Deliverable 3:
Component Diagram, Class Diagram, OCL

Autori:	**Matricola:**
Erik Nielsen	209300
Giulia Panozzo	211315
Nicola Gianuzzi	209309

DOCUMENTO DI PROGETTO

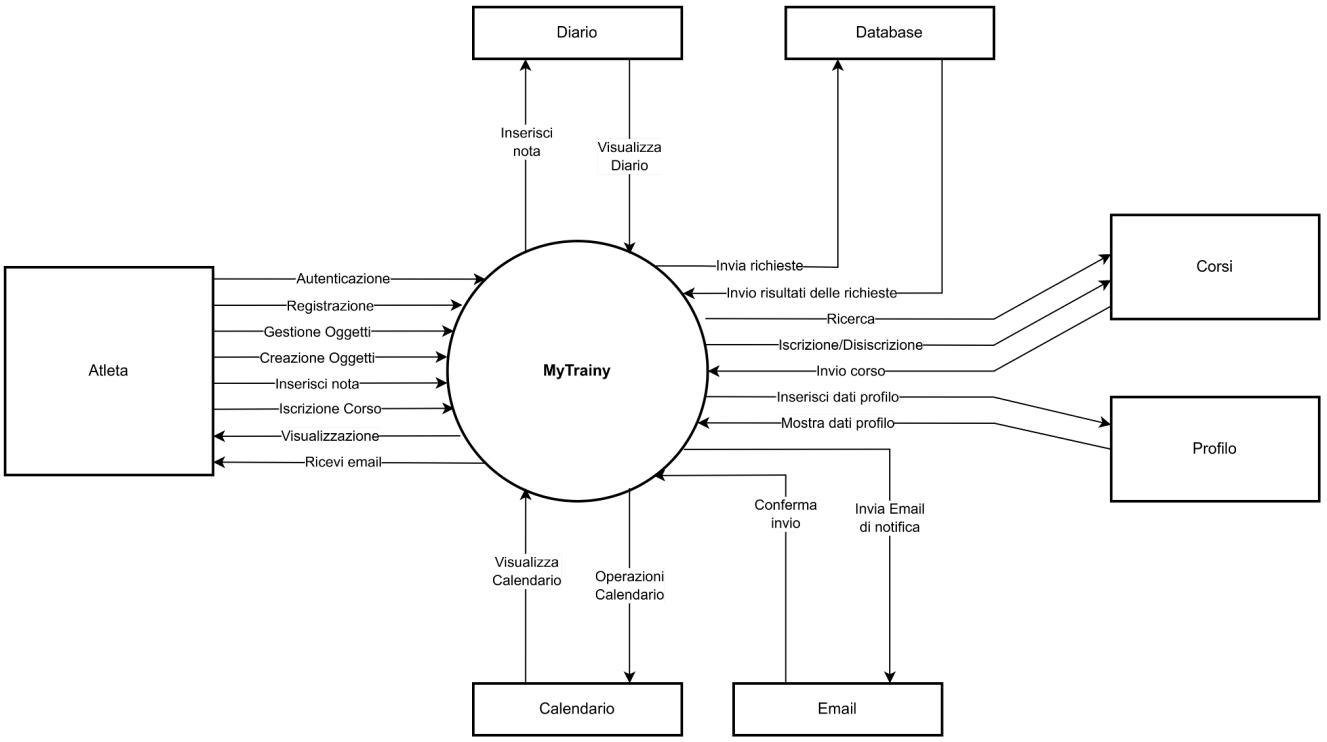
Indice

SCOPO DEL DOCUMENTO

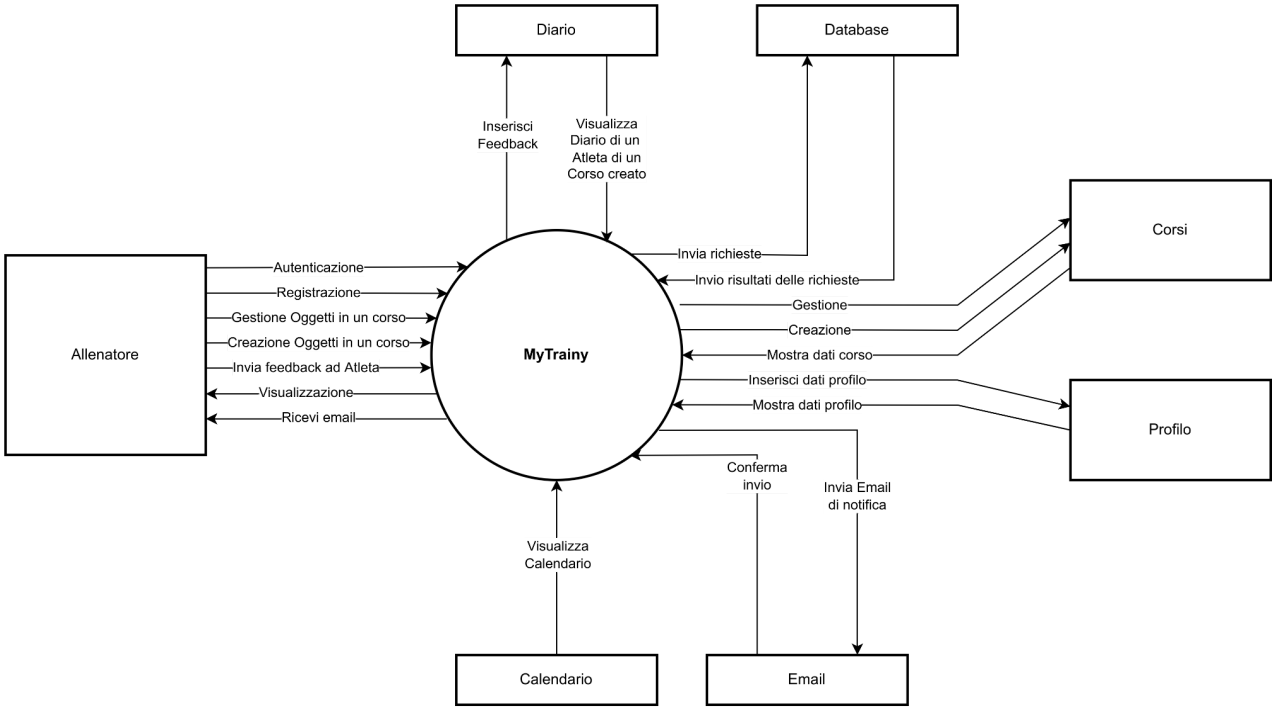
Nel presente documento viene presentata l'architettura in termini di componenti interni al sistema definiti sulla base dei requisiti analizzati nei precedenti documenti, minimizzando il livello di *coesione*. A partire dall'analisi dei Context Diagram precedentemente creati, viene poi adottato l'uso di Component Diagram per rappresentare l'interconnessione tra i vari componenti, identificando quindi le interfacce tra questi e verso sistemi esterni. Viene inoltre valutato il livello di accoppiamento tra i componenti.

Viene poi utilizzato il Class Diagram in Unified Modeling Language e in Object Constraint Language per definire nel dettaglio l'architettura del sistema, dettagliando da un lato le classi che dovranno essere implementate a livello di codice e dall'altro la logica che regola il comportamento del software. Le classi vengono rappresentate tramite un diagramma delle classi in linguaggio UML, mentre la logica utilizza OCL perché tali concetti non sono esprimibili in nessun altro modo formale nel contesto di UML.

CONTEXT DIAGRAM



Context Diagram per l'applicazione MyTrainy per l'utente Atleta



Context Diagram per l'applicazione MyTrainy per l'utente Allenatore

ANALISI DEI COMPONENTI

Definizione dei componenti:

In questa sezione vengono definiti i componenti.
NB: Per una questione di coerenza e uniformità a livello funzionale, abbiamo raggruppato i componenti (front-end e back-end) che si occupano di una stessa area dell'applicazione.

Login Page e Authentication System

Motivazione: Dato il RF2 l'utente si autentica, e sono stati quindi identificati un componente Login Page (front-end) e un componente Authentication System (back-end) per la gestione dell'autenticazione. All'utente è richiesto l'inserimento delle credenziali nella pagina di Login e l'autenticazione viene poi gestita nel back-end. In particolare quindi i due componenti si occupano della gestione del login dell'utente autenticandolo e mantenendo poi attiva la sessione utente all'interno dell'applicazione.
Coesione:

- Login Page = Livello 5 - Comunicazionale
  Spiegazione: Il componente gestisce la visualizzazione della procedura di autenticazione all'interno dell'applicazione.
- Authentication System = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di autenticazione dell'utente all'interno del sistema.

