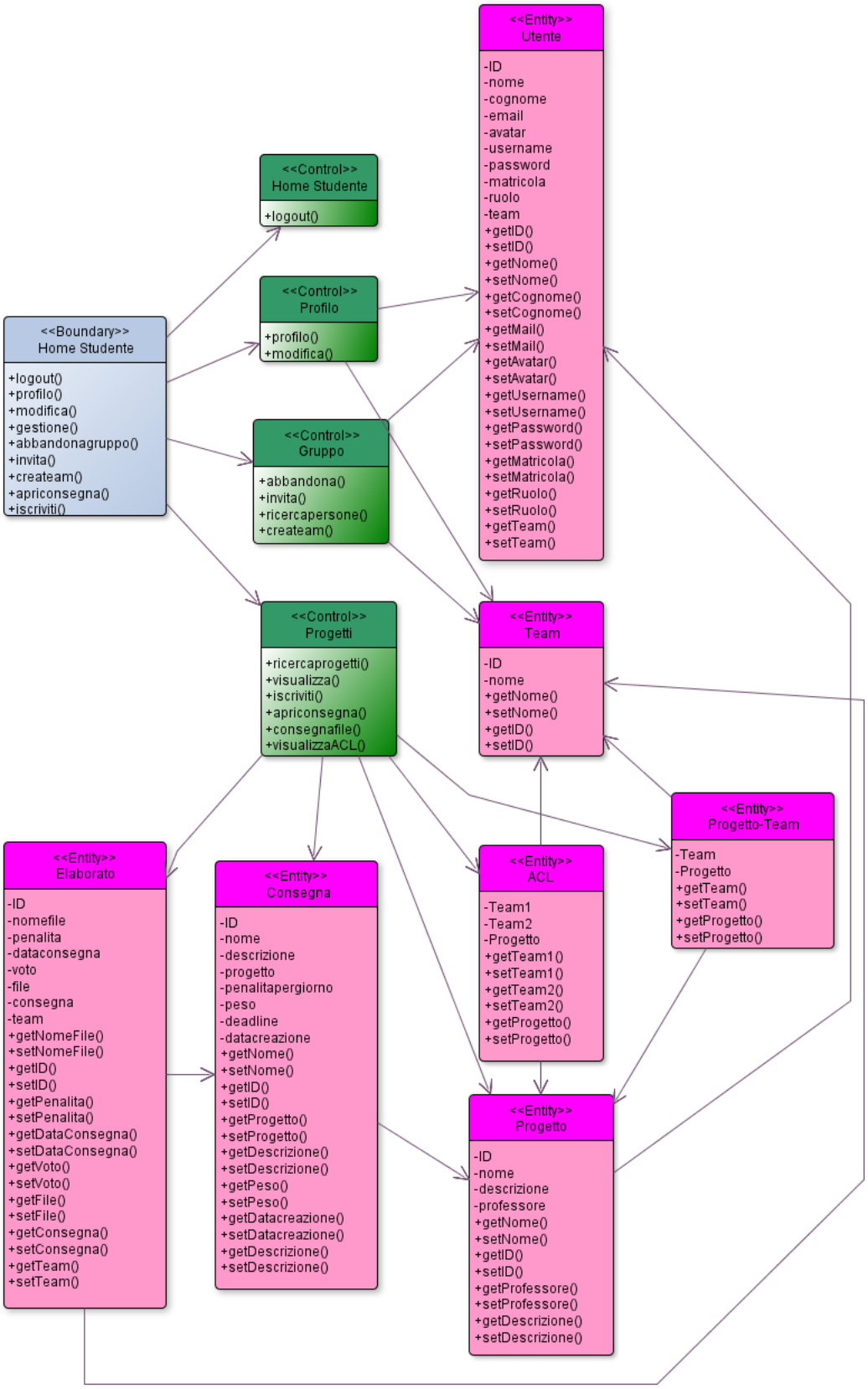

Figura 13: Composizione della funzionalità per uno studente.

Figura 14: Composizione della funzionalità per un professore.

