

# *StudyHub*



Studenti  
Danilo Verde 1000069238  
Nicolò Mazzola 1000067652

## ***Prefazione***

Il presente documento offre una panoramica dettagliata dello sviluppo dell'applicazione *StudyHub*, realizzata nell'ambito del corso di Ingegneria del Software.

L'implementazione è stata condotta utilizzando il linguaggio Java all'interno dell'ambiente di sviluppo Visual Studio Code. La progettazione è stata supportata dall'uso di *Asta*, con l'intento di presentare esclusivamente le versioni finali di ciascuna componente, elaborate al termine di tutte le fasi progettuali. Eventuali versioni intermedie sono disponibili all'interno delle rispettive sottocartelle della documentazione.

Il documento approfondisce inoltre le fasi finali del processo di sviluppo, con particolare attenzione al testing e al refactoring del codice. Queste attività hanno permesso di migliorare significativamente la qualità complessiva dell'applicazione, ottimizzandone efficienza e manutenibilità.

# Indice

1. <i>Introduzione</i> .....	4
2. <i>Obiettivi e requisiti</i> .....	4
3. <i>Obiettivi dei casi d'uso</i> .....	5
4. <i>Casi d'uso</i> .....	5
5. <i>Regole di business</i> .....	12
6. <i>Specifiche supplementari</i> .....	13
7. <i>Glossario</i> .....	13
8. <i>Analisi orientata agli oggetti</i> .....	13
9. <i>Modello di dominio</i> .....	14
10. <i>SSD e contratti delle operazioni</i> .....	20

# **1. Introduzione**

L'obiettivo del progetto è quello di realizzare un software che rivoluzioni l'esperienza accademica degli studenti. StudyHub rappresenta un ecosistema completo, unificando una serie di strumenti essenziali per la gestione accademica, la collaborazione e il potenziamento delle abilità di studio tramite strumenti come un calendario accademico intuitivo, un sistema di gestione delle attività e dei compiti, la possibilità di condividere, scaricare e commentare appunti per favorire la condivisione delle conoscenze.

## **1.1. Scopo**

Il presente documento ha l'obiettivo di fornire una chiara visione del progetto StudyHub, delinearne gli obiettivi principali e mettere in evidenza le funzionalità chiave attraverso la dettagliata esposizione delle caratteristiche dell'applicazione, delle sue potenzialità e dei benefici offerti agli utenti, e per consolidare una comprensione comune tra gli stakeholder riguardo alle finalità e agli aspetti distintivi di StudyHub.

## **1.2. Portata**

Il presente documento è dedicato al progetto StudyHub, realizzato dagli studenti Nicolò Mazzola e Danilo Verde. L'obiettivo primario è quello di sviluppare un applicativo che semplifichi l'organizzazione giornaliera della vita studentesca tramite una piattaforma flessibile e adattabile utilizzabile in diversi dispositivi e in diversi sistemi operativi con una principale enfasi sulla semplicità d'utilizzo.

# **2. Obiettivi e requisiti**

### **- Gestione Utenti**

- Registrazione e login con credenziali personali.
- Modifica del profilo utente con dati anagrafici e livelli di studio.
- Memorizzazione sicura delle credenziali.
- Eliminazione del profilo.

### **- Gestione Corsi**

- Creazione di corsi con specifiche dettagliate (nome, livello, lingua, costo, durata).
- Ricerca ed iscrizione ai corsi con un sistema di pagamento integrato.
- Gestione e visualizzazione dei contenuti dei corsi (appunti, materiali didattici, iscritti al corso).
- Eliminazione dei corsi creati.

### **- Gestione Gruppi di Studio**

- Creazione di gruppi studio con parametri personalizzabili (lingua, durata, admin).
- Iscrizione ai gruppi studio con sistema di autenticazione tramite password.
- Gestione delle iscrizioni e degli studenti nei gruppi studio.
- Eliminazione dei gruppi studio creati.

### **- Gestione dei materiali didattici**

- Caricamento di appunti e materiali in diversi formati.
- Associazione dei contenuti di corsi.
- Visualizzazione e gestione degli appunti.
- Visualizzazione e gestione dei contenuti.

- Eliminazione di contenuti personali e condivisi.

### **3. Obiettivi dei casi d'uso**

<b>Attore principale</b>	<b>Obiettivo</b>	<b>Caso d'uso</b>
Studente	Registrarsi alla piattaforma e creare il proprio profilo inserendo i propri dati. Possibilità di eliminazione del profilo.	<b>UC1: Iscrizione profilo</b>
Studente	Modificare il proprio profilo cambiando i dati inseriti.	<b>UC2: Modifica profilo.</b>
Studente	Creare corsi a cui altri studenti possono iscriversi. Possibilità di eliminazione di un corso proprio.	<b>UC3: Creazione di un corso</b>
Studente	Iscriversi ai corsi di studio già creati da altri studenti, effettuando il pagamento se richiesto. Possibilità di disiscriversi da un corso.	<b>UC4: Iscrizione ad un corso.</b>
Studente	Creare un gruppo di studio, ovvero un gruppo di studenti che possono studiare insieme materie comuni. Possibilità di eliminare un gruppo studio proprio.	<b>UC5: Creazione di un gruppo studio</b>
Studente	Iscriversi ad un gruppo studio già creato da altri studenti. Disiscriversi da un gruppo studio.	<b>UC6: Iscrizione ad un gruppo studio</b>
Studente	Caricare contenuti in un corso che si è creato, da mettere a disposizione degli iscritti al corso. Eliminazione di un contenuto proprio da un corso.	<b>UC7: Caricamento di contenuti</b>
Studente	Caricamento appunti personali, condivisibili con altri studenti. Eliminazione di un appunto proprio.	<b>UC8: Caricamento di appunti</b>
Studente	Download appunti/contenuti da gruppi di studio/corsi. Eliminazione di appunti/contenuti propri.	<b>UC9: scarica contenuto/appunto da corso/gruppo studio.</b>
Studente	Visualizzazione dati studente: corsi, gruppi, appunti, contenuti e modifica appunti.	<b>UC10: visualizza dati studente.</b>

### **4. Casi d'uso**

**UC1: Iscrizione profilo.**

<b>Nome</b>	UC1: iscrizione profilo
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	<p>Studente alunno: vuole avere la possibilità di iscriversi nel sistema, in modo da poter usufruire dei corsi, dei gruppi studio e dei materiali didattici degli altri iscritti.</p> <p>Studente docente: vuole iscriversi nel sistema per potere fornire corsi e materiale didattico specifico agli altri studenti presenti nel corso.</p>
<b>Pre-condizioni</b>	Lo studente non è registrato nel sistema.
<b>Garanzia di successo</b>	Lo studente è autenticato nel sistema e può utilizzare le diverse funzioni del sistema stesso.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente accede a StudyHub e seleziona “Registrazione utente”.</li> <li>2. In fase di registrazione, l'utente inserisce i dati relativi a sé stesso.</li> <li>3. Lo studente completa la registrazione e può usufruire delle funzioni del sistema.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. Le credenziali dell'utente potrebbero essere già in uso.</li> <li>2. L'utente sceglie di eliminare il profilo anziché crearlo, eliminando prima tutte le sue iscrizioni, contenuti dei corsi e iscrizioni ai gruppi studio.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per la registrazione. Poche credenziali uniche richieste.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Una volta per ogni ciclo di vita dello studente nell'app.
<b>Varie</b>	

## ***UC2: Modifica profilo.***

<b>Nome</b>	UC2: modifica profilo
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente: vuole modificare le proprie credenziali all'interno del sistema.
<b>Pre-condizioni</b>	Lo studente è registrato nel sistema.
<b>Garanzia di successo</b>	Lo studente è ancora autenticato nel sistema e può utilizzare le diverse funzioni del sistema stesso.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente accede a StudyHub e seleziona “Modifica profilo”.</li> <li>2. L'utente inserisce i nuovi dati relativi al suo profilo.</li> <li>3. I dati nel profilo vengono modificati e registrati nel sistema.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. Le nuove credenziali dell'utente non sono valide o sono già in uso.</li> </ol>

<b>Requisiti speciali</b>	Interfaccia intuitiva per la modifica. Poche credenziali uniche e differenti dalle precedenti richieste.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Poche volte al mese.

### **UC3: Creazione di un corso**

<b>Nome</b>	UC3: creazione di un corso
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente docente: vuole creare un corso per fornire accesso agli altri studenti al proprio materiale di studio relativo al corso stesso.
<b>Pre-condizioni</b>	Lo studente è autenticato nel sistema. Il corso non deve esistere.
<b>Garanzia di successo</b>	Lo studente torna al menù principale. Il sistema registra il corso nel sistema.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Lo studente accede a StudyHub e seleziona “Crea corso”.</li> <li>2. Lo studente inserisce i dati del corso.</li> <li>3. Il sistema registra il corso all'interno del sistema e riporta lo studente al menù principale con un messaggio di conferma.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a creare il corso.</li> <li>2. Il corso esiste già, la creazione viene annullata.</li> <li>3. L'utente sceglie l'eliminazione del corso anziché la creazione, eliminandone le iscrizioni e tutti i contenuti.</li> <li>4. I dati del corso sono già utilizzati.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per la creazione del corso.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Una/poche volte per ogni studente docente.
<b>Varie</b>	

### **UC4: ricerca ed iscrizione ad un corso**

<b>Nome</b>	UC4: ricerca ed iscrizione ad un corso
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente alunno: vuole avere la possibilità di iscriversi ad un corso di studio dal quale ricavare documenti di studio e lezioni.

	Studente docente: fornisce un corso a cui iscrversi (eventualmente dietro compenso) e desidera avere l'elenco degli studenti iscritti e condividere documenti e lezioni con loro
<i>Pre-condizioni</i>	Lo studente è autenticato nel sistema. Il corso deve esistere.
<i>Garanzia di successo</i>	Lo studente è iscritto al corso selezionato. Il sistema registra lo studente.
<i>Scenario principale di successo</i>	<ol style="list-style-type: none"> <li>1. L'utente accede a StudyHub e seleziona "Ricerca corso".</li> <li>2. Una volta inseriti i dati relativi al corso stesso, il sistema mostra i corsi disponibili sulla base delle informazioni inserite.</li> <li>3. Il cliente seleziona il corso e, se ha un costo, viene eseguito un pagamento, altrimenti viene direttamente iscritto.</li> <li>4. Il sistema registra lo studente inserito, sia lato "Studente" che lato "Corso".</li> </ol>
<i>Estensioni</i>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a iscriversi.</li> <li>2. Lo studente non ha dei dati di pagamento validi, quindi annulla l'iscrizione.</li> <li>3. L'utente è già iscritto ad un corso, quindi viene annullata l'iscrizione e l'utente torna indietro alla scheda iniziale.</li> <li>4. L'utente seleziona la disiscrizione da un corso piuttosto che l'iscrizione, in modo da eliminarsi dallo stesso, compresi tutti i contenuti che ha eventualmente caricato nel corso.</li> </ol>
<i>Requisiti speciali</i>	Interfaccia intuitiva per la selezione e l'iscrizione. Corsi già presenti nel sistema.
<i>Elenco delle varianti tecnologiche e dei dati</i>	
<i>Frequenza di ripetizioni</i>	Poche volte al giorno.
<i>Varie</i>	

## ***UC5: Creazione gruppo studio***

<i>Nome</i>	UC5: creazione gruppo studio.
<i>Portata</i>	Gestione sistema di StudyHub
<i>Livello</i>	Obiettivo utente
<i>Attore primario</i>	Studente
<i>Parti interessate e interessi</i>	Studente alunno: vuole creare un gruppo studio per condividere gli appunti con altri studenti alunni.
<i>Pre-condizioni</i>	Lo studente è autenticato nel sistema. Lo studente non ha un gruppo studio con quelle specifiche credenziali.
<i>Garanzia di successo</i>	Il sistema registra il gruppo tra quelli dello studente.
<i>Scenario principale di successo</i>	<ol style="list-style-type: none"> <li>1. Lo studente accede a StudyHub e seleziona "Crea gruppo studio".</li> <li>2. Lo studente inserisce i dati del gruppo.</li> <li>3. Il sistema registra il gruppo nella lista dei gruppi dello studente.</li> </ol>