Registration Page e Registration System

Motivazione: Dato il RF1, l'utente se non è già autenticato all'interno del sistema ha bisogno di registrarsi e abbiamo quindi identificato i componenti Registration Page (front-end) e Registration System per la gestione della registrazione. All'utente è richiesto l'inserimento delle proprie credenziali e di effettuare una scelta di tipologia per l'utente, in seguito la registrazione è gestita nel back-end.
Coesione:

- Registration Page = Livello 5 - Comunicazionale
  Spiegazione: Il componente gestisce la visualizzazione della procedura di registrazione all'interno dell'applicazione.
- Registration System = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di registrazione di un utente all'interno del sistema.

Calendar Homepage

Motivazione: Dato il RF3, l'utente in seguito all'accesso al sistema ottiene la Home Page con il proprio calendario; abbiamo quindi identificato il componente di front-end della pagina web principale, la Calendar Homepage. Questo componente di front-end ottiene i dati relativi agli allenamenti dell'utente una volta che egli si è autenticato nel sistema. Da questa pagina si ha

poi accesso alle altre pagine dell'applicazione per cui vengono visualizzati i dati specifici per ognuna di esse.
Coesione: Livello 6 - Informazionale
Spiegazione: Il componente contiene elementi che hanno porzioni di codice indipendenti e un proprio punto di ingresso ed uscita; tutti gli elementi però agiscono sulla stessa struttura dati.

Email Server

Motivazione: Dati RF1, RF2, RF3, RF4, RF6, l'utente riceve una email in diversi casi (ad avvenuta registrazione, come notifica per l'allenamento, ...) ed è per questo che abbiamo individuato il componente Email Server che gestisce questa funzionalità.
Coesione: Livello 7 - Funzionale
Spiegazione: Il componente gestisce la funzionalità di invio delle email all'utente.

Program/Card Page, Create Program/Card Page e Program/Card Management

Motivazione: Dati RF3, RF4, l'utente può gestire i propri programmi e le proprie schede di allenamento, per questo motivo abbiamo individuato due componenti di front-end: uno per la visualizzazione dei propri programmi e schede (Program/Card Page), che riceve tutti i dati e gli aggiornamenti relativi ad essi, e uno per la pagina di creazione di questi due elementi (Create Program/Card Page), che ottiene dall'utente le informazioni per la creazione di un nuovo programma/scheda. Inoltre è necessario inserire un componente Program/Card Management (di back-end) per la parte di gestione. Quest'ultimo può ricevere in input dati dall'utente che decide di apportare modifiche ai suoi programmi/schede e inoltre riceve il nuovo programma/scheda creata nel front-end nominato precedentemente.
Coesione:

- Program/Card Page = Livello 6 - Informazionale
  Spiegazione: Il componente gestisce la visualizzazione della scheda e/o del programma; ogni elemento è indipendente ma tutti agiscono sulla stessa struttura dati.
- Create Program/Card Page = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di creazione del programma e/o della scheda.
- Program/Card Management = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di gestione (modifica, cancellazione) del programma e/o della scheda.

Course Page, Create Course Page e Course Management

Motivazione: Dato RF4, l'utente allenatore può creare e gestire i propri corsi da fornire agli utenti atleti.
In modo simmetrico alla parte che riguarda i programmi e le schede, abbiamo individuato due componenti di Front-end, uno per la visualizzazione del corso (Course Page) e uno per la creazione di uno nuovo (Create Course Page), e un componente di Back-end per la parte di gestione (Course Management).
Course Page riceve in input gli i dati e gli aggiornamenti relativi ai corsi e dall'utente riceve i dati per la ricerca nel caso in cui egli volesse effettuarla; Create Course Page necessita di

informazioni da parte dell'utente per la creazione di un nuovo corso; il componente Course Management necessita anch'esso di input da parte dell'utente in caso di modifiche relative ai corsi e in più riceve il nuovo corso creato nel front-end.

Coesione:

- Course Page = Livello 6 - Informazionale
  Spiegazione: Il componente gestisce la visualizzazione del corso in dettaglio e la ricerca dei corsi.
- Create Course Page = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di creazione del corso.
- Course Management = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di gestione (modifica, cancellazione) del corso.

Diary Page e Diary Management

Motivazione: Dato RF5 l'utente può gestire un diario relativo all'allenamento. Abbiamo quindi identificato il componente Diary Page (front-end) e il componente Diary Management (back-end) per la parte di gestione.
Diary Page ottiene i dati relativi al diario e ai suoi aggiornamenti; il componente di back-end invece permette di gestire il diario e riceve in input dall'utente commenti o feedback.

Coesione:

- Diary Page = Livello 6 - Informazionale
  Spiegazione: Il componente gestisce la visualizzazione del diario e delle sue componenti (commenti, feedback).
- Diary Management = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di gestione (modifica, inserimento commenti/feedback) del diario.

Profile Page e Profile Management

Motivazione: Dato RF6 l'utente può gestire il proprio profilo personale all'interno dell'applicazione. Abbiamo quindi identificato il componente Profile Page (front-end) e il componente Profile Management (back-end) per la parte di gestione.
Profile Page ottiene i dati relativi al profilo e ai suoi aggiornamenti; il componente di back-end invece permette di gestire il profilo e riceve in input dall'utente informazioni come per esempio dati anagrafici e foto profilo e la nuova password nel caso in cui la si voglia cambiare.

Coesione:

- Profile Page = Livello 6 - Informazionale
  Spiegazione: Il componente gestisce la visualizzazione del profilo e dei suoi componenti (dati anagrafici, foto profilo, ...)
- Profile Management = Livello 7 - Funzionale
  Spiegazione: Il componente gestisce la funzionalità di gestione (modifica, inserimento dati) del profilo.

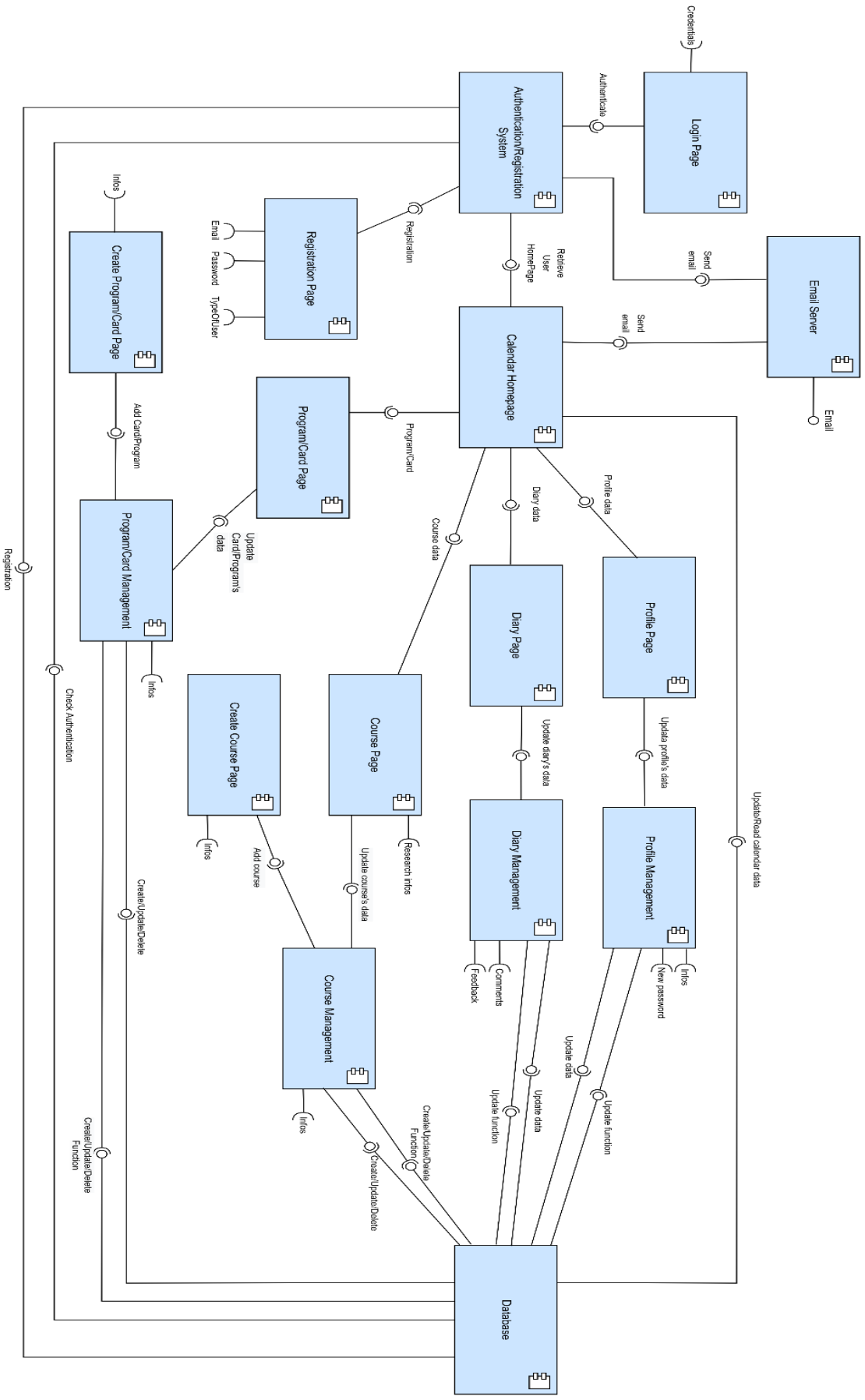
Database

Motivazione: Il Database è un componente essenziale per il nostro sistema, in quanto ci permette di organizzare tutti i dati della nostra applicazione, fornisce ai vari componenti di gestione (back-end) le funzionalità CRUD che necessitano e controlla la validità dell'autenticazione dell'utente nel sistema.
In input riceve le registrazioni dei nuovi utenti e tutti gli aggiornamenti e le modifiche relative ai dati di ogni singolo utente.
Coesione: Livello 2 - Logica
Spiegazione: Il componente contiene al suo interno elementi correlati tra loro, di cui uno o più vengono selezionati dal modulo (componente) chiamante.