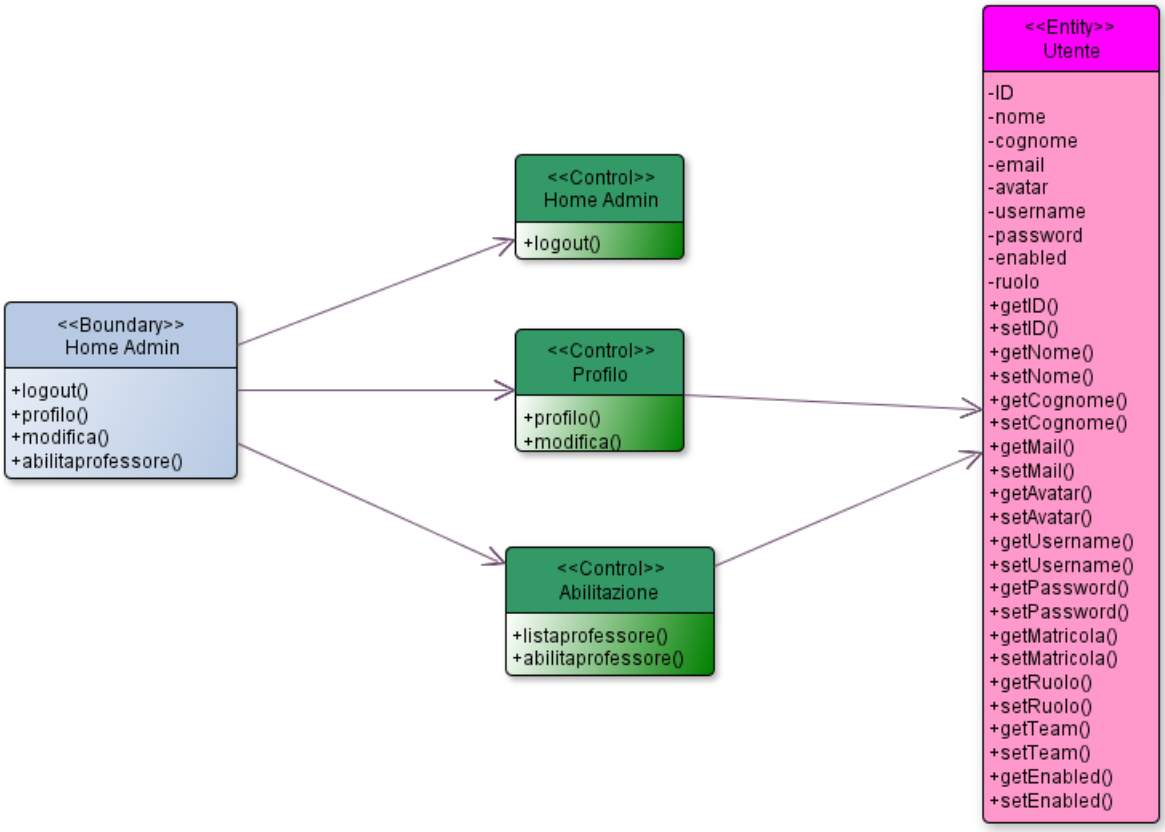

Figura 15: Composizione della funzionalità per un amministratore.