<i>Estensioni</i>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a creare il gruppo.</li> <li>2. Lo studente tenta di creare un gruppo con credenziali già in uso (gruppo già esistente), quindi torna al menù principale.</li> <li>3. Il nome del gruppo studio è già in uso, quindi annulla la creazione e torna al menù principale.</li> <li>4. Si sceglie l'eliminazione del gruppo studio, in modo da eliminare tutte le iscrizioni dello stesso gruppo.</li> </ol>
<i>Requisiti speciali</i>	Interfaccia intuitiva per la creazione del gruppo.
<i>Elenco delle varianti tecnologiche e dei dati</i>	
<i>Frequenza di ripetizioni</i>	Poche volte al giorno.
<i>Varie</i>	

### **UC6: Ricerca e iscrizione ad un gruppo studio**

<i>Nome</i>	UC6: ricerca ed iscrizione ad un gruppo studio
<i>Portata</i>	Gestione sistema di StudyHub
<i>Livello</i>	Obiettivo utente
<i>Attore primario</i>	Studente
<i>Parti interessate e interessi</i>	Studente alunno: vuole avere la possibilità di iscriversi ad un gruppo di studio dal quale ricavare materiale didattico e condivide il proprio con gli altri studenti.
<i>Pre-condizioni</i>	<p>Lo studente è autenticato nel sistema.</p> <p>Il gruppo deve esistere.</p> <p>Il gruppo deve avere numero partecipanti massimo non raggiunto o essere aperto ad un numero illimitato di iscrizioni.</p>
<i>Garanzia di successo</i>	<p>Lo studente è iscritto al gruppo selezionato.</p> <p>Il sistema registra lo studente tra gli studenti del corso.</p>
<i>Scenario principale di successo</i>	<ol style="list-style-type: none"> <li>1. L'utente accede a StudyHub e seleziona “Iscrizione ad un gruppo studio”.</li> <li>2. Una volta inseriti i dati relativi al gruppo stesso, il sistema verifica i dati inseriti e inserisce lo studente direttamente nel gruppo selezionato.</li> <li>3. Il sistema registra lo studente inserito, sia lato “Studente” che lato “Gruppo studio”.</li> </ol>
<i>Estensioni</i>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a iscriversi al gruppo.</li> <li>2. Lo studente inserisce delle credenziali del gruppo invalide, viene riportato al menù principale con un messaggio di errore specifico.</li> <li>3. Lo studente è già iscritto, viene riportato al menù principale con un messaggio di errore.</li> <li>4. Il gruppo risulta essere pieno, annullando l'iscrizione dello studente.</li> </ol>

	5. L'utente sceglie di eliminare il gruppo studio, eliminando tutte le iscrizioni degli studenti.
<b>Requisiti speciali</b>	Interfaccia intuitiva per la selezione e l'iscrizione. Gruppo già presente nel sistema.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Poche volte al giorno.
<b>Varie</b>	

### **UC7: caricamento contenuti in un corso**

<b>Nome</b>	UC7: caricamento contenuti di un corso
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente docente: vuole inserire i suoi contenuti in un corso per condividerli con gli altri studenti del corso.
<b>Pre-condizioni</b>	Lo studente è autenticato nel sistema. Il corso deve esistere. Lo studente è il creatore del corso.
<b>Garanzia di successo</b>	Lo studente torna al menù principale. Il sistema registra il contenuto nel corso.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Lo studente accede a StudyHub e seleziona “Carica contenuto”.</li> <li>2. Lo studente seleziona il corso al quale caricare il contenuto tra quelli che ha creato.</li> <li>3. Lo studente inserisce i dati del contenuto.</li> <li>4. Il sistema registra il contenuto all'interno del corso e riporta lo studente al menù principale con un messaggio di conferma.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a caricare i contenuti.</li> <li>2. Lo studente non ha dei corsi creati, quindi annulla il pagamento.</li> <li>3. Il contenuto non può essere caricato perché è già presente.</li> <li>4. Il contenuto non può essere caricato perché è troppo grande.</li> <li>5. L'utente decide di eliminare il contenuto dal corso.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per il caricamento del contenuto.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Poche volte al giorno.
<b>Varie</b>	

### **UC8: caricamento appunti**

<b>Nome</b>	UC8: caricamento appunti
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente

<b>Parti interessate e interessi</b>	Studente: vuole inserire gli appunti personali nel sistema per un'eventuale condivisione.
<b>Pre-condizioni</b>	Lo studente è autenticato nel sistema.
<b>Garanzia di successo</b>	Il sistema registra l'appunto tra quelli dello studente e nella lista di tutti gli appunti.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Lo studente accede a StudyHub e seleziona “Carica appunto”.</li> <li>2. Lo studente inserisce i dati dell'appunto.</li> <li>3. Il sistema registra l'appunto all'interno della lista degli appunti dello studente.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. In qualsiasi momento, il sistema fallisce: il sistema ricostruisce lo stato precedente, rileva le anomalie e mostra l'errore allo studente. Dopo la risoluzione, lo studente può ritentare a caricare l'appunto.</li> <li>2. L'appunto risulta essere già caricati nella lista appunti di studenti, quindi il caricamento viene annullato.</li> <li>3. L'utente sceglie l'eliminazione di un appunto dalla lista dei suoi appunti caricati.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per il caricamento degli appunti.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Poche volte al giorno.
<b>Varie</b>	

### **UC9: scarica contenuti/appunti da un corso/gruppo studio**

<b>Nome</b>	UC9: scarica contenuti/appunti da un corso/gruppo studio.
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente: vuole entrare in possesso di contenuti/appunti presenti in corsi/gruppi studio.
<b>Pre-condizioni</b>	<p>Lo studente è registrato nel sistema.            Lo studente deve essere iscritto ad un corso/gruppo studio per poter scaricare un appunto/contenuto.</p>
<b>Garanzia di successo</b>	Lo studente è autenticato nel sistema e può ottenere informazioni sui contenuti/appunti che ha scaricato.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>4. L'utente già autenticato, seleziona “Scarica contenuto/appunto da un corso/gruppo studio”.</li> <li>5. L'utente sceglie il corso/gruppo studio dal quale vuole scaricare il contenuto/gruppo studio.</li> <li>6. Il contenuto/appunto selezionato viene aggiunto alla lista dei contenuti/appunti dello studente.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. Il contenuto/appunto è inesistente, l'utente viene riportato alla schermata iniziale con un messaggio di errore.</li> <li>2. Il contenuto/appunto è già stato scaricato, l'utente viene riportato alla schermata iniziale con un messaggio di errore.</li> <li>3. Non esistono corsi/gruppi studio.</li> <li>4. Non esistono appunti/contenuti nel gruppo studio/corso.</li> </ol>

	5. L'utente può eliminare un contenuto/gruppo studio dalla sua lista.
<b>Requisiti speciali</b>	Interfaccia intuitiva per la scelta dei corsi/gruppi studio e degli appunti/contenuti.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Poche volte al giorno.
<b>Varie</b>	

### **UC10: visualizzazione dati dello studente**

<b>Nome</b>	UC10: visualizzazione dati dello studente
<b>Portata</b>	Gestione sistema di StudyHub
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Studente
<b>Parti interessate e interessi</b>	Studente: vuole conoscere i propri dati nel sistema, in modo da avere una visione completa dei corsi e dei gruppi studio a cui è iscritto, degli appunti e dei contenuti in suo possesso.
<b>Pre-condizioni</b>	Lo studente è registrato nel sistema. Non deve necessariamente essere iscritto a dei corsi/gruppi studio, stessa cosa per appunti/contenuti.
<b>Garanzia di successo</b>	Lo studente è autenticato nel sistema e può utilizzare le diverse funzioni del sistema stesso.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente già autenticato, seleziona “Visualizza informazioni studente”.</li> </ol>
<b>Estensioni</b>	<ol style="list-style-type: none"> <li>1. L'utente vuole aggiungere un contenuto/appunto ad un corso (creato)/gruppo studio.</li> <li>2. L'utente vuole eliminare un contenuto da un corso creato.</li> <li>3. L'utente vuole eliminare un appunto creato da lui da un gruppo studio.</li> <li>4. L'utente vuole scaricare un contenuto/appunto da un corso/gruppo studio.</li> <li>5. L'utente vuole modificare un appunto personale.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per la comprensione dei dati mostrati.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Molte volte.
<b>Varie</b>	

## **5. Regole di business**

- Un corso può avere una durata massimo di un mese.
- I gruppi di studio possono avere massimo 10 partecipanti.
- I gruppi studio hanno una durata massima di una settimana.
- Gli utenti devono essere autenticati per accedere a qualsiasi funzionalità.
- I pagamenti per i corsi sono obbligatori per i corsi a pagamento prima dell'iscrizione.
- Gli appunti caricato non possono superare dimensione di 10MB.

## **6. Specifiche supplementari**

- Usabilità: l'interfaccia deve essere intuitiva e accessibile su desktop e mobile.
- Sicurezza. Tutti i dati utente devono essere protetti con crittografia.
- Performance: il sistema deve garantire tempi di risposta inferiori a tre secondi per le operazioni principali.
- Scalabilità: il sistema deve supportare un numero crescente di utente senza degradare le prestazioni.

## **7. Glossario**

- Studente: utente registrato sulla piattaforma StudyHub.
- Corso: insieme di materiali e lezioni disponibili per gli studenti.
- Gruppo studio: un ambiente collaborativo per studenti con interessi simili.
- Appunto: documento caricato dagli studenti per la condivisione nei gruppi studio.
- Pagamento: transazione per accedere ai corsi a pagamento.
- DatiPagamento: metodo di pagamento dell'utente.
- Contenuto: materiale inserito in un corso dal creatore e disponibile agli iscritti.
- Iscrizione: iscrizione di uno studente ad un corso.

## **8. Analisi orientata agli oggetti**

L'implementazione dell'applicazione è stata guidata da un processo iterativo ed evolutivo basato sul modello UP (Unified Process), suddiviso in tre fasi. Grazie a questo metodo è stato possibile sviluppare gradualmente l'architettura del software StudyHub.

Ogni iterazione tratta i seguenti punti:

- Iterazione 1:

- Lo scopo dell' iterazione seguente sarà quello di:
  - Raffinare la visione.
  - Implementare in maniera iterativa il nucleo dell'architettura del software.
  - Risolvere le problematiche relative ai rischi maggiori.
  - Identificare la maggior parte dei requisiti e la portata.
  - Fornire delle stime più realistiche del piano di lavoro e delle risorse complessive.
- I requisiti scelti su cui concentrarsi sono i seguenti:
  - Implementare gli scenari di successo dei primi casi d'uso:
  - UC2: iscrizione ad un corso di studio.UC5: caricamento di contenuti.
  - UC6: caricamento di appunti.
  - Implementare i casi d'uso di startup per gestire l'inizializzazione di questa specifica iterazione.

- Iterazione 2:

- Riordinamento dei diversi casi d'uso.
  - UC2 diventa UC4: Iscrizione ad un corso.
  - UC6 diventa UC7: Caricamento di contenuti.

- UC7 diventa UC8: Caricamento di appunti.
- Implementazione degli scenari di successo dei seguenti casi d'uso:
  - UC1: iscrizione profilo.
  - UC2: modifica profilo.
  - UC3: creazione di un corso.
  - UC5: creazione di un gruppo studio.
  - UC6: ricerca ed iscrizione ad un gruppo studio.
- Implementazione di una funzione di login che consenta l'accesso allo studente con le proprie credenziali.
- Implementazione di un menù di scelta relativo alla registrazione, login o uscita dal software.
- Miglioramento della comprensibilità e dell'efficienza del codice.