Tabella riassuntiva dei livelli di coesione dei componenti:

Componenti / Livelli di Coesione	1. **Causale**	2. **Logica**	3. **Temporale**	4. **Procedurale**	5. **Comunicazionale**	6. **Informazionale**	7. **Funzionale**
Login Page					X		
Authentication System							X
Registration Page					X		
Registration System							X
Calendar Homepage						X	
Email Server							X
Program/Card Page						X	
Create Program/Card Page							X
Program/Card Management							X
Course Page						X	
Create Course Page							X
Course Management							X
Diary Page						X	
Diary Management							X
Profile Page						X	
Profile Management							X
Database		X					

COMPONENT DIAGRAM



Descrizione Componenti e Interfacce

Login Page

Descrizione: il componente si occupa delle funzionalità di front-end della pagina di autenticazione. Al suo interno si trovano il form per inserire le credenziali, il link per la registrazione e la richiesta delle credenziali in caso di smarrimento.

Interfaccia richiesta - Credentials: username e password fanno parte delle credenziali. Sono richieste all'utente per l'autenticazione al servizio.

Interfaccia fornita - Authenticate: La sessione di autenticazione dell'utente è condivisa all'interno dell'applicazione. Questa include dati relativi al nome dell'utente, metodo di autenticazione, e data dell'ultimo log-in.

Registration Page

Descrizione: il componente si occupa delle funzionalità di front-end della pagina di registrazione. Al suo interno si trovano il form per inserire le credenziali e il tipo di utente per i nuovi utilizzatori del servizio.

Interfaccia richiesta - Email, Password, Type of user: email, password, type of user fanno parte delle informazioni che l'utente deve fornire ai fini della registrazione.

Interfaccia fornita - Registration: la registrazione include la conferma di avvenuta registrazione e l'invio delle informazioni al componente Authentication/Registration System.

Authentication/Registration System

Descrizione: il componente si occupa delle funzionalità di back-end dell'autenticazione e registrazione. In particolare gestisce il controllo delle credenziali facendo un "check" con il database prima di passare alla Homepage e svolge i compiti di invio dati al database in caso di richiesta di registrazione.

Interfaccia richiesta - Registration: la registrazione include la conferma di avvenuta registrazione e la ricezione delle informazioni dal componente registration page.

Interfaccia richiesta - Check authentication: il componente richiede il controllo delle credenziali inserite al database

Interfaccia fornita - Retrieve user homepage: ad autenticazione avvenuta avviene il passaggio dalla pagina di login alla propria homepage calendario.

Interfaccia fornita - Send email: nei casi previsti avviene una richiesta di invio notifica via email

Calendar Homepage

Descrizione: il componente si occupa delle funzionalità di front-end della pagina calendario che funge da principale homepage del servizio. Da questa pagina ci sono tutti i collegamenti per usufruire delle principali funzioni dell' applicazione, nonché gli altri elementi di front-end.

Interfaccia richiesta - Retrieve user homepage: dopo l'autenticazione, l'interfaccia recupera le informazioni utente per ottenere il calendario personale.

Interfaccia richiesta - Update/read calendar data: il database fornisce le operazioni di CRUD al calendario per dare le informazioni all'utente.

Interfaccia fornita - Course data: i dati necessari per l'utente riguardo i corsi in cui è coinvolto vengono forniti dal calendario al corso.

Interfaccia fornita - Program/card: i dati necessari per l'utente riguardo i programmi e schede in cui è coinvolto vengono forniti dal calendario alla pagina programmi/schede.

Interfaccia fornita - Diary data: i dati necessari per l'utente riguardo il diario personale vengono forniti dal calendario alla pagina del diario.

Interfaccia fornita - Profile data: i dati necessari per l'utente riguardo il profilo personale vengono forniti dal calendario alla pagina programmi/schede.

Interfaccia fornita - Send email: i dati necessari per l'invio delle notifiche email vengono forniti dal calendario all'Email server

Course Page

Descrizione: il componente svolge le funzionalità principali di front-end della pagina dei corsi. In particolare ha la lista dei corsi, permette una ricerca dei corsi, e ha il collegamento alle principali funzionalità di gestione dei corsi.

Interfaccia richiesta - Course data: la pagina dei corsi ottiene i dati necessari dell'utente dall'homepage.

Interfaccia richiesta - Update course's data: la pagina dei corsi ottiene i dati necessari per aggiornarsi dopo eventuali modifiche in gestione corsi.

Interfaccia richiesta - Research infos: si possono inserire delle parole chiave per ricercare i corsi preferiti.

Course Management

Descrizione: il componente svolge le funzionalità principali di back-end della pagina dei corsi. Permette la modifica e la cancellazione dei corsi e ha il collegamento per creare nuovi corsi e per permettere le operazioni di CRUD con il database.

Interfaccia richiesta - Create/Update/Delete function: vengono richieste le operazioni di CRUD al database per garantire i dati corretti e necessari.

Interfaccia richiesta - Infos: vengono inserite le informazioni necessarie per permettere il corretto funzionamento e sequenza del componente.

Interfaccia richiesta - Add course: il nuovo corso viene inserito nella gestione corsi per effettuare le operazioni necessarie.

Interfaccia fornita - Create/Update/Delete: vengono effettuate le operazioni di CRUD con il database per garantire i dati corretti e necessari.

Interfaccia fornita - Update course's data: i dati modificati all'interno della gestione vengono inseriti nel front-end e aggiornati.

Create Course Page

Descrizione: il componente svolge le funzionalità principali di front-end della pagina di creazione dei corsi. In particolare sono presenti tutti i form e moduli che permettono la creazione di nuovi corsi e i collegamenti alla gestione di essi per portare a termine le operazioni di creazione.

Interfaccia richiesta - Infos: vengono inserite le informazioni necessarie per permettere il corretto funzionamento e sequenza del componente.

Interfaccia fornita - Add course: le informazioni ottenute dal front-end vengono fatte gestire dal componente di gestione.

Program/Card Page

Descrizione: il componente svolge le funzionalità principali di front-end della pagina dei programmi e schede. In particolare ha i collegamenti per procedere a tutta la gestione dei programmi e delle schede.

Interfaccia richiesta - Program/card: dall'homepage vengono fornite le informazioni necessarie al front-end di programmi/schede.

Interfaccia richiesta - Update Program/Card data: le informazioni da gestione programmi/schede vengono aggiornate nel front-end.

Program/Card Management

Descrizione: il componente svolge le funzionalità principali di back-end della pagina di programmi e schede. Permette la modifica e la cancellazione di programmi/schede e ha il collegamento per creare nuovi corsi e per permettere le operazioni di CRUD con il database.

Interfaccia richiesta - Infos: vengono inserite le informazioni necessarie per permettere il corretto funzionamento e sequenza del componente.

Interfaccia richiesta - Add card/program: le informazioni necessarie alla creazione di un nuovo programma vengono gestite per portare al termine l'operazione.

Interfaccia richiesta - Create/Update/Delete function: vengono richieste le operazioni di CRUD al database per garantire i dati corretti e necessari.

Interfaccia fornita - Create/Update/Delete: vengono effettuate le operazioni di CRUD con il database per garantire i dati corretti e necessari.

Interfaccia fornita - Update Program/Card data: le informazioni da gestione programmi/schede vengono aggiornate nel front-end.

Create Program/Card Page

Descrizione: il componente svolge le funzionalità principali di front-end della pagina di creazione di programmi/schede. In particolare sono presenti tutti i form e moduli che permettono la creazione di nuovi programmi/schede e i collegamenti alla gestione di essi per portare a termine le operazioni di creazione.