5.3 Diagrammi di sequenza

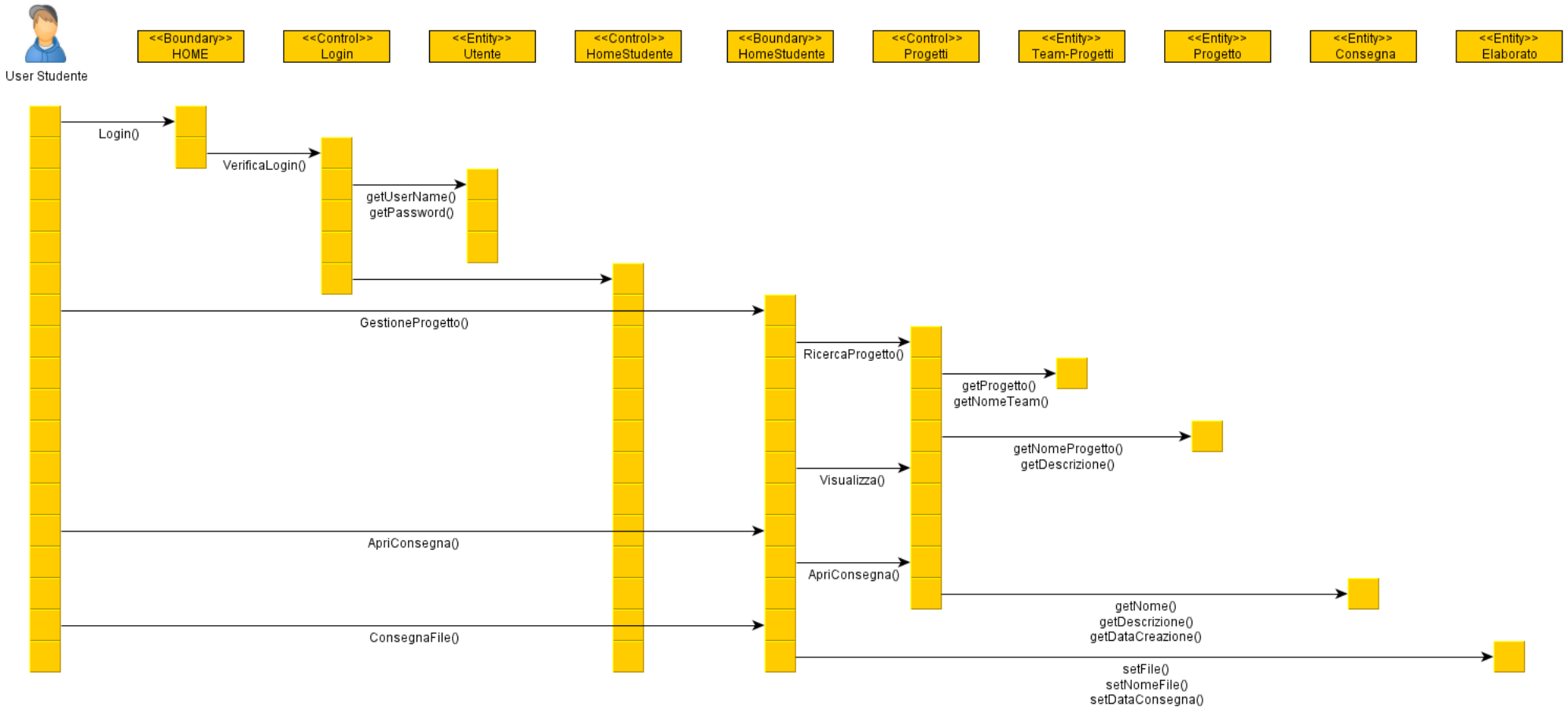

Figura 16: Diagramma di sequenza per l'invio di un file da parte di uno studente.

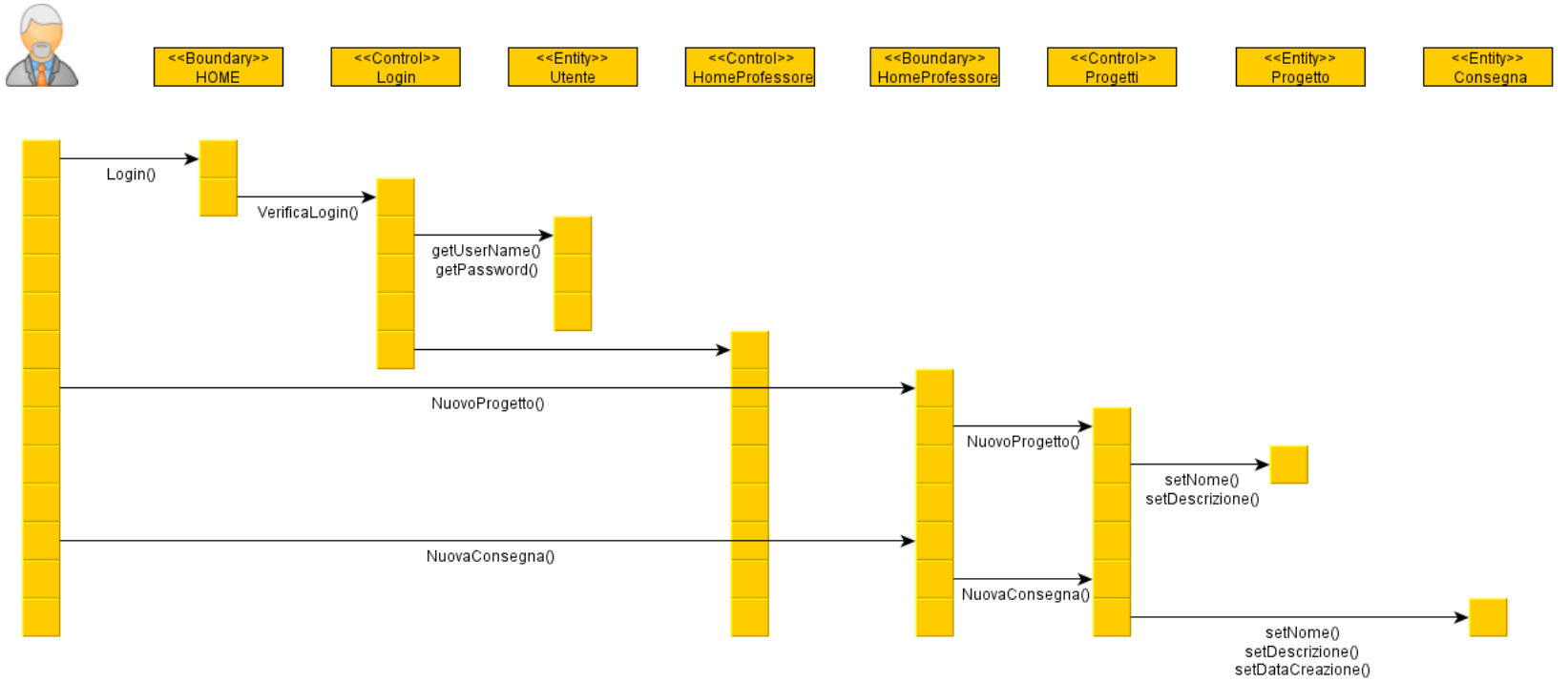

Figura 17: Diagramma di sequenza per la creazione di un progetto da parte di un professore.