- Iterazione 3:

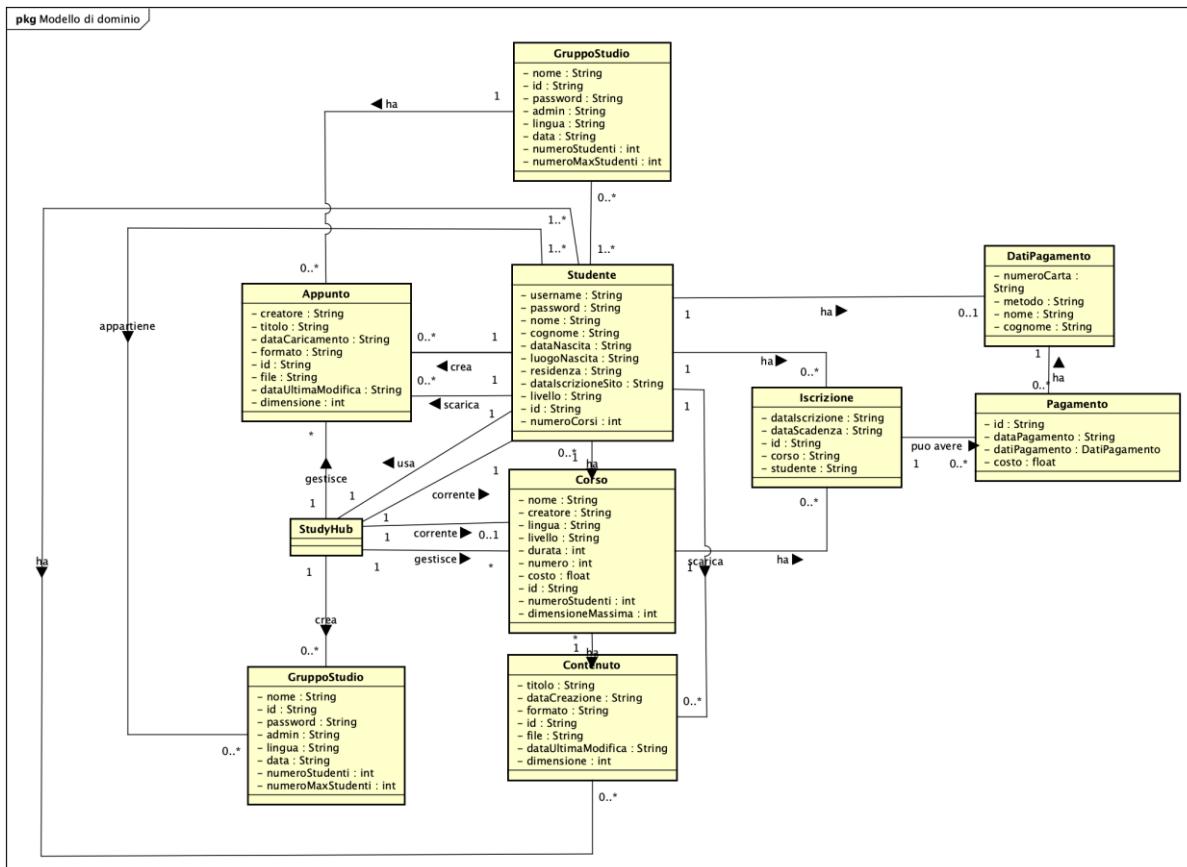
- Implementazione di scenari alternativi:
  - UC1: eliminazione profilo.
  - UC3: eliminazione di un corso.
  - UC4: disiscrizione da un corso.
  - UC5: eliminazione di un gruppo studio.
  - UC6: abbandono di un gruppo studio.
  - UC7: eliminazione di un contenuto.
  - UC8: eliminazione di un appunto.
- Implementazione di scenari negativi:
  - UC1: username già in uso.
  - UC3: dati corso errati (nome che contiene caratteri non consentiti).
  - UC4: iscrizione già eseguita e pagamento fallito.
  - UC5: nome gruppo già in uso.
  - UC6: iscrizione già eseguita o gruppo pieno.
  - UC7: file non supportato o troppo grande.
  - UC8: appunti duplicati.

- Iterazione 4:

- Implementazione UC9: scaricare un contenuto/appunto da un corso/gruppo studio.
- Implementazione UC10: visualizzazione dei dati dello studente.
- Migliore gestione delle schermate di accesso alle varie funzioni e relative modifiche.
- Pulizia del codice.

## **9. Modello di dominio**

La Modellazione del Business è la disciplina che si occupa di fornire una visione completa del dominio dell'applicativo. In particolare, essa include la creazione del modello di dominio e di un elaborato grafico che individua concetti, attributi e associazioni rilevanti. Il modello di dominio viene iterato tenendo conto di ciascuna Iterazione e si presenta come segue:



**Studente:** rappresenta un utente dell'applicazione. Questo contiene:

- username, password, nome, cognome → credenziali e dati personali
- dataNascita, luogoNascita, residenza → informazioni anagrafiche
- dataIscrizione, livello, numeroCorsi → informazioni di iscrizione e partecipazione

**Relazioni:**

- ha una o più Iscrizioni
- ha al massimo un DatiPagamento
- può creare uno o più corsi
- può caricare o scaricare appunti
- può fare parte di più gruppi studio

**Iscrizione:** definisce l'associazione tra uno Studente e un Corso. Contiene:

- dataIscrizione, dataScadenza → date di inizio e fine
- id, corso, studente → identificativi delle entità coinvolte

**Relazioni:**

- può avere uno o più Pagamenti

**Pagamento:** gestisce i pagamenti effettuati dallo Studente. Contiene:

- id → identificativo del pagamento
- dataPagamento → data della transazione
- costo → importo del pagamento

**Relazioni:**

- ha un DatiPagamento associato

**DatiPagamento:** memorizza i dettagli di pagamento dello studente. Contiene:

- numeroCarta → numero della carta di pagamento
- metodo → tipo di pagamento (es. carta di credito, PayPal, ecc.)
- nome, cognome → titolare della carta

Relazioni:

- può essere associato a un Pagamento

**Corso:** rappresenta un corso disponibile su *StudyHub*. Contiene:

- nome, creatore, livello, lingua, durata → informazioni generali
- numero, costo → dettagli identificativi e di prezzo
- numeroStudenti, dimensioneMassima → gestione della capienza

Relazioni:

- ha uno o più Studenti iscritti
- può avere più contenuti caricati

**Contenuto:** rappresenta il materiale didattico di un Corso. Contiene:

- titolo, dataCreazione, formato, id → dettagli del file
- dataUltimaModifica, dimensione → aggiornamenti e grandezza

Relazioni:

- è associato a un Corso

**Appunto:** materiale didattico creato dagli studenti. Contiene:

- creatore, titolo, dataCaricamento, formato → dettagli dell'appunto
- id, dataUltimaModifica, dimensione → informazioni di gestione

Relazioni:

- appartiene a uno Studente
- può essere caricato su un gruppo studio
- creato tramite *StudyHub*

**GruppoStudio:** permette la collaborazione tra studenti. Contiene:

- nome, id, password, admin, lingua, data → informazioni del gruppo
- numeroStudenti, numeroMaxStudenti → gestione della capienza

Relazioni:

- uno Studente può appartenere a più GruppiStudio
- può avere più appunti caricati

- creato tramite *StudyHub*

**StudyHub:** è il sistema centrale che gestisce tutti gli elementi.

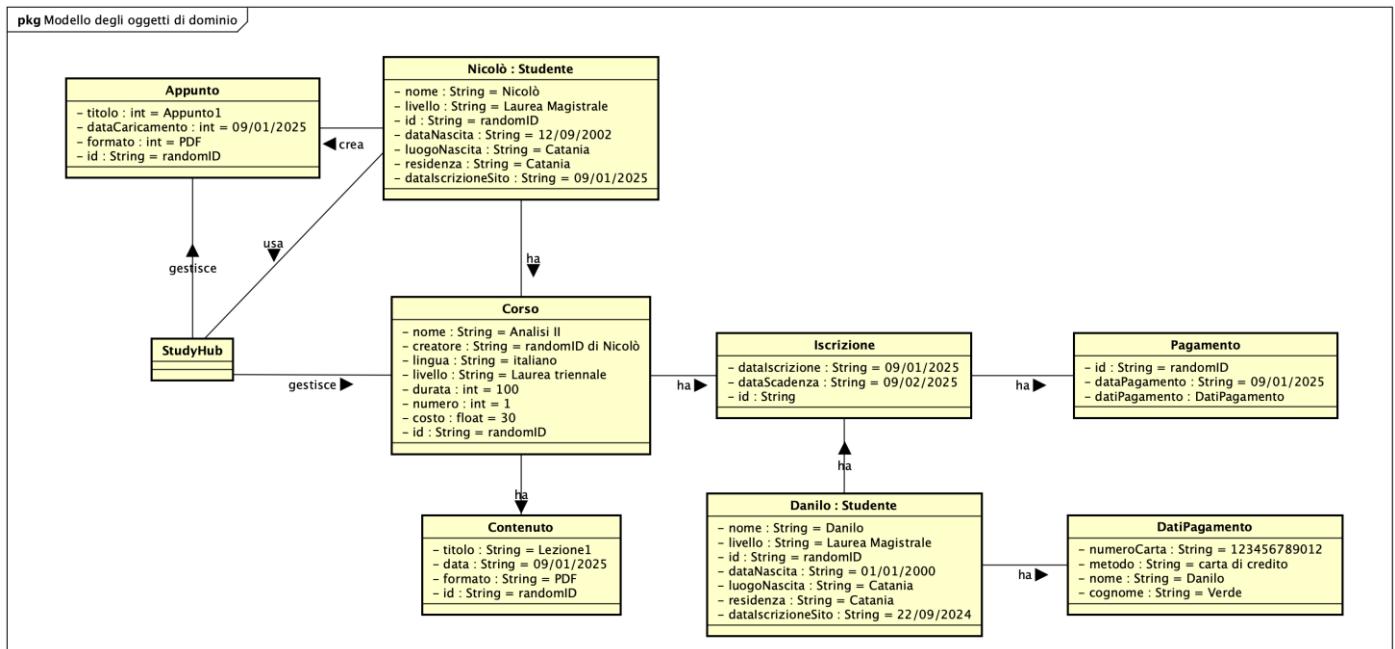
Relazioni:

- gestisce Studenti, Corsi e GruppiStudio
- crea Appunti e GruppiStudio

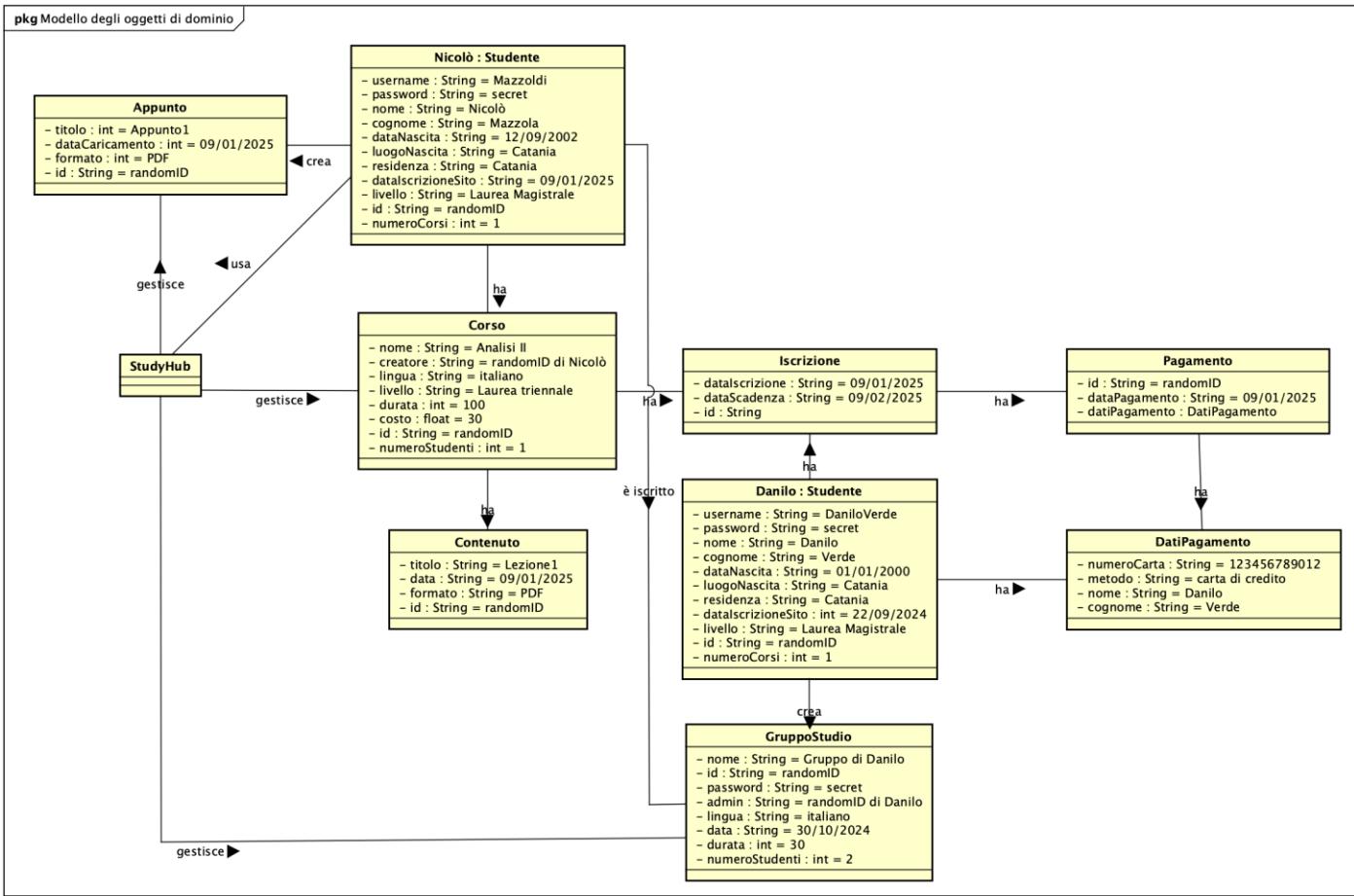
## 10. Modello degli oggetti

Il modello degli oggetti di dominio fornisce una rappresentazione strutturata degli elementi chiave di un sistema, evidenziando le entità principali, i loro attributi e le relazioni che le interconnettono. Questo schema consente di visualizzare e comprendere in modo chiaro l'organizzazione dei dati all'interno del dominio di riferimento, facilitando l'analisi e la progettazione del sistema.