Interfaccia richiesta - Infos: vengono inserite le informazioni necessarie per permettere il corretto funzionamento e sequenza del componente.

Interfaccia fornita - Add card/program: : le informazioni necessarie alla creazione di un nuovo programma vengono passate a gestione per portare al termine l'operazione.

Profile Page

Descrizione: il componente svolge le funzionalità di front-end della pagina del profilo. Qui è possibile vedere tutte le proprie informazioni e ci sono i collegamenti che permettono la modifica delle informazioni personali e cambio password.

Interfaccia richiesta - Profile data: dall'homepage vengono fornite le informazioni necessarie al front-end della pagina profilo.

Interfaccia richiesta - Update profile's data: le informazioni da gestione profilo vengono aggiornate nel front-end.

Profile Management

Descrizione: il componente svolge le funzionalità di back-end della pagina del profilo. Qui è possibile modificare le informazioni del proprio profilo e la password, e in particolare ci sono i collegamenti alle impostazioni di CRUD con il database.

Interfaccia richiesta - Infos: vengono inserite le informazioni necessarie per permettere il corretto funzionamento e sequenza del componente.

Interfaccia richiesta - New password: vengono inserite le informazioni necessarie per permettere l'aggiornamento della password.

Interfaccia richiesta - Create/Update/Delete function: vengono richieste le operazioni di CRUD al database per garantire i dati corretti e necessari.

Interfaccia fornita - Create/Update/Delete: vengono effettuate le operazioni di CRUD con il database per garantire i dati corretti e necessari.

Interfaccia fornita - Update profile's data: le informazioni da gestione profilo vengono aggiornate nel front-end.

Diary Page

Descrizione: il componente svolge le funzionalità di front-end della pagina del del diario. Qui è possibile visualizzare tutte le schede del diario ovvero gli allenamenti passati, sono presenti i collegamenti che permettono di commentare e dare un feedback alla scheda.

Interfaccia richiesta - Diary data: all'homepage vengono fornite le informazioni necessarie al front-end della pagina diario.

Interfaccia richiesta - Update diary data: le informazioni da gestione diario vengono aggiornate nel front-end.

Diary Management

Descrizione: il componente svolge le funzionalità di back-end della pagina del del diario. Qui avviene la parte di gestione delle schede del diario, ovvero il commento e il feedback delle singole attività e in particolare ci sono i collegamenti alle impostazioni di CRUD con il database.

Interfaccia richiesta - Comment: viene inserito un commento del singolo allenamento per tenere traccia delle sensazioni dell'utente

Interfaccia richiesta - Feedback: viene dato un feedback al singolo allenamento per avere un resoconto più immediato sugli allenamenti

Interfaccia richiesta - Create/Update/Delete function: vengono richieste le operazioni di CRUD al database per garantire i dati corretti e necessari.

Interfaccia fornita - Create/Update/Delete: vengono effettuate le operazioni di CRUD con il database per garantire i dati corretti e necessari.

Interfaccia fornita - Update diary's data: le informazioni da gestione diario vengono aggiornate nel front-end.

Database

Descrizione: il componente svolge tutte le funzionalità necessarie al database, in particolare gestione e salvataggio dei dati, nonché ha tutti i collegamenti alla funzioni di CRUD con le altre componenti.

Interfaccia richiesta - Registration: vengono gestite le operazioni di CRUD in seguito alla registrazione di un nuovo utente

Interfaccia richiesta - Update data: il database esegue le funzioni di CRUD per aggiornare i dati del componente

Interfaccia fornita - Create/Update/Delete function: il database effettua le operazioni di CRUD con i componenti a cui si interfaccia

Interfaccia fornita - Check authentication: il database fornisce il controllo delle credenziali d'accesso inserito dall'utente per l'autenticazione

Email Server

Descrizione: il componente svolge tutte le funzionalità di back-end che riguardano lo scheduling e l'invio di email di notifica, sia nei casi previsti da autenticazione/registrazione e sia da quelli del calendario

Interfaccia richiesta - Send email: vengono ricevute le informazioni necessarie per inviare le notifiche specifiche all'utente via email

Interfaccia richiesta - Email: vengono mandate le notifiche specifiche all'utente via email

Livello di accoppiamento dei componenti

Misura del grado di accoppiamento tra i moduli : Livelli di Coupling

1. **Content** (un modulo fa diretto riferimento al contenuto di un altro modulo)
2. **Common** (due moduli che accedono alla stessa struttura dati
3. **Control** (un modulo controlla esplicitamente l'esecuzione di un altro modulo)
4. **Stamp** (due moduli che si passano come argomento una struttura dati, della quale si usano solo alcuni elementi)
5. **Data** (due moduli che si passano argomenti omogenei, ovvero argomenti semplici o strutture dati delle quali si usano tutti gli elementi)

Accoppiamento Componenti	Login Page	Authentication/Registration System	Registration Page	Calendar Homepage	Email Server	Program/Card Page	Create Program/Card Page	Program/Card Management	Course Page	Create Course Page	Course Management	Diary Page	Diary Management	Profile Page	Profile Management	Database
Login Page		**5**	/	/	/	/	/	/	/	/	/	/	/	/	/	/
Authentication/Registration System			**5**	**5**	**3**	/	/	/	/	/	/	/	/	/	/	**4**
Registration Page				/	/	/	/	/	/	/	/	/	/	/	/	/
Calendar Homepage					**3**	**4**	/	/	**4**	/	/	**4**	/	**4**	/	**4**
Email Server						/	/	/	/	/	/	/	/	/	/	/
Program/Card Page							/	**4**	/	/	/	/	/	/	/	/
Create Program/Card Page								**5**	/	/	/	/	/	/	/	/
Program/Card Management									/	/	/	/	/	/	/	**4**
Course Page										/	**4**	/	/	/	/	/
Create Course Page											**5**	/	/	/	/	/
Course Management												/	/	/	/	**4**
Diary Page													**4**	/	/	/
Diary Management														/	/	**4**
Profile Page															**4**	/
Profile Management																**4**
Database																

Di seguito vengono spiegati i livelli di accoppiamento tra i vari componenti:

Login Page - Authentication/Registration System : Livello Data
Spiegazione: i moduli si scambiano i dati riguardo la sessione utente come email, password per effettuarne l'autenticazione.

Authentication/Registration System - Registration Page : Livello Data
Spiegazione: i moduli si scambiano i dati richiesti all'utente per effettuare la registrazione nel sistema.

Authentication/Registration System - Calendar Homepage : Livello Data
Spiegazione: una volta che l'utente è autenticato i moduli si scambiano i dati che sono stati recuperati dal database cosi' che vengano mostrati nella homepage dell'applicazione.

Authentication/Registration System - Email Server : Livello Control
Spiegazione: il componente Authentication/Registration system controlla l'invio delle email di notifica all'utente ad avvenuta registrazione o in caso di dimenticanza delle credenziali.

Authentication/Registration System - Database : Livello Stamp
Spiegazione: il componente Authentication/Registration system passa parte della sua struttura dati al database.

Calendar Homepage - Email Server : Livello Control
Spiegazione: il componente Calendar Homepage controlla l'invio delle email di notifica all'utente per l'avviso dell'allenamento in programma o delle modifiche avvenute.

Calendar Homepage - Program/Card Page : Livello Stamp
Spiegazione: il componente Calendar Homepage passa la parte della struttura dati che riguarda i programmi e le schede.

Calendar Homepage - Course Page : Livello Stamp
Spiegazione: il componente Calendar Homepage passa la parte della struttura dati che riguarda i corsi.

Calendar Homepage - Diary Page : Livello Stamp
Spiegazione: il componente Calendar Homepage passa la parte della struttura dati che riguarda il diario.

Calendar Homepage - Profile Page : Livello Stamp
Spiegazione: il componente Calendar Homepage passa la parte della struttura dati che riguarda il profilo.

Calendar Homepage - Database : Livello Stamp
Spiegazione: il componente Calendar Homepage passa la parte della struttura dati al Database.

Program/Card Page - Program/Card Management : Livello Stamp
Spiegazione: il componente Program/Card Page passa parte della struttura dati alla componente Program/Card Management.

Create Program/Card Page - Program/Card Management : Livello Data
Spiegazione: il componente create Program/Card Page passa l'intera struttura dati alla componente di gestione (Program/Card Management).

Program/Card Management - Database : Livello Stamp
Spiegazione: il componente Program/Card Management passa parte della struttura dati al Database.

Course Page - Course Management : Livello Stamp
Spiegazione: il componente Course Page passa parte della struttura dati alla componente di gestione Course Management.