5.4 Diagrammi di dettaglio

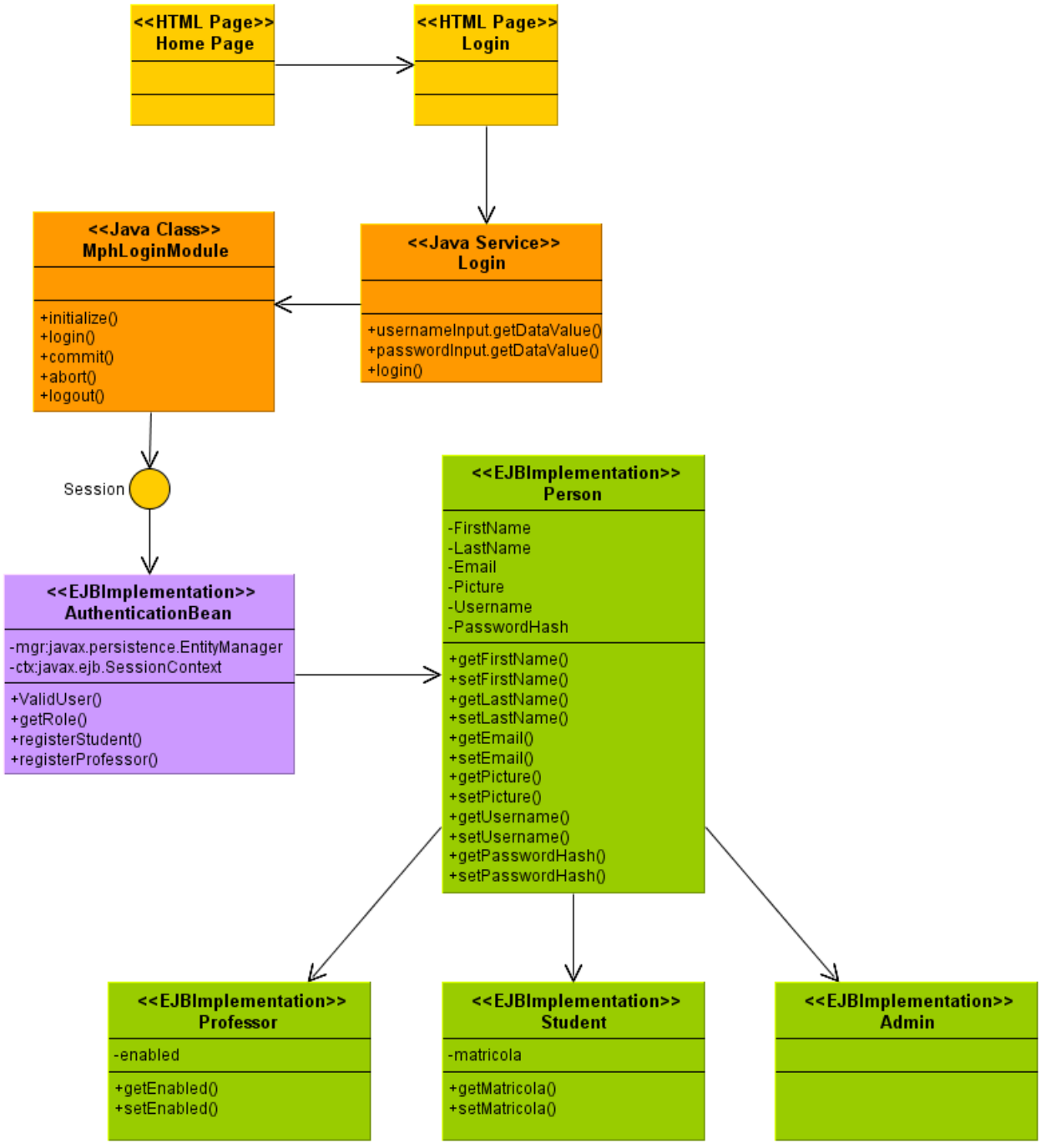

Figura 18: Componenti per la funzionalità di login.