**Iterazione 1:** lo studente Nicolò crea un appunto personale (Appunto1), crea un corso di Analisi II che costa 30 euro mensili e vi carica un contenuto (Lezione1). Al corso è iscritto Danilo, che ha una Iscrizione con un Pagamento eseguito con il suo DatiPagamento.



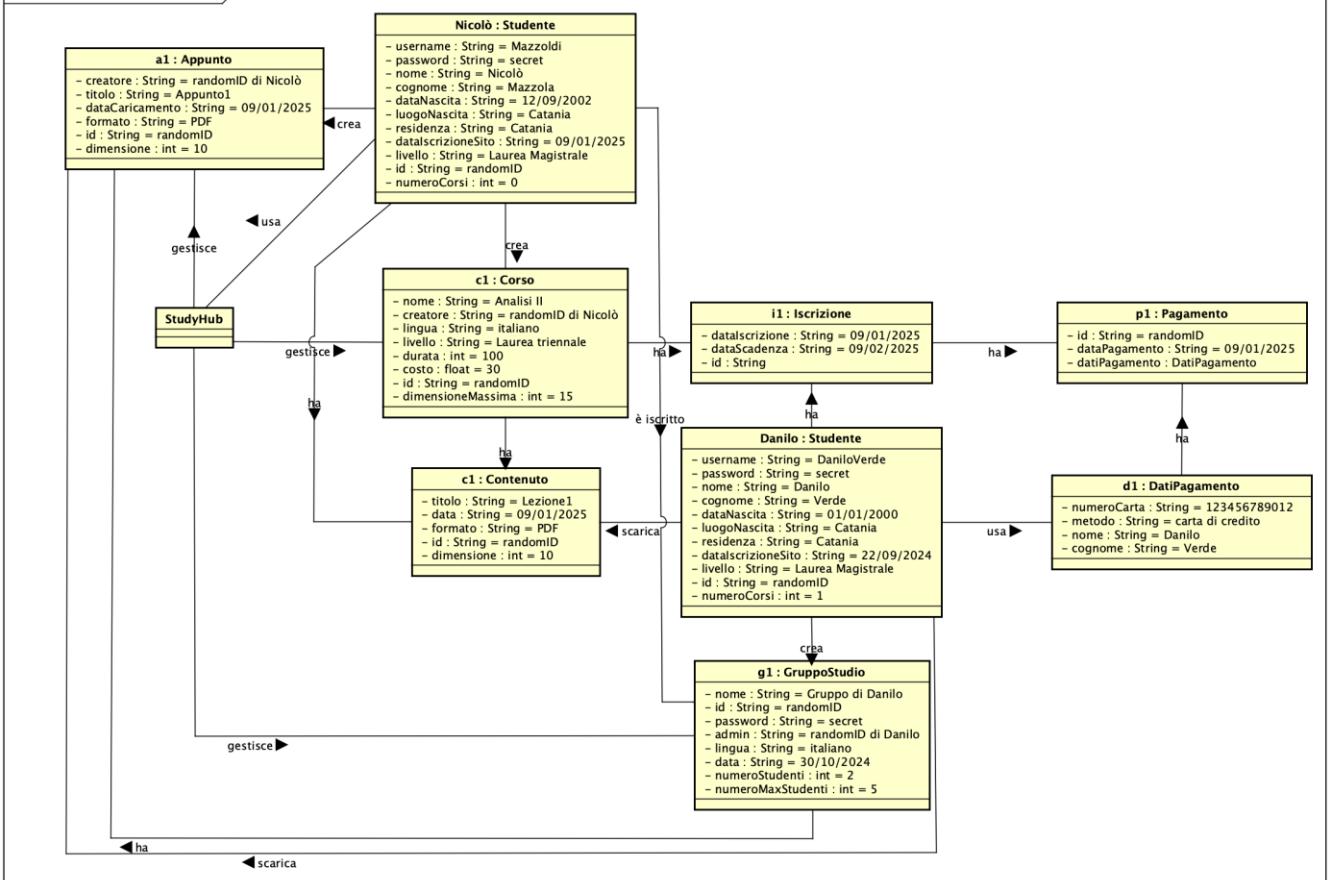
**Iterazione 2:** lo studente Danilo ha creato un GruppoStudio (Gruppo di Danilo) a cui si è iscritto Nicolò. È stato esplicitato il collegamento tra Pagamento e DatiPagamento.



**Iterazione 3:** non sono state apportate modifiche significative al modello degli oggetti di dominio, visto che l'iterazione si è concentrata su scenari negativi e alternativi.

**Iterazione 4:** L'appunto creato da Nicolò è stato caricato sul gruppo studio creato da Danilo, che lo ha scaricato, così come il contenuto caricato nel corso di Analisi II di Nicolò.

pkg Modello degli oggetti di dominio



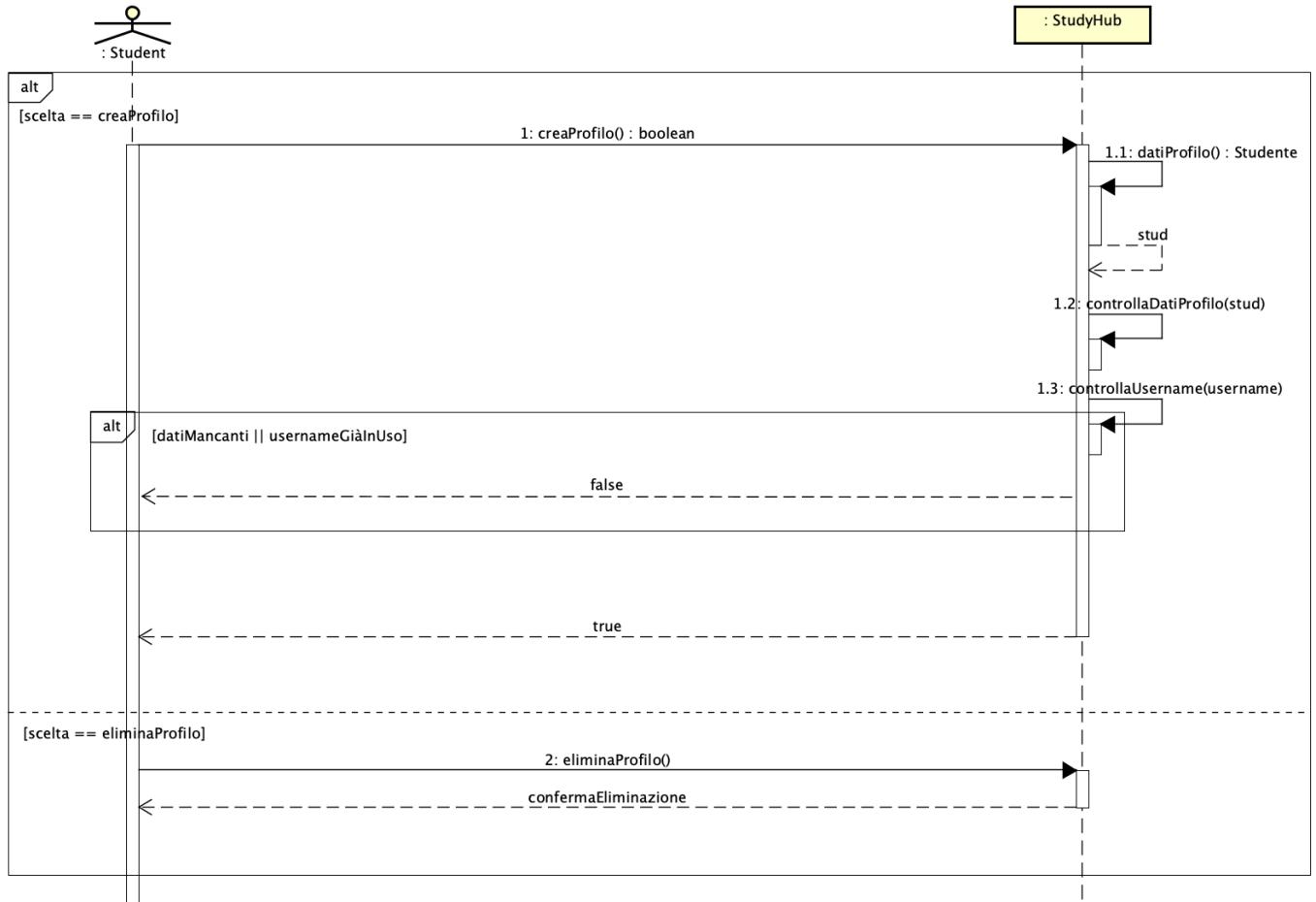
## 11. SSD e contratti delle operazioni

Nel contesto dell'analisi orientata agli oggetti, successivamente vediamo la creazione dei Diagrammi di Sequenza di Sistema (SSD) al fine di delineare il flusso degli eventi di input e output relativi ai vari casi d'uso analizzati in ciascuna iterazione. Questi diagrammi forniscono una rappresentazione visuale del comportamento del sistema in risposta alle interazioni utente.

Successivamente, le operazioni di sistema rilevanti individuate nei Diagrammi di Sequenza di Sistema verranno dettagliatamente descritte mediante l'utilizzo di Contratti. I Contratti rappresentano una formalizzazione delle responsabilità, delle condizioni pre e post-operazione, e delle eccezioni associate a ciascuna operazione di sistema.

In questo modo, si ottiene una visione dettagliata e chiara del funzionamento del sistema, consentendo una migliore comprensione delle interazioni e delle responsabilità delle componenti coinvolte. Tale approccio facilita anche la fase successiva di progettazione, fornendo una base solida per l'implementazione delle funzionalità richieste.

### UC1



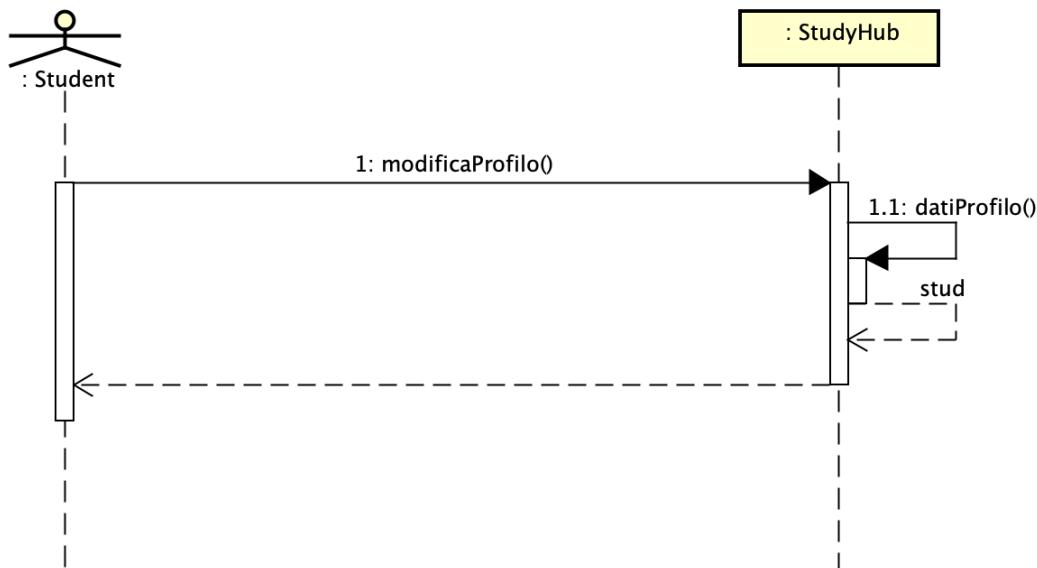
Contratto CO1: creaProfilo

<b>Operazioni</b>	creaProfilo()
<b>Riferimenti</b>	UC1: Creazione di un corso.
<b>Pre-condizioni</b>	Lo studente non è registrato nel sistema.
<b>Post-condizioni</b>	Lo studente è autenticato e inserito nel sistema, un'istanza è stata creata.