Create Course Page - Course Management : Livello Data
Spiegazione: il componente Create Course Page passa l'intera struttura dati alla componente di gestione (Course Management).

Course Management - Database : Livello Stamp
Spiegazione: il componente Course Management passa parte della struttura dati al Database.

Diary Page - Diary Management : Livello Stamp
Spiegazione: il componente Diary Page passa parte della struttura dati alla componente gestione Diary Management.

Diary Management - Database : Livello Stamp
Spiegazione: il componente Diary Management passa parte della struttura dati al Database.

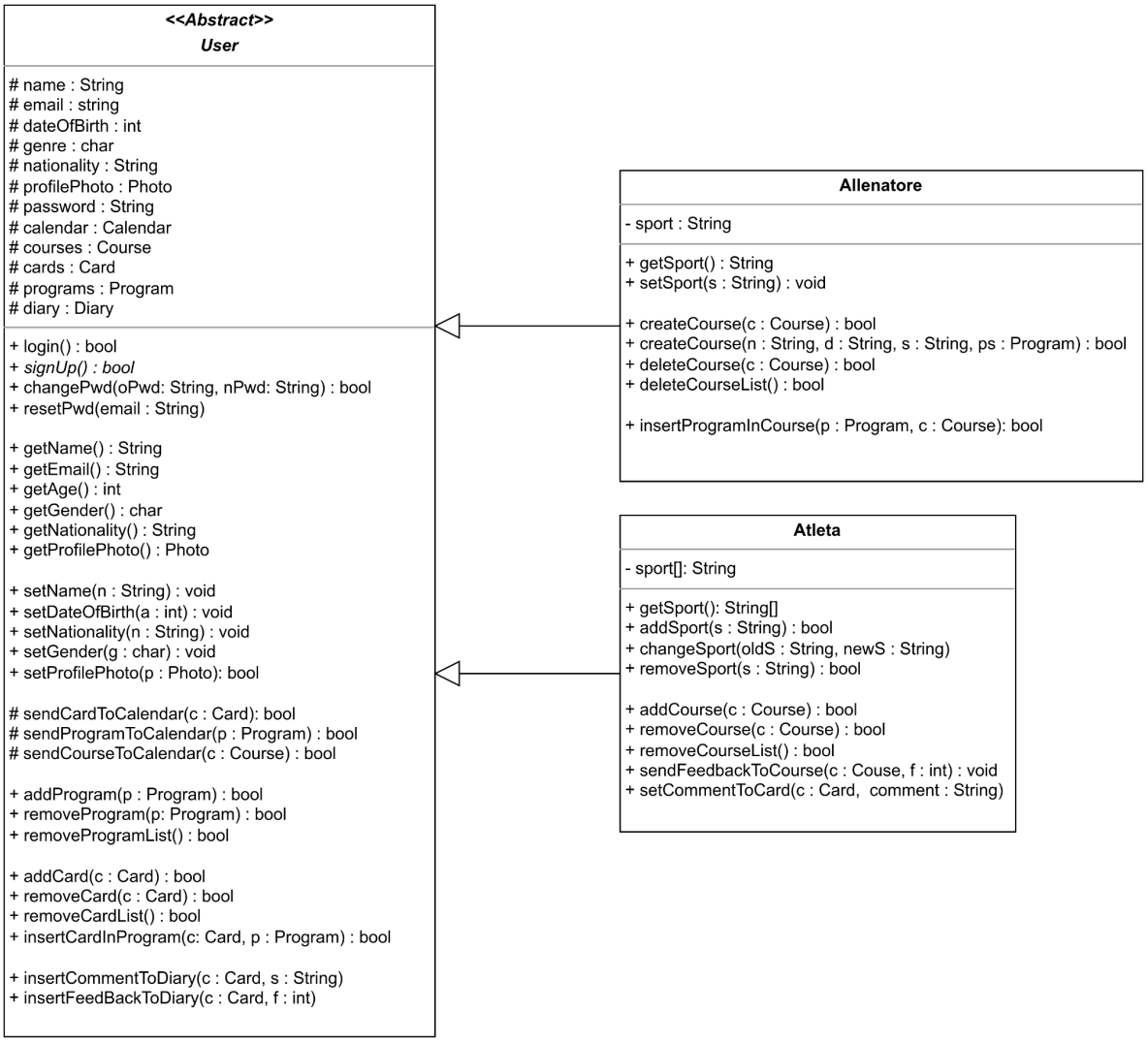
Profile Page - Profile Management : Livello Stamp
Spiegazione: il componente Profile Page passa parte della struttura dati alla componente di gestione Profile Management.

Profile Management - Database : Livello Stamp
Spiegazione: il componente Profile Management passa parte della struttura dati al Database.

CLASS DIAGRAM

Nel presente capitolo vengono presentate le classi previste nell'ambito del progetto MyTrainy. Ogni attore e ogni sistema esterno presenti nel diagramma di contesto diventano una o più classi. Ogni componente presente nel diagramma dei componenti diventa una o più classi. Tutte le classi individuate sono caratterizzate da un nome, una lista di attributi che identificano i dati gestiti all'interno della classe. Ogni classe può essere anche associata ad altre classi e, tramite questa associazione, è possibile fornire informazioni su come le classi si relazionano tra di loro.
Riportiamo in seguito le classi individuate a partire dai diagrammi di contesto e dei componenti. In questo processo si è proceduto anche nel massimizzare la coesione e minimizzare l'accoppiamento tra classi.

Utenti



Il diagramma in figura è costituito da tre classi: User, Allenatore e Atleta.
La classe User è astratta e contiene la maggior parte degli attributi e delle operazioni, tra cui:

- L'autenticazione, registrazione e recupero password

- Gestione del profilo (nome, nazionalità, genere, foto profilo, data di nascita, password)
- Gestione delle schede e dei programmi che l'utente crea

La gestione dei corsi verrà implementata dalle sue sottoclassi, in quanto eseguono operazioni diverse se allenatore o atleta (vedi i deliverable precedenti).
Il metodo signUp() è astratto e verrà implementato nelle sottoclassi
Riportiamo una prima documentazione delle tre classi appena descritte:

User:

Attributi	Metodi	
1. **name**: user name 2. **email** : user email address 3. **dateOfBirth**: user age 4. **genre** : user genre (M, F, ND) 5. **nationality** : user nationality 6. **profilePhoto** : photo that user inserts in his profile 7. **password**: user password 8. **calendar**: user calendar 9. **courses**: attribute that it'll be implemented in subclasses 10. **cards**: list of Cards created by user 11. **programs**: list of Programs created by user 12. **diary**: user's diary	• login(): it's used to authenticate the user • signUp(): It's used to register the user. It's an abstract method • changePwd(oPwd : String, newPwd : String): It's used to change user password • resetPwd(email : String): It's used to reset the password (email required) • getName(): It returns user name • getEmail(): It returns user email address • getAge(): It returns user age • getGender(): It returns user genre • getNationality(): It returns user nationality • getProfilePhoto(): It returns user photo profile • setName(n : String): It set or modify user name • setDateOfBirth(a : int) : it set or modify user date of birth • setNationality(n : String): it set or modify user nationality • setGender(g : char): it set or modify user genre	• setProfilePhoto(p : Photo): it set user profile photo • sendCardToCalendar(c : Card): send a card in calendar • sendProgramToCalendar(p : Program): send a list of card in calendar • sendCourseToCalendar(c : Course): send a list of program in calendar • addProgram(p : Program): it adds an user Program • removeProgram(p : Program): it removes an user Program • removeProgramList(): it removes all user Programs • addCard(c : Card): it adds an user Card • removeCard(c : Card): it removes an user Card • removeCardList(): it removes all user Card • insertCardInProgram(c : Card, p : Program): it inserts a Card in a Program • insertCommentToDiary(c : Card, s : String) • insertFeedBackToDiary(c : Card, f : int)