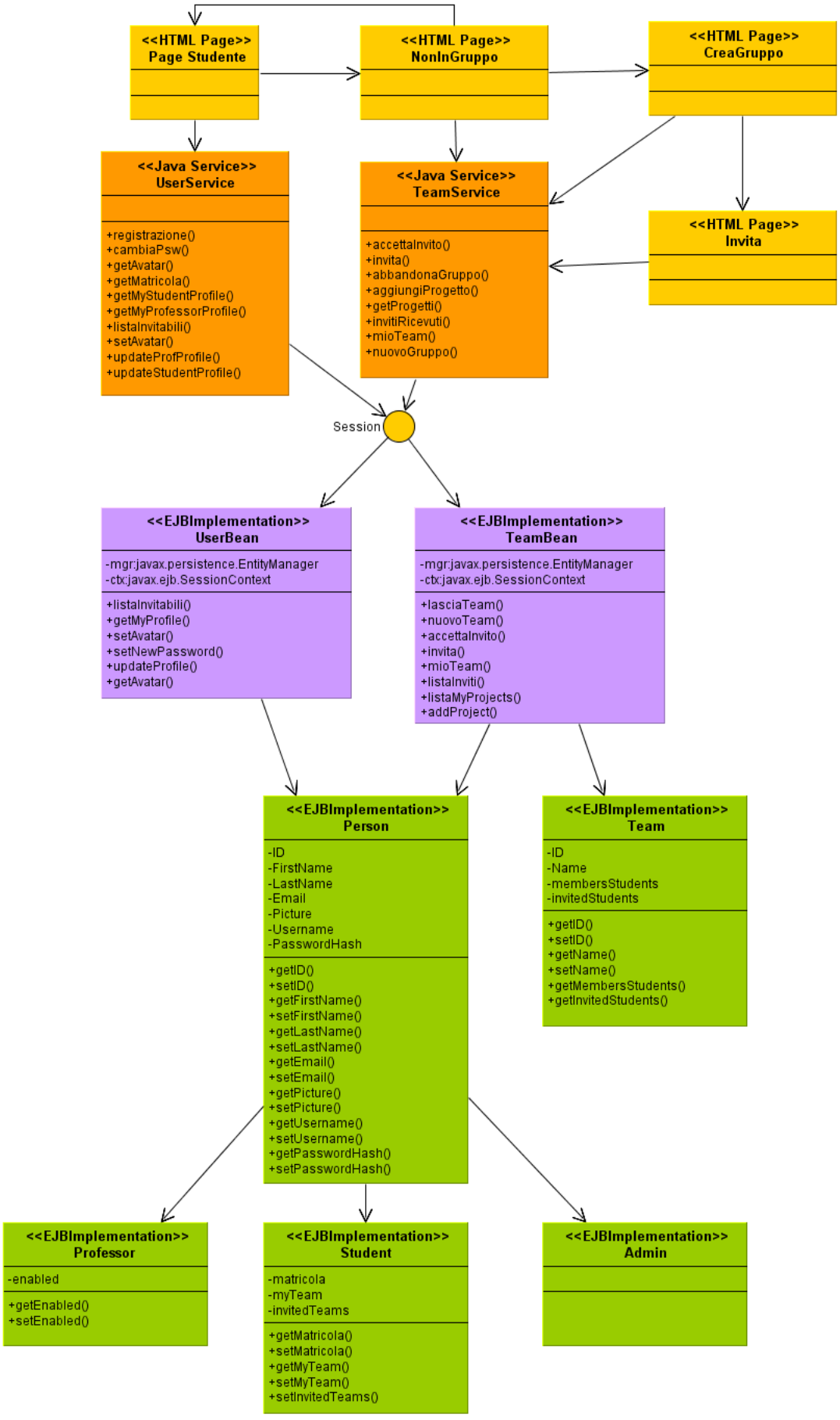

Figura 19: Componenti per le funzionalità di uno studente non iscritto ad un gruppo.