### Contratto CO2: eliminaProfilo

<b>Operazioni</b>	eliminaProfilo()
<b>Riferimenti</b>	UC1: Creazione di un corso.
<b>Pre-condizioni</b>	Lo studente è registrato nel sistema.
<b>Post-condizioni</b>	Lo studente è portato alla pagina di login, registrazione o uscita dal sistema.

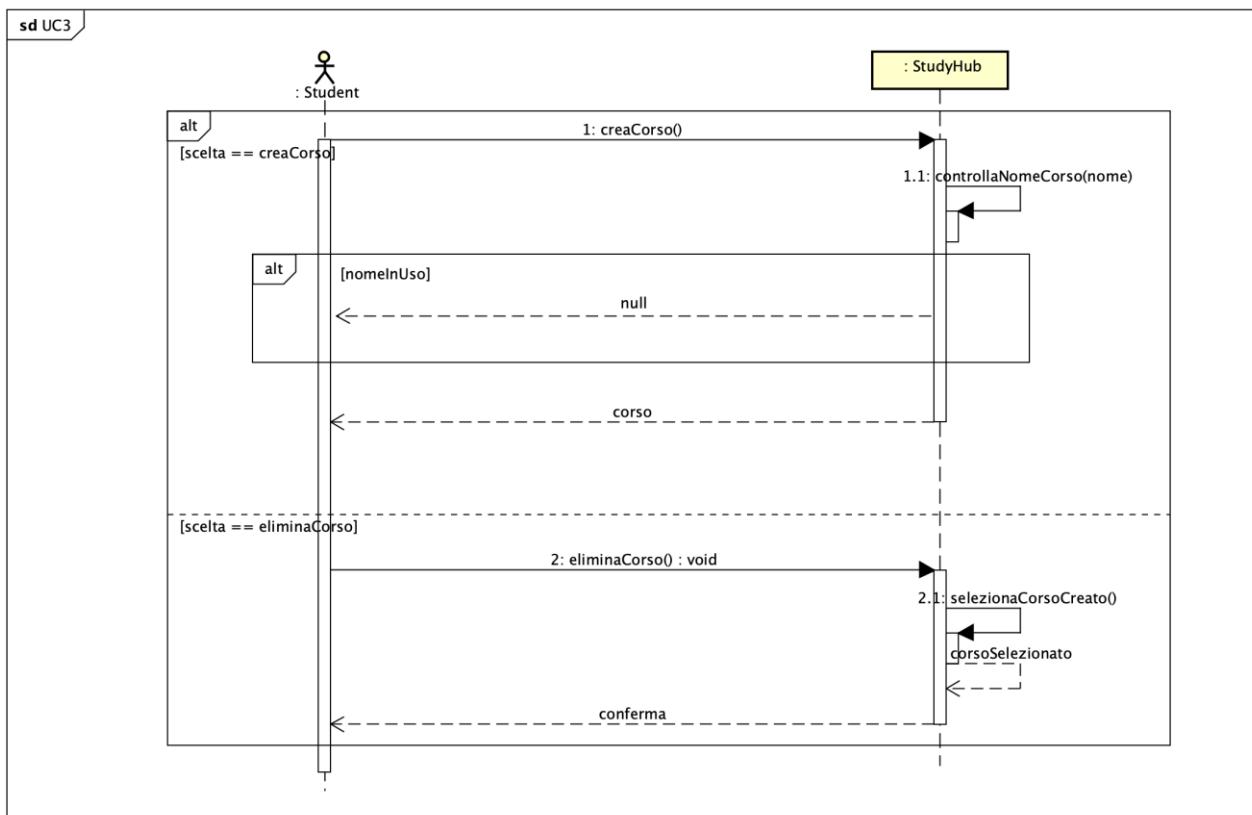
UC2



### Contratto CO1: modificaProfilo

<b>Operazioni</b>	modificaProfilo()
<b>Riferimenti</b>	UC2: Modifica profilo.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema. Lo studente è già autenticato.
<b>Post-condizioni</b>	Lo studente è ancora autenticato nel sistema e i suoi dati sono stati modificati.

## UC3



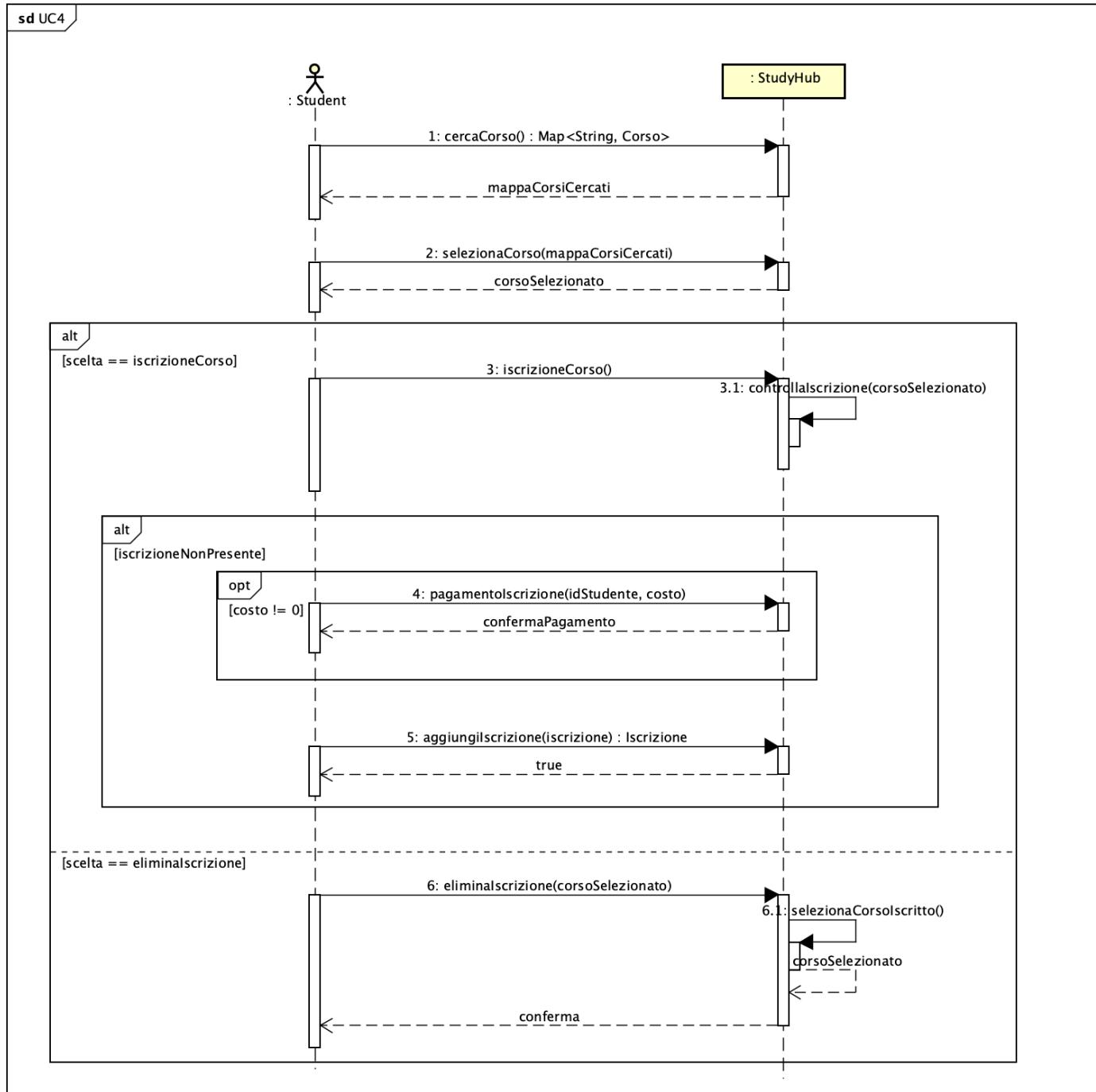
Contratto CO1: creaCorso

<b>Operazioni</b>	<b>creaCorso()</b>
<b>Riferimenti</b>	UC3: Creazione di un corso.
<b>Pre-condizioni</b>	Lo studente è autenticato nel sistema e il corso non deve esistere.
<b>Post-condizioni</b>	Lo studente è autenticato e inserito nel sistema, un'istanza è stata creata.

Contratto CO2: eliminaCorso

<b>Operazioni</b>	<b>eliminaCorso()</b>
<b>Riferimenti</b>	UC3: Creazione di un corso.
<b>Pre-condizioni</b>	Lo studente è l'admin del corso.
<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. Un corso con le stesse credenziali può essere creato.

## UC4



### Contratto CO1: selezionaCorso

<b>Operazioni</b>	selezionaCorso()
<b>Riferimenti</b>	UC4: Iscrizione ad un corso.
<b>Pre-condizioni</b>	Esiste una lista di corsi e lo studente è iscritto e autenticato nel sistema.
<b>Post-condizioni</b>	È stata selezionata un'istanza di un corso.

### Contratto CO2: iscrizioneCorso

<b>Operazioni</b>	iscrizioneCorso()
<b>Riferimenti</b>	UC4: Iscrizione ad un corso.
<b>Pre-condizioni</b>	È stato selezionato un corso ed è stata creata un'istanza di esso.
<b>Post-condizioni</b>	È stata creata un'istanza di iscrizione ed è stata ricevuta una conferma di essa.

#### Contratto CO3: pagamentoIscrizione

<b>Operazioni</b>	pagamentoIscrizione()
<b>Riferimenti</b>	UC4: Iscrizione ad un corso.
<b>Pre-condizioni</b>	Il corso prevede un costo ed è stata creata un'istanza di un iscrizione.
<b>Post-condizioni</b>	È stata ricevuta una conferma di pagamento e di avvenuta iscrizione.

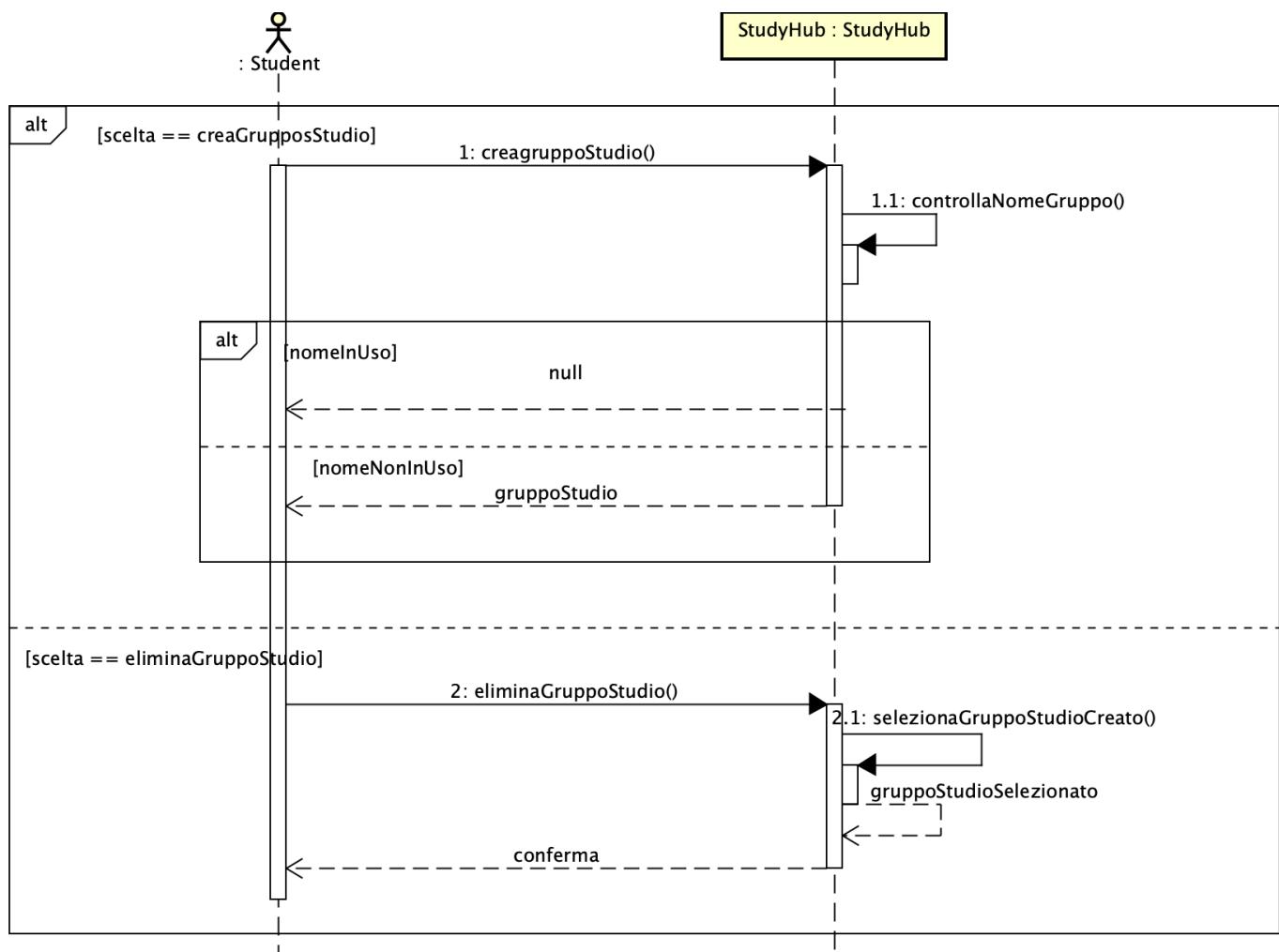
#### Contratto CO4: eliminaIscrizione

<b>Operazioni</b>	eliminaCorso()
<b>Riferimenti</b>	UC4: Iscrizione ad un corso.
<b>Pre-condizioni</b>	Lo studente è iscritto al corso.
<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. Non può più caricare contenuti nel corso.