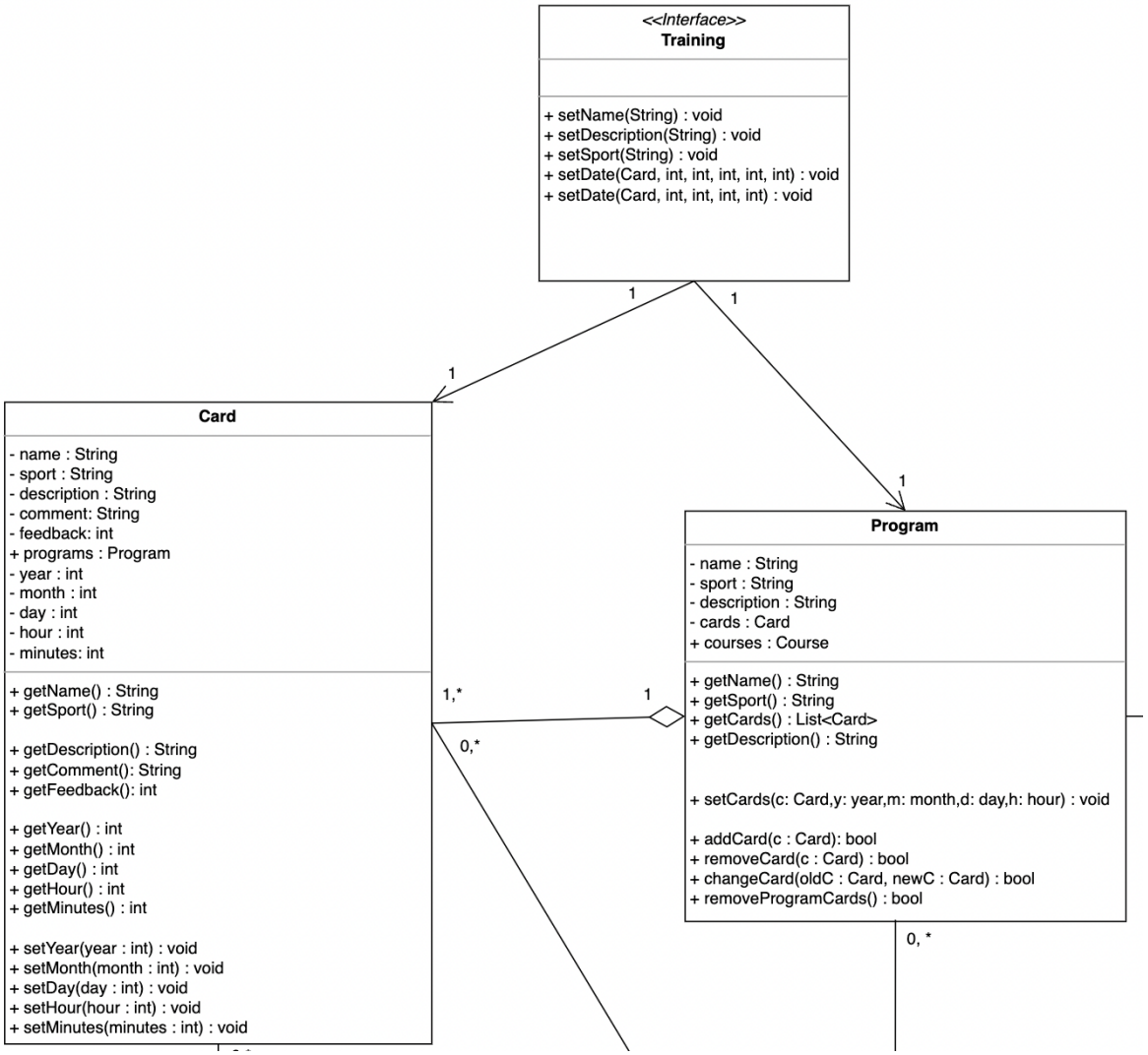
Atleta:

Attributi	Metodi
• **sport[]**: it's an array of strings which contain the ATL favorite sports	• getSports(): it returns an array of string which contains all the sport • addSport(s : String): It adds a sport • changeSport(oldS : String, newS : String): it replaces a sport selected with another one • removeSport(s : String): it removes the sport selected • addCourse(c : Course): it adds a Course in Course List • removeCourse(c : Course): it removes a course in Course List • removeCourseList(): it removes all courses in Course List • sendFeedbackToCourse(c : Couse, f : int): it send the feedback to the course • setCommentToCard(c : Card, comment : String): it send the comment to the diary

Allenatore:

Attributi	Metodi
• **sport** : it's a string that indicates the ALL sport	• getSport(): it returns ALL sport • setSport(): it set or modify the ALL sport • createCourse(c : Course): it creates a new Course and it'll be contained in Course list • deleteCourse(c : Course): it deletes a Course that ALL created • deleteCourseList(): it deletes all Course list that ALL created • insertProgramInCourse(p : Program, c : Course): it inserts a Program in a Course

I blocchi base: Card, Program e Training



Il diagramma in figura è costituito da due classi (Card e Program) ed un'interfaccia (Training).
La classe Card è il "blocco" base per ogni allenamento, ed è costituita da:

- Un nome
- Lo sport a cui è riferita
- Una descrizione (al fine di spiegare a cosa sono mirati gli esercizi)
- Un commento, che inserirà lo User dopo aver svolto l'allenamento
- Un feedback, utilizzato come indice per capire la validità o efficacia dell'allenamento
- Una serie di parametri utilizzati per sapere quando verranno svolti gli allenamenti

La classe Program è una collezione di Card che consente di pianificare degli allenamenti.
L'interfaccia Training è costituita da quattro operazioni che verranno implementati sia dalla classe Card che Program
Riportiamo una prima documentazione delle tre classi appena descritte:

Card:

Attributi	Metodi	
• **name**: Card's name • **sport**: sport referred • **description**: description of the Card • **comment**: string • **feedback**: int • **year** : • **month** : • **day** : • **hour** : • **minutes** :	• getName(): it returns the Card name • getSport(): it returns the sport to which the Card is referred to • getDescription(): it returns the Card description • getComment(): • getFeedback(): • getYear(): it returns the year to do the Card • getMonth(): it return the month to do the Card • getDay() : it return the day to do the Card • getHour() : it return the hour to do the Card	• getMinutes(): it return the minutes to do the Card • setName(n : String): it allows to set or modify the Card name • setSport(s : String): it allows to set or modify the sport selected • setYear(): it allows to set or modify the year • setMonth(): it allows to set or modify the month • setDay() : it allows to set or modify the day • setHour() : it allows to set or modify the hour • setMinutes(): it allows to set or modify the minutes

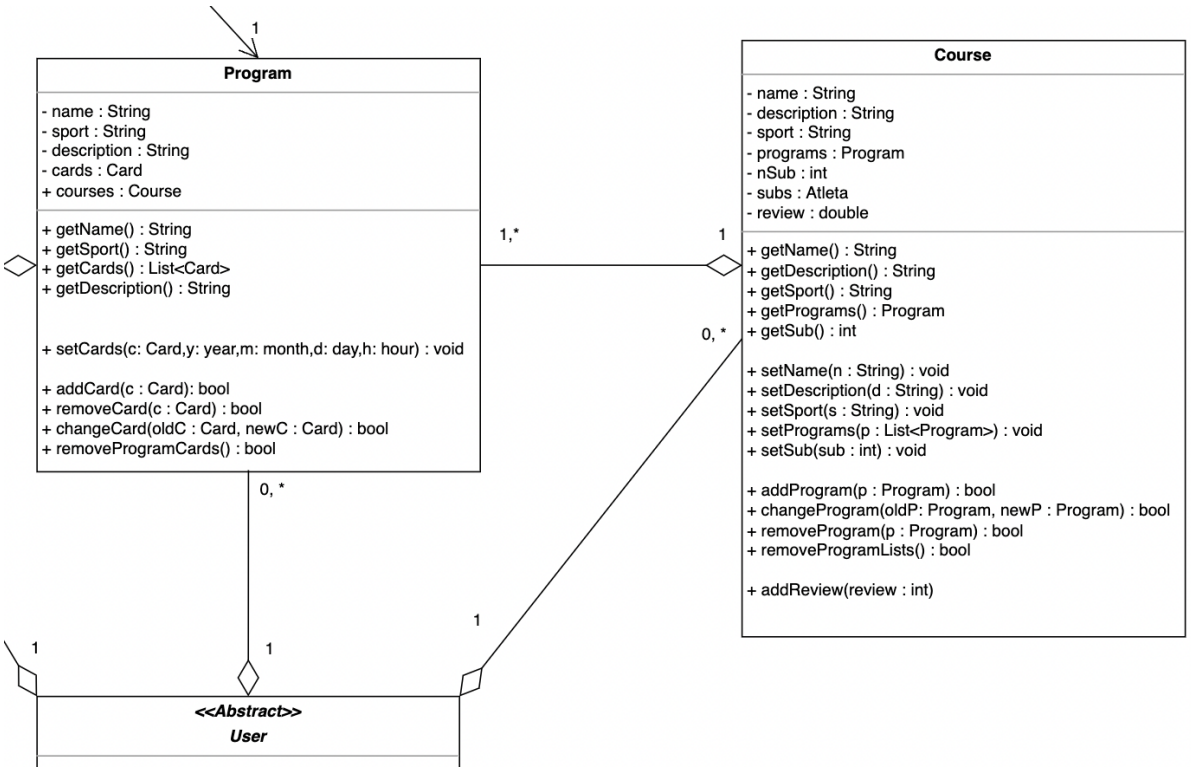
Program:

Attributi	Metodi
• **name**: Program name • **sport**: which sport the Program is referred to • **description**: Program description • **cards**: list of Cards contained in the Program	• getName(): it returns the Card name • getSport(): it returns the sport to which the Program is referred to • getCards(): it returns all the Program cards • getDescription(): it returns the Program description • setCards(p :Pair<Card, Date>): it allows to set or modify completely the Card list • addCard(c : Card): it allows to add a Card in the Program • removeCard(c : Card): it allows to remove a Card contained in the Program • changeCard(oldC : Card, newC : Card): it allows to modify a Card contained in the Program to another one • removeProgramCards(): it removes all Cards contained in the Program

Training:

Metodi
• setName(String) • setDescription(String) • setSport(String) • setDate(Card, int, int, int, int, int) • setDate(Card, int, int, int, int)

Course: la condivisione di Program



Come menzionato negli scorsi deliverable, L'allenatore può creare dei corsi costituiti da programmi in modo che gli atleti possano iscriversi e seguire le schede da lui create.
Come possiamo notare dalla figura, la classe Course è costituita da:

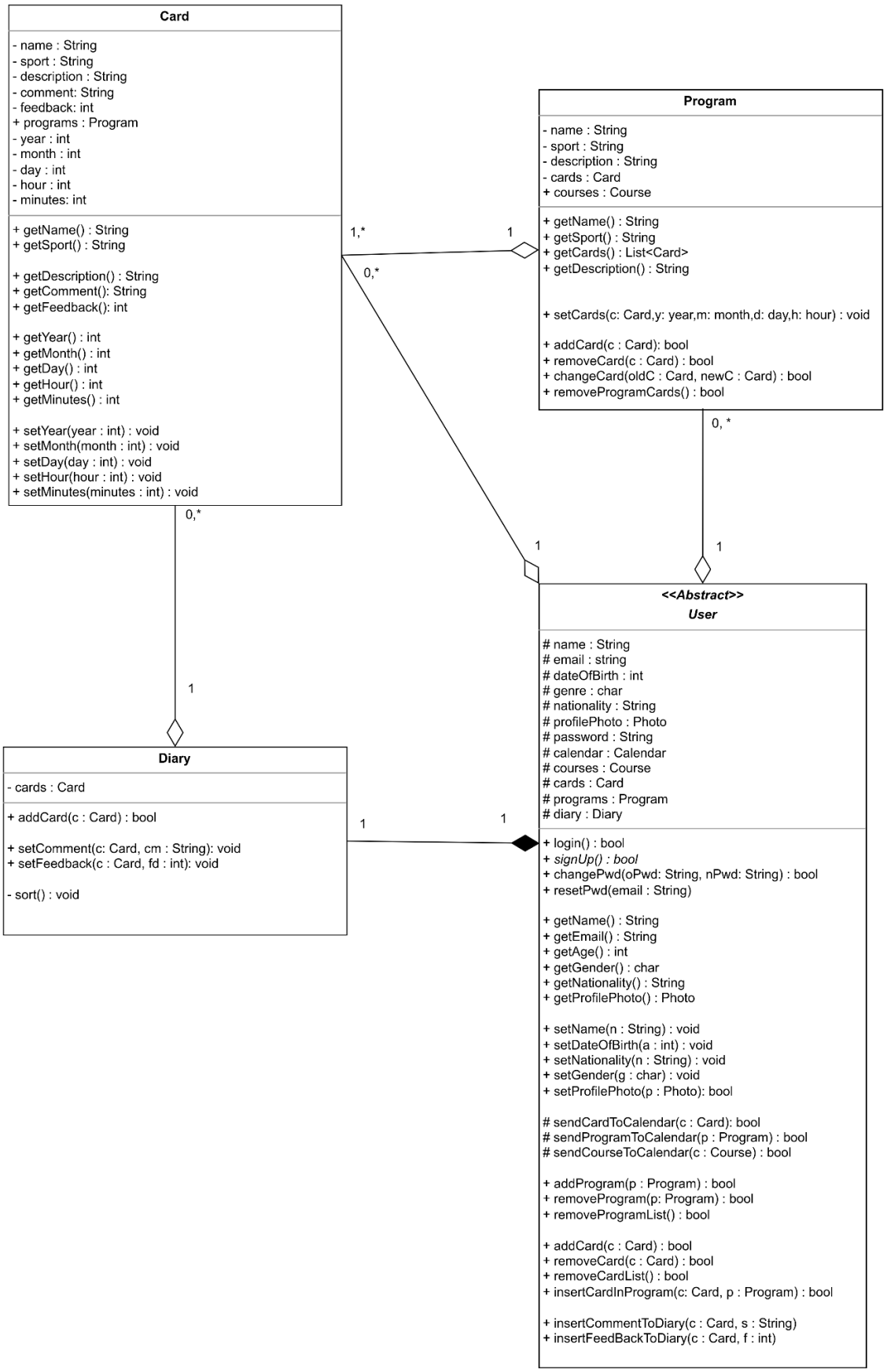
- Il nome del corso
- Lo sport a cui è rivolto il corso
- Una descrizione del corso (come ad esempio a che tipo di persone sono mirati gli allenamenti, che tipologia di esercizi contiene ecc.)
- Ed una lista di programmi: per rendere il corso una struttura versatile sia per l'allenatore (che magari deve fare diverse tipologie di allenamento) che per gli atleti, che magari sono interessati solo ad una specifica parte

Riportiamo una prima documentazione della classe Course:

Course:

Attributi	Metodi
• **description** : description of the Course • **sport** : which sport the Course is referring to • **programs**: list of Course Programs	• getName() : it returns the Course name • getDescription() : it returns the Course description • getSport(): it returns the sport referred • getPrograms(): it returns the Programs list • setName(n : String): it allows to set or modify the Program name • setDescription(d : String): it allows to set or modify the Program description • setSport(s : String): it allows to set or modify the sport • setPrograms(p : List<Program>): it allows to set or modify the Programs list • addProgram(p : Program): add a new Program in the course • changeProgram(oldP : Program, newP: Program): it replaces a program with another one • removeProgram(p : Program): it removes a program • removeProgramList(): it removes the Program list • addReview(review : int): add review to che course