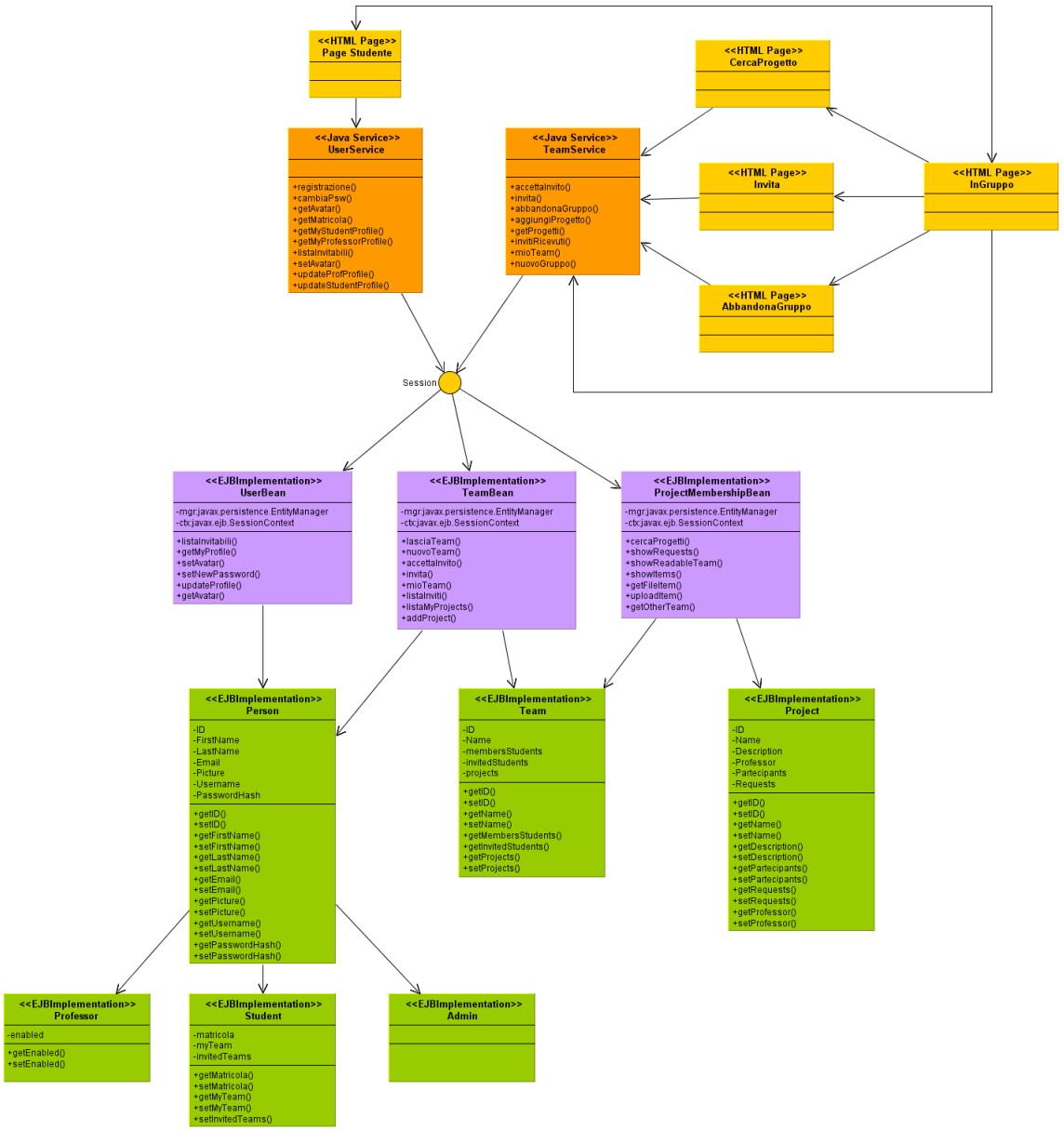

Figura 20: Componenti per le funzionalità di uno studente iscritto ad un gruppo ma senza un progetto.