#### Contratto CO5: selezionaCorsoIscritto

<b>Operazioni</b>	selezionaCorsoIscritto()
<b>Riferimenti</b>	UC4: Iscrizione ad un corso.
<b>Pre-condizioni</b>	Lo studente è iscritto al corso ed è in fase di disiscrizione.
<b>Post-condizioni</b>	Il sistema continua con la fase di disiscrizione.

**UC5**



### Contratto CO1: creaGruppoStudio

<b>Operazioni</b>	creaGruppoStudio()
<b>Riferimenti</b>	UC5: Creazione gruppo studio.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema. Il gruppo non è già esistente.
<b>Post-condizioni</b>	È stata aggiunta un'istanza di GruppoStudio e viene aggiunta alla lista dei gruppi totali e dello studente admin.

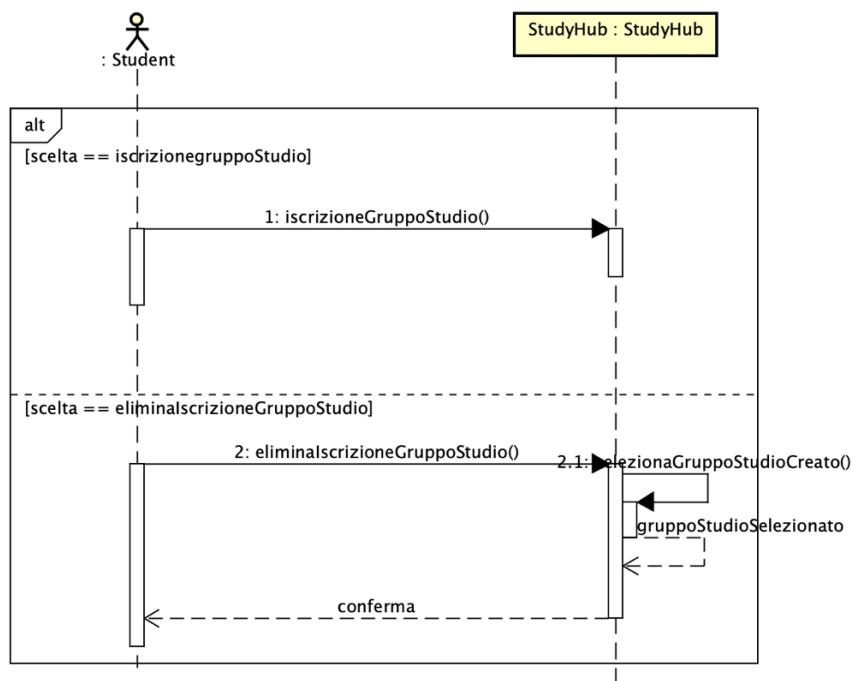
### Contratto CO2: eliminaGruppoStudio

<b>Operazioni</b>	eliminaGruppoStudio()
<b>Riferimenti</b>	UC5: Creazione gruppo studio.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema ed è proprietario del gruppo.
<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. Lo studente può accedere al gruppo.

## Contratto CO3: selezionaGruppoStudioCreato

<b>Operazioni</b>	selezionaGruppoStudioCreato()
<b>Riferimenti</b>	UC5: Creazione gruppo studio.
<b>Pre-condizioni</b>	Lo studente è proprietario del gruppo
<b>Post-condizioni</b>	Il sistema continua con la fase di eliminazione del gruppo studio che ha selezionato.

## UC6



## Contratto CO1: iscrizioneGruppoStudio

<b>Operazioni</b>	iscrizioneGruppoStudio()
<b>Riferimenti</b>	UC6: Iscrizione gruppo studio.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema. Lo studente non è già iscritto al gruppo. Il gruppo è esistente.
<b>Post-condizioni</b>	Lo studente viene aggiunto alla lista degli studenti del gruppo, e viceversa.

## Contratto CO2: eliminarscrizioneGruppoStudio

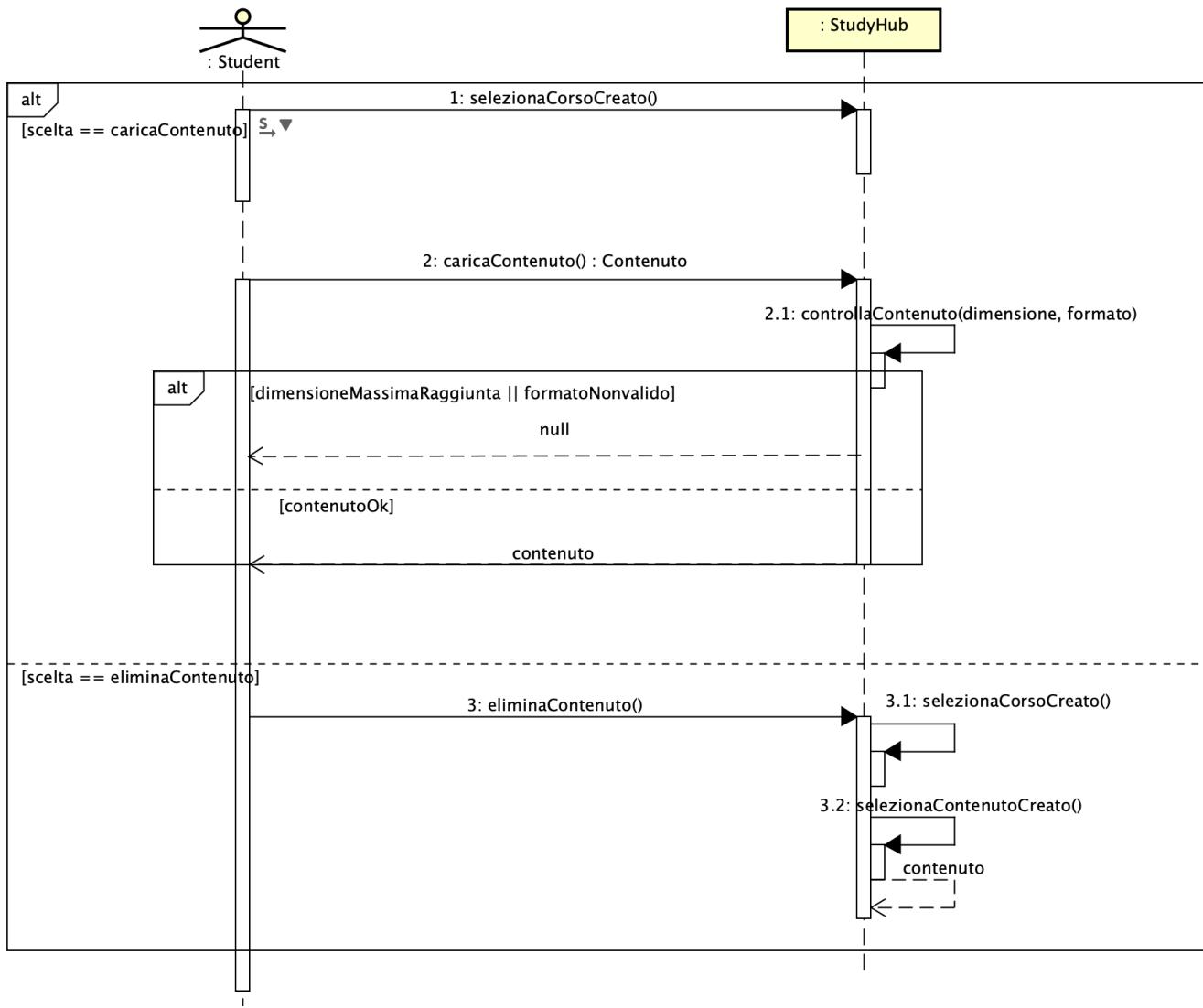
<b>Operazioni</b>	eliminaIscrizioneGruppoStudio()
<b>Riferimenti</b>	UC6: Iscrizione ad un gruppo studio.
<b>Pre-condizioni</b>	Lo studente è iscritto al gruppo studio.

<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. Non può più caricare accedere al gruppo.
------------------------	--

### Contratto CO3: selezionaGruppoStudioIscritto

<b>Operazioni</b>	selezionaGruppoStudioIscritto()
<b>Riferimenti</b>	UC6: Iscrizione ad un gruppo studio.
<b>Pre-condizioni</b>	Lo studente è iscritto al gruppo studio ed è in fase di disiscrizione.
<b>Post-condizioni</b>	Il sistema continua con la disiscrizione al gruppo.

**UC7**



### Contratto CO1: selezionaCorsoCreato

<b>Operazioni</b>	selezionaCorsoCreato()
<b>Riferimenti</b>	UC7: Carica contenuto.
<b>Pre-condizioni</b>	Lo studente ha una lista di corsi creati.
<b>Post-condizioni</b>	Viene selezionato uno tra i corsi creati e viene settato come corsoSelezionato.

### Contratto CO2: caricaContenuto

<b>Operazioni</b>	caricaContenuto()
<b>Riferimenti</b>	UC7: Carica contenuto.
<b>Pre-condizioni</b>	Lo studente è il creatore del corso.
<b>Post-condizioni</b>	Viene selezionato uno tra i corsi creati e viene settato come corsoSelezionato.