Diary: annotare gli allenamenti svolti

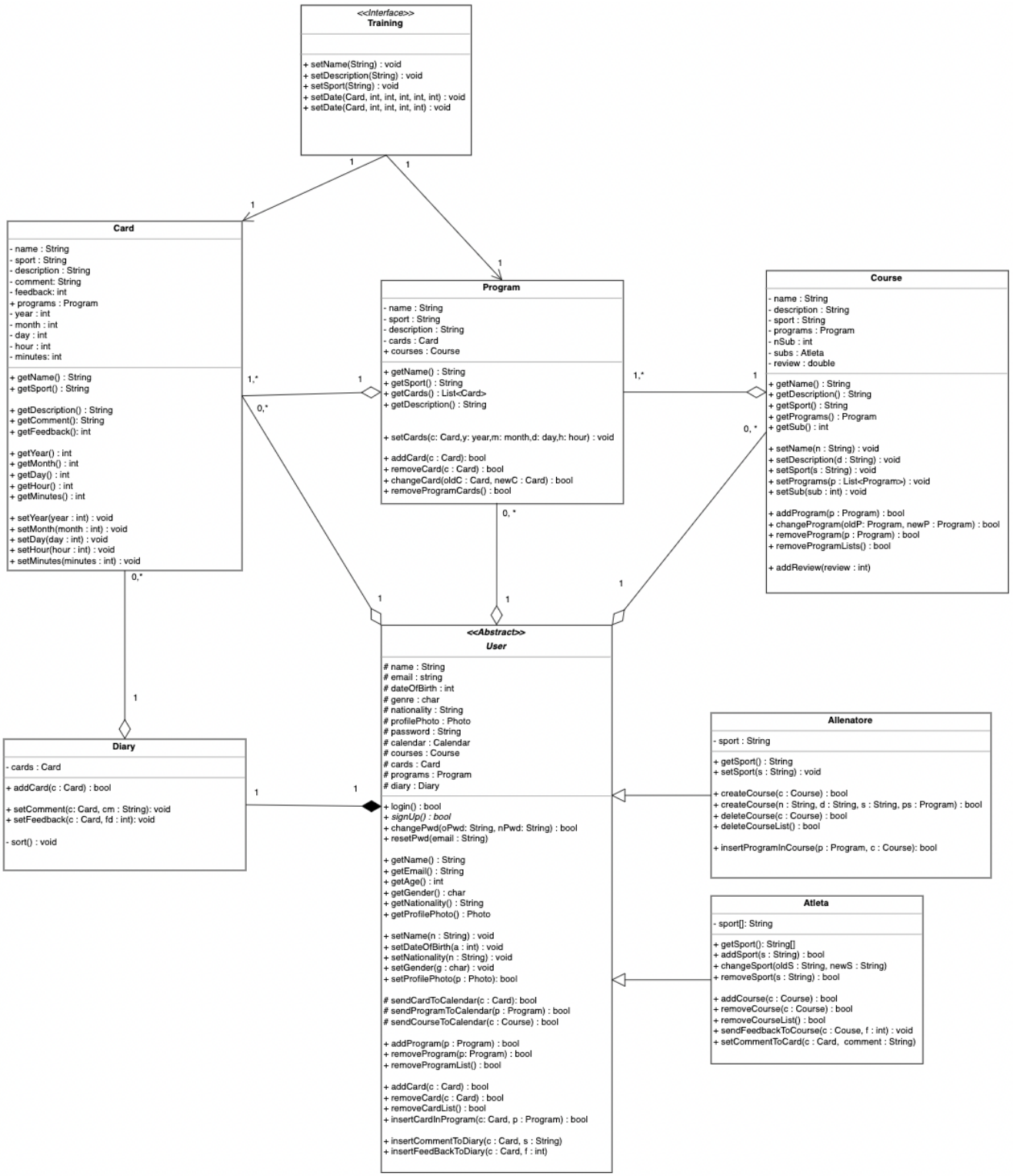


L'ultima classe di MyTrainy è Diary, che, come già detto nei deliverable precedenti, ha lo scopo di raccogliere le informazioni degli allenamenti svolti per notare il progresso dell'atleta. La classe Diary è una collezione delle Card dell'utente **che sono già state svolte**; l'atleta, quando finisce un allenamento, può scrivere un commento sull'attività svolta. Se la Card svolta appartiene ad un corso, l'allenatore invia successivamente un feedback all'atleta per segnalare come ha svolto gli esercizi ed incentivarlo nel percorso di allenamento.
Riportiamo una prima documentazione della classe Diary:

Diary:

Attributi	Metodi
• cards : list of Cards created by user that has data not NULL	• addCard(c : Card) : it'll be automatically invoked when user create a new user Card and it's set for a day • setComment(c : Card, cm : String): • setFeedBack(c : Card, fd : int): • sort(): it's an internal method used to sort (by date) the Cards list

Class Diagram

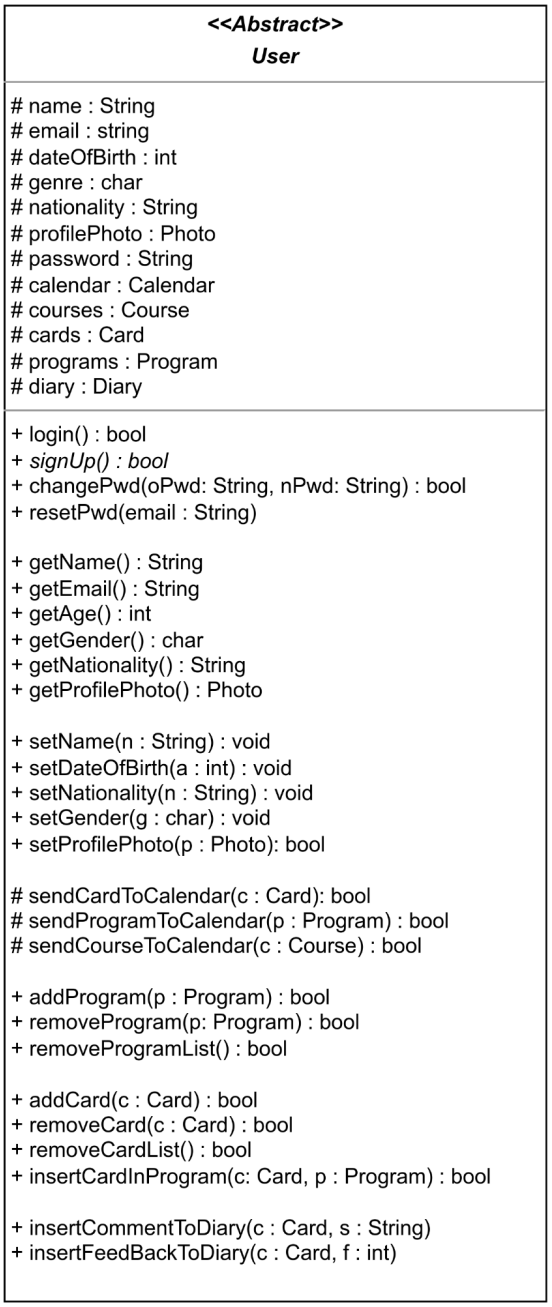


CODICE IN OBJECT CONSTRAINT LANGUAGE

In questo capitolo è descritta in modo formale la logica prevista nell'ambito di alcune operazioni di alcune classi. Tale logica viene descritta in OCL perché tali concetti non sono esprimibili in nessun altro modo formale nel contesto di UML.

Login e Registrazione

Le funzionalità permesse dal servizio all'utente sono permesse previa autenticazione e una corretta registrazione all'applicazione. Questa condizione su questa classe:

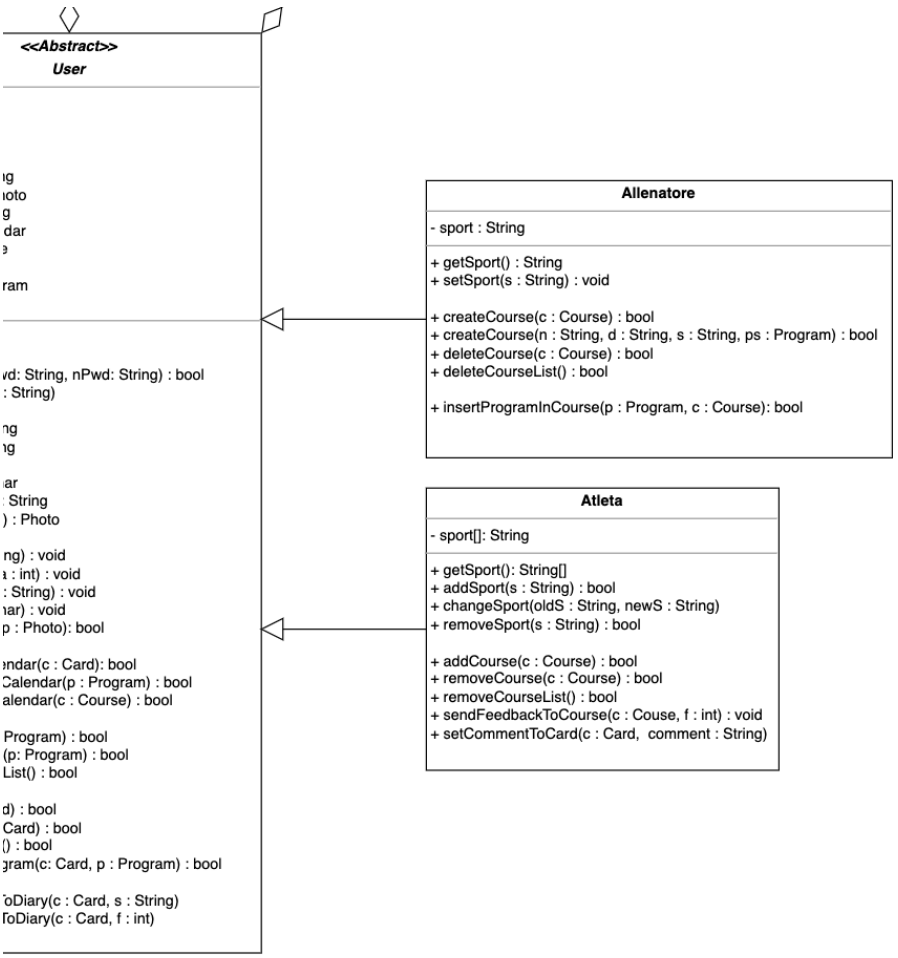


è espressa in OCL attraverso una precondizione con questo codice:

```
context User inv: login() = true AND singUp() = true
```

Utente di tipo allenatore

Nel caso l'utente si fosse registrato come allenatore, ha a disposizione delle funzionalità aggiuntive nel caso di gestione corsi, in particolare può aggiungere modificare e cancellare programmi e schede a partire dal corso che organizza. Questa condizione su questa classe:



è espressa in OCL attraverso una precondizione con questo codice:

```
Context Allenatore::addProgram()
POST programs[i].courses[i] <> NULL
```

```
Context Allenatore::addCard()
POST cards[i].programs[i].courses[i] <> NULL
```

CLASS DIAGRAM CON CODICE OCL

Riportiamo infine il diagramma delle classi con tutte le classi fino ad ora presentate ed il codice OCL individuato.