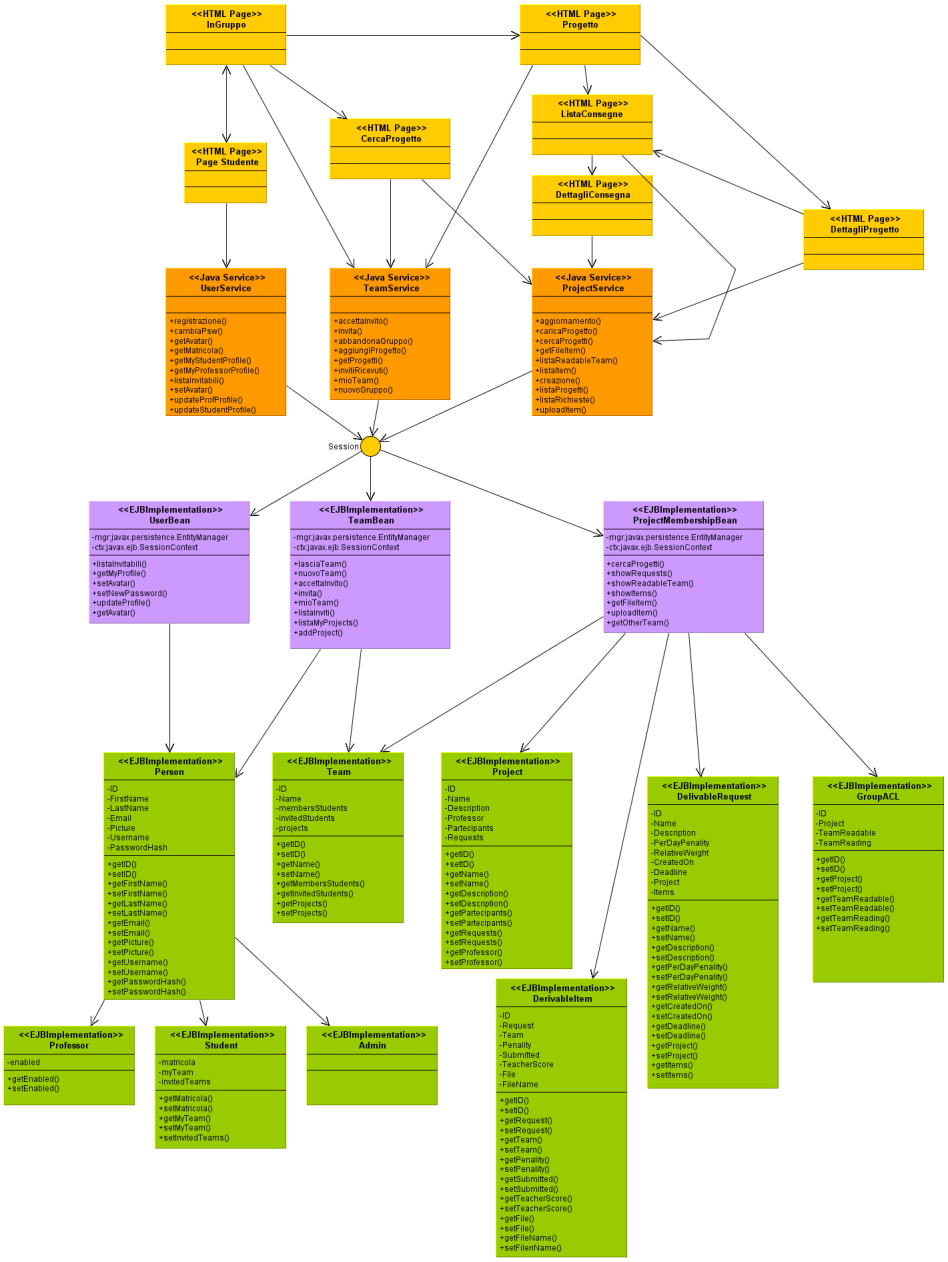

Figura 21: Componenti per le funzionalità di uno studente iscritto ad un gruppo con un progetto attivo.