### Contratto CO3: eliminaContenuto

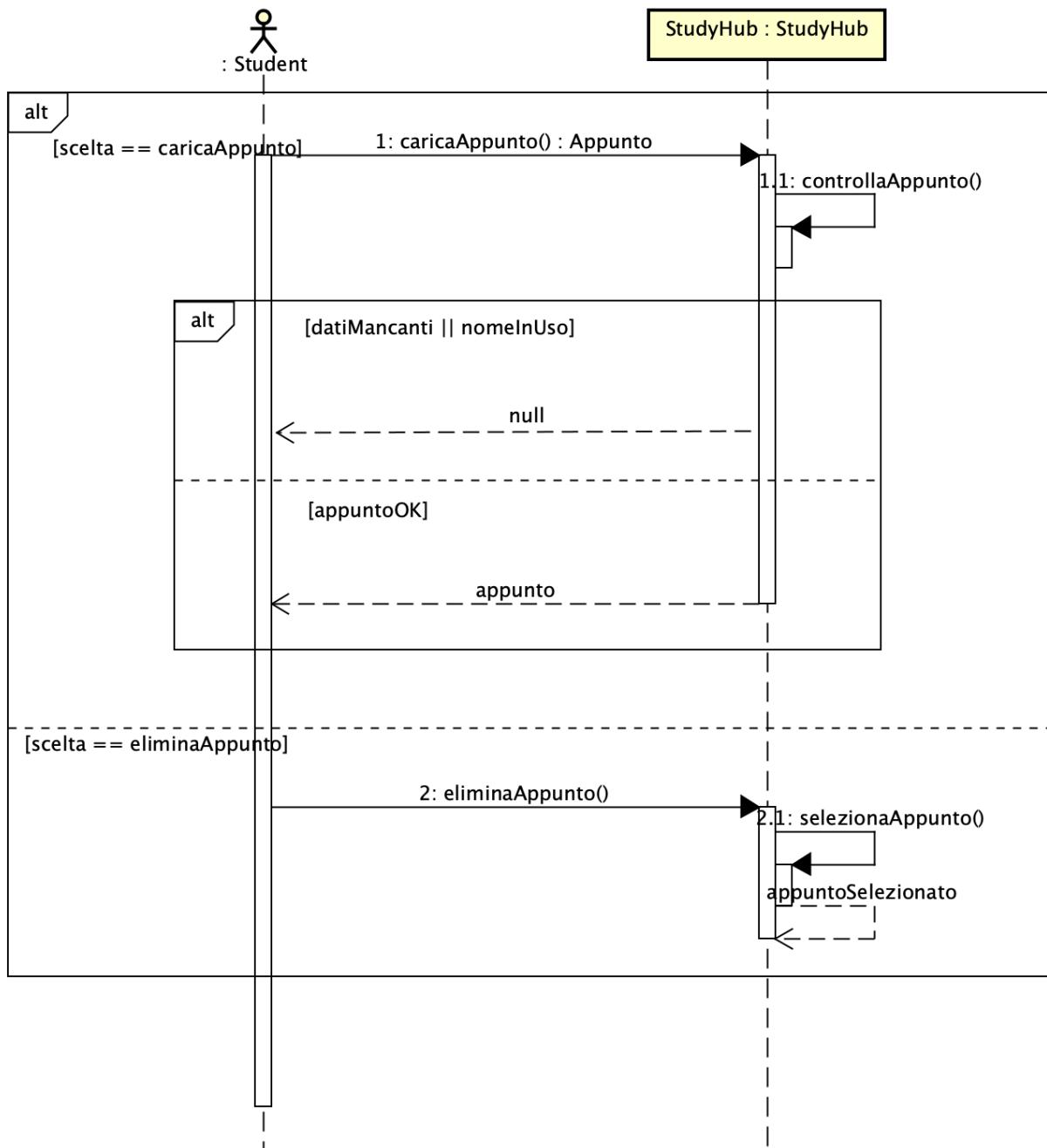
<b>Operazioni</b>	eliminaContenuto()
<b>Riferimenti</b>	UC7: Carica contenuto.

<b>Pre-condizioni</b>	Lo studente è iscritto al corso.
<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. Il contenuto non è più accessibile, un altro identico potrebbe essere ricaricato.

Contratto CO4: selezionaContenutoCreato

<b>Operazioni</b>	selezionaContenutoCreato()
<b>Riferimenti</b>	UC7: Carica contenuto.
<b>Pre-condizioni</b>	Lo studente è iscritto al corso ed è in fase di eliminazione del contenuto.
<b>Post-condizioni</b>	Il sistema continua con l'eliminazione del contenuto.

*UC8*



### Contratto CO1: caricaAppunto

<b>Operazioni</b>	<b>caricaAppunto()</b>
<b>Riferimenti</b>	UC8: Carica appunto.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema.
<b>Post-condizioni</b>	È stata aggiunta un'istanza di un appunto e viene aggiunta una alla lista appunti.

### Contratto CO2: eliminaAppunto

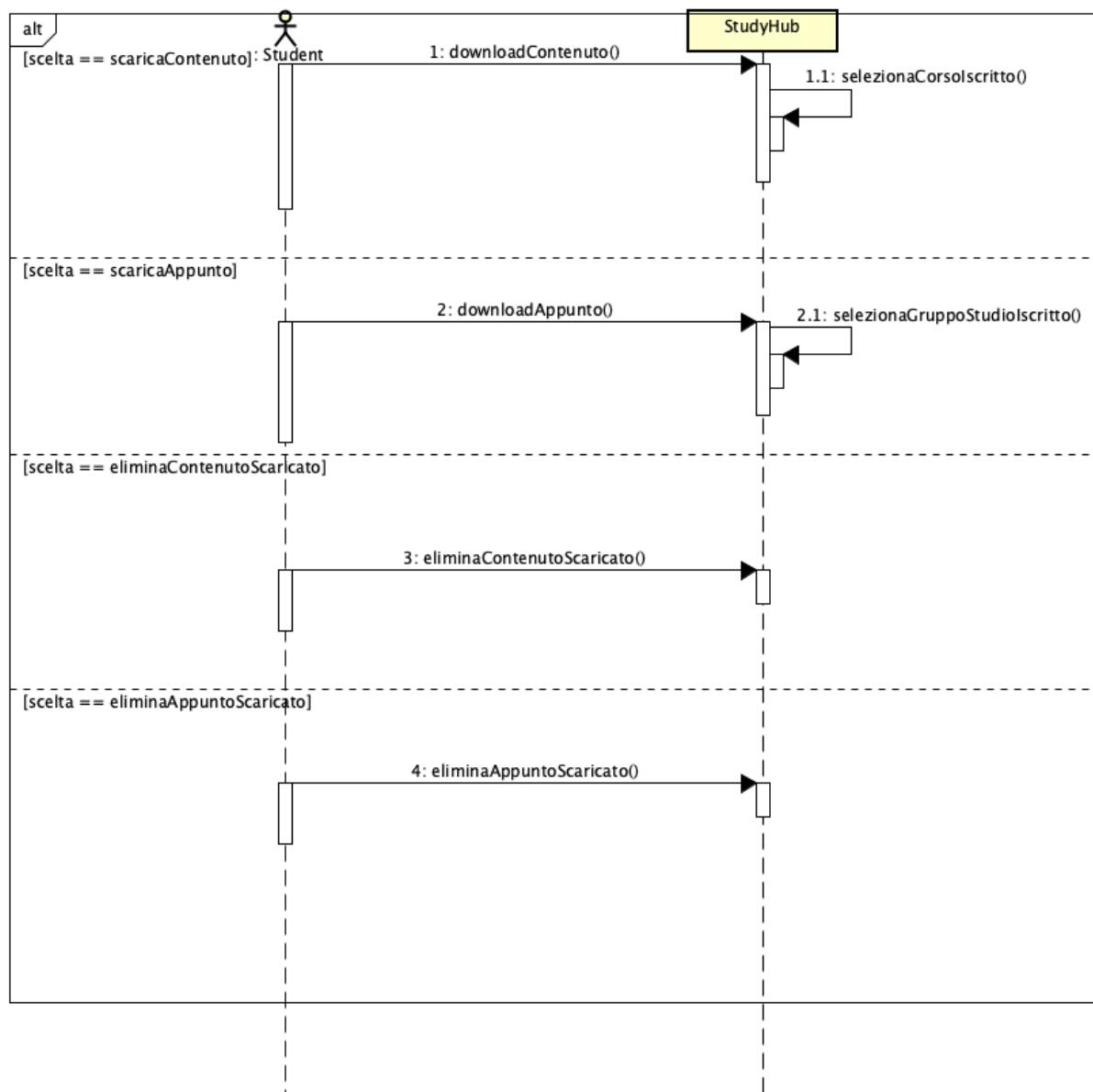
<b>Operazioni</b>	<b>eliminaAppunto()</b>
<b>Riferimenti</b>	UC8: Carica appunto.
<b>Pre-condizioni</b>	Lo studente è iscritto nel sistema.

<b>Post-condizioni</b>	Lo studente è portato alla pagina principale dell'app, già autenticato. L'appunto non è più accessibile, un altro identico potrebbe essere ricaricato.
------------------------	--

### Contratto CO3: selezionaAppunto

<b>Operazioni</b>	selezionaAppunto()
<b>Riferimenti</b>	UC8: Carica appunto.
<b>Pre-condizioni</b>	Lo studente è in fase di eliminazione dell'appunto.
<b>Post-condizioni</b>	Il sistema continua con l'eliminazione dell'appunto.

### UC9



### Contratto CO1: downloadAppunto

<b>Operazioni</b>	downloadAppunto()
<b>Riferimenti</b>	UC9: scarica contenuto/appunto da corso/gruppo studio.
<b>Pre-condizioni</b>	Lo studente è registrato nel sistema. Lo studente è iscritto al corso e il contenuto è presente.
<b>Post-condizioni</b>	Lo studente ritorna al menu principale.

Contratto CO2: downloadContenuto

<b>Operazioni</b>	downloadContenuto()
<b>Riferimenti</b>	UC9: scarica contenuto/appunto da corso/gruppo studio.
<b>Pre-condizioni</b>	Lo studente è registrato nel sistema. Lo studente è iscritto al gruppo studio e l'appunto è presente
<b>Post-condizioni</b>	Lo studente ritorna al menu principale.

Contratto CO3: eliminaAppuntoScaricato

<b>Operazioni</b>	eliminaAppuntoScaricato()
<b>Riferimenti</b>	UC9: scarica contenuto/appunto da corso/gruppo studio.
<b>Pre-condizioni</b>	Lo studente ha almeno un contenuto nella sua lista.
<b>Post-condizioni</b>	Lo studente ritorna al menu principale.

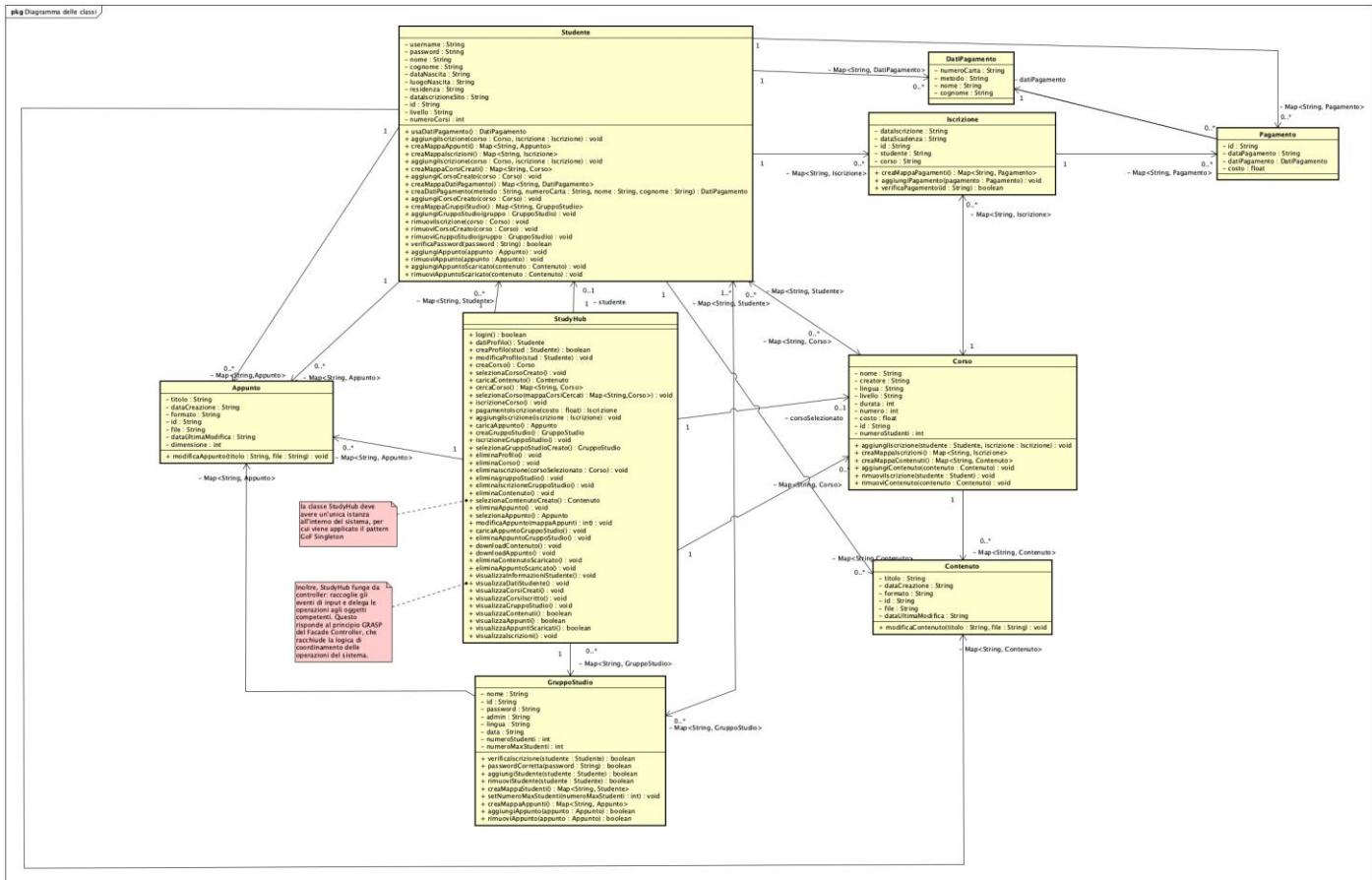
Contratto CO4: eliminaContenutoScaricato

<b>Operazioni</b>	eliminaContenutoScaricato()
<b>Riferimenti</b>	UC9: scarica contenuto/appunto da corso/gruppo studio.
<b>Pre-condizioni</b>	Lo studente ha almeno un appunto nella sua lista.
<b>Post-condizioni</b>	Lo studente ritorna al menu principale.

## 12. Progettazione

L'elaborato principale di questa fase che è stato preso in considerazione è il Modello di Progetto, ovvero l'insieme dei diagrammi che descrivono la progettazione logica sia da un punto di vista dinamico (Diagrammi di Interazione) che da un punto di vista statico (Diagramma delle Classi).

### Diagramma delle classi



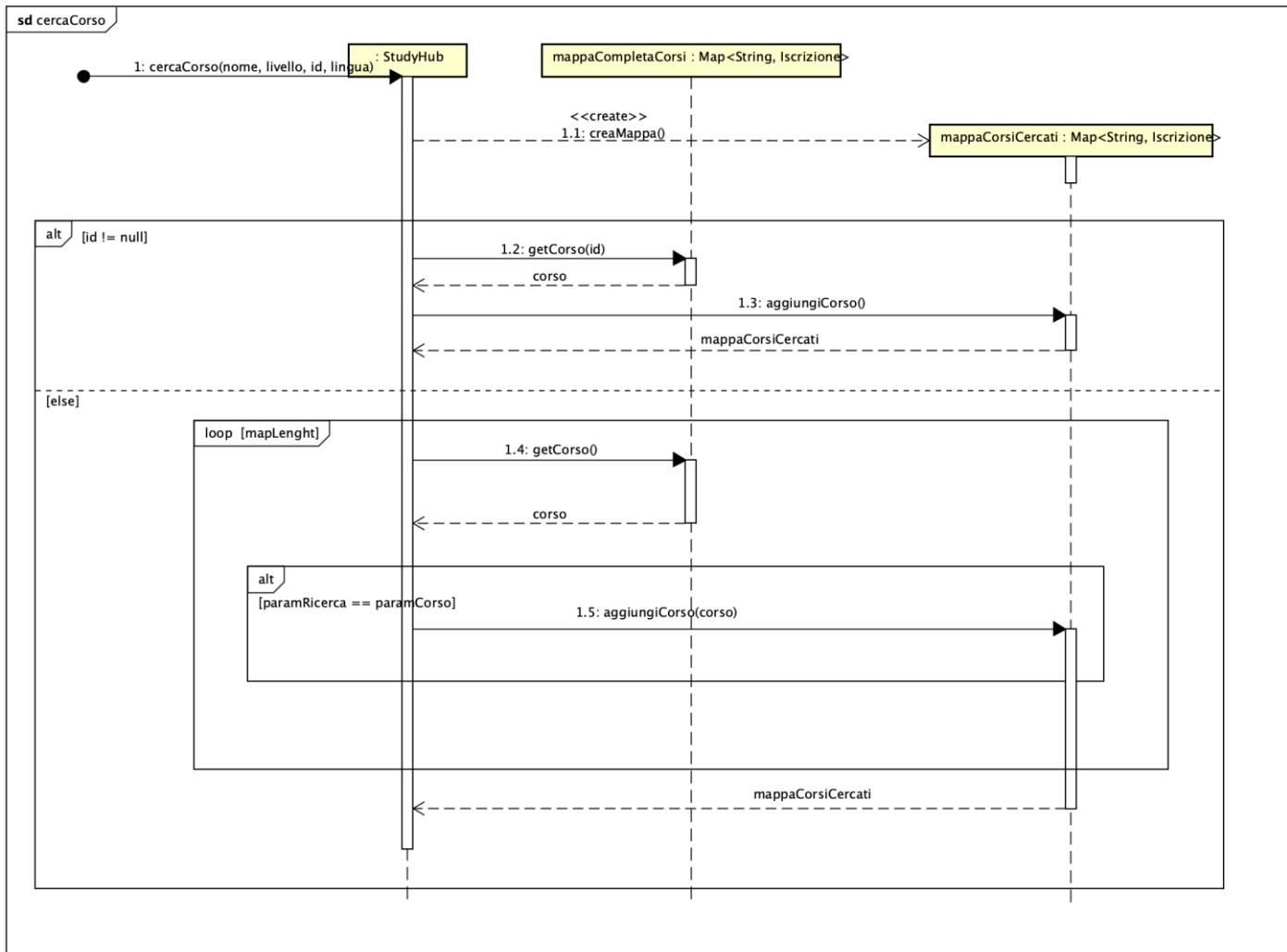
Abbiamo usato il pattern GoF Singleton e quello GRASP del Facade Controller con la classe *StudyHub*. Il costruttore è privato e la classe possiede un campo statico (instance) insieme a un metodo pubblico (getInstance) per ottenere l'unica istanza, garantendo così che venga creata una sola istanza della classe. Inoltre, *StudyHub* funge da controller centrale: raccoglie gli eventi di input tramite il menu interattivo e delega le operazioni (registrazione, login, creazione di corsi, iscrizioni, ecc.) agli oggetti competenti (come *Studente*, *CORSO*, *GruppoStudio*). Questo risponde al principio GRASP del Controller, che racchiude la logica di coordinamento delle operazioni del sistema. Pur non essendo formalmente implementato come una classe separata, *StudyHub* agisce da Facade concentrando e semplificando l'accesso a molteplici funzionalità (gestione di profili, corsi, gruppi studio, pagamenti, appunti, ecc.). In questo modo viene offerta un'interfaccia unificata al sistema, nascondendo la complessità delle interazioni tra i vari componenti.

## Diagramma di sequenza

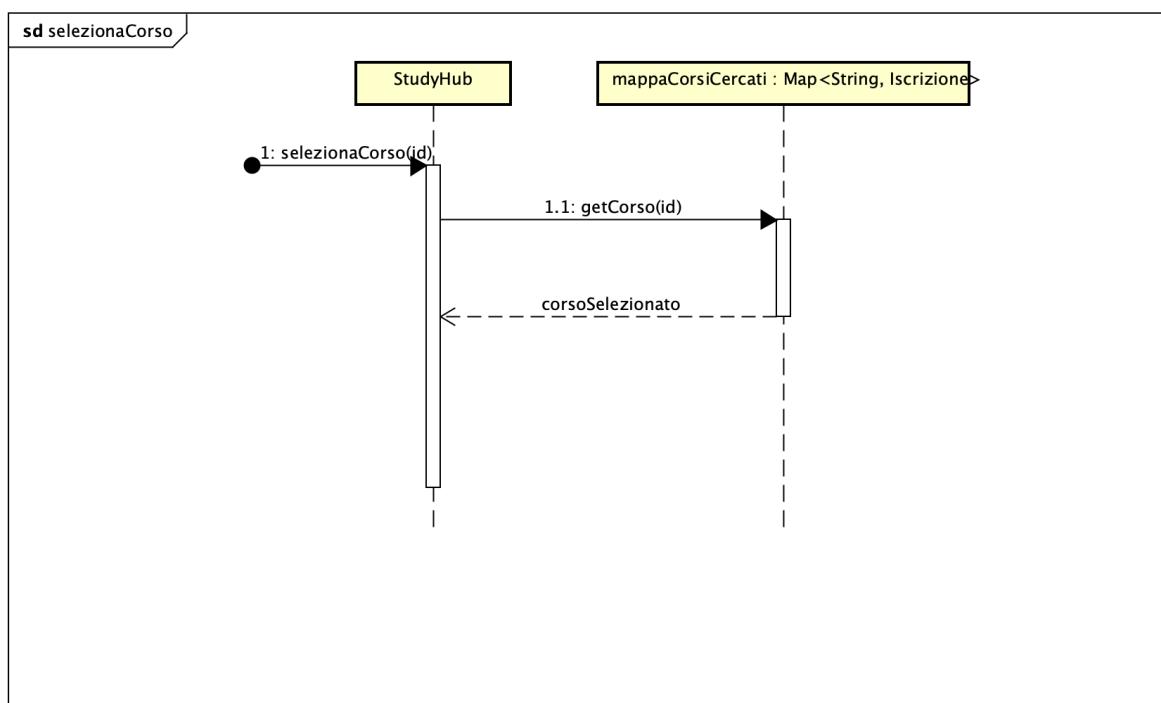
## Iterazione 1

UC2

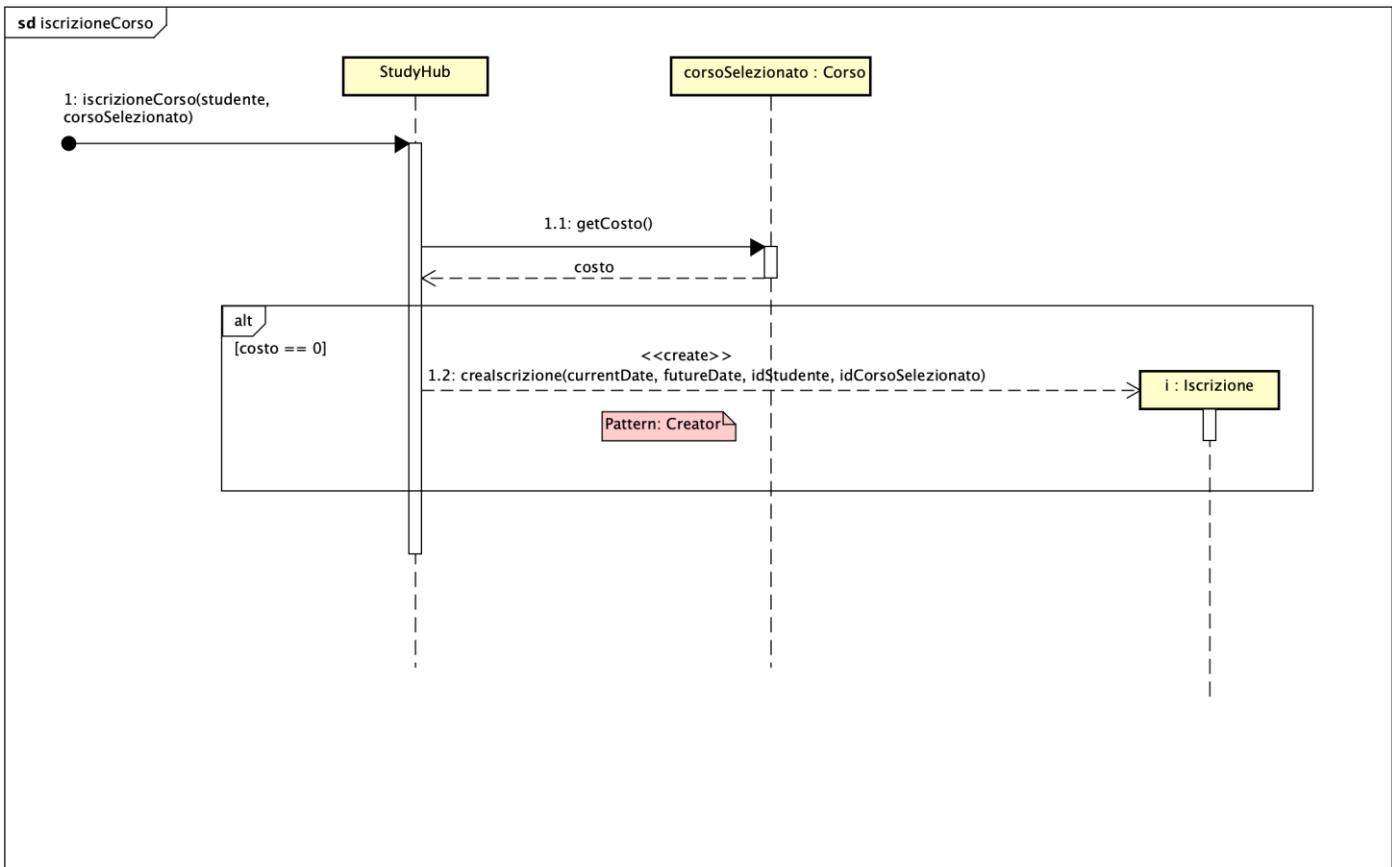
- cercaCorso