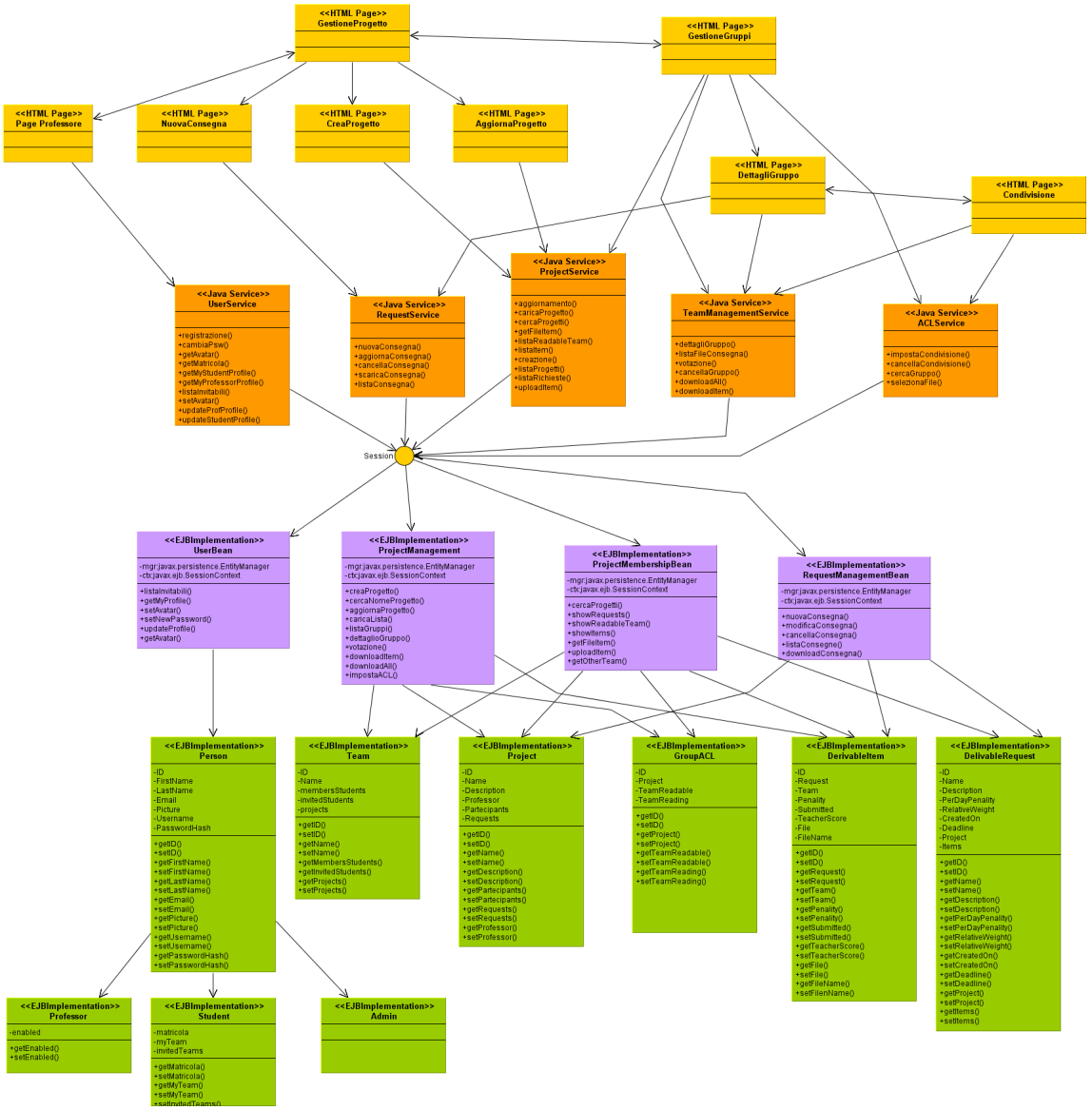

Figura 22: Componenti per le funzionalità di un professore abilitato.

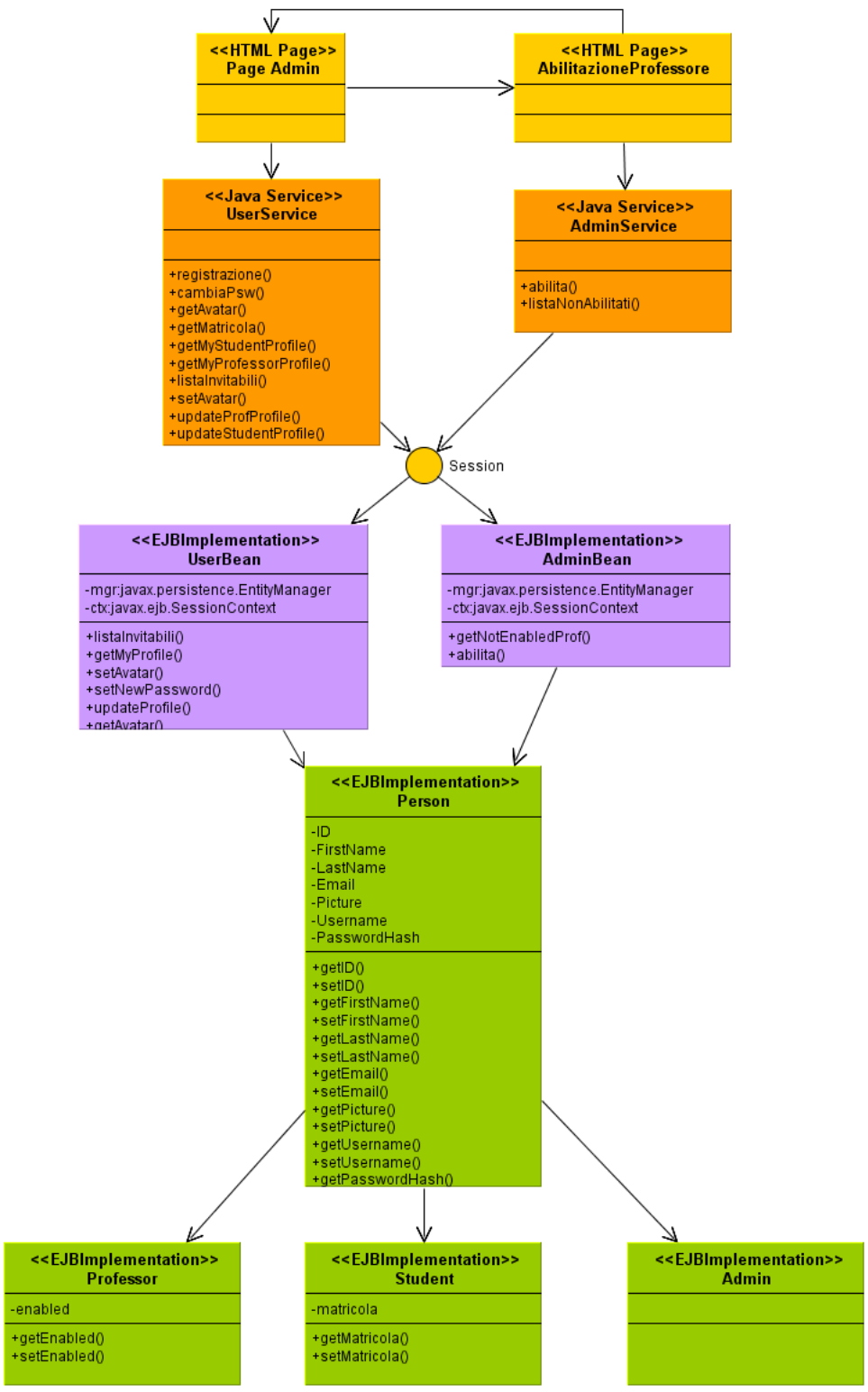

Figura 23: Componenti per le funzionalità di un amministratore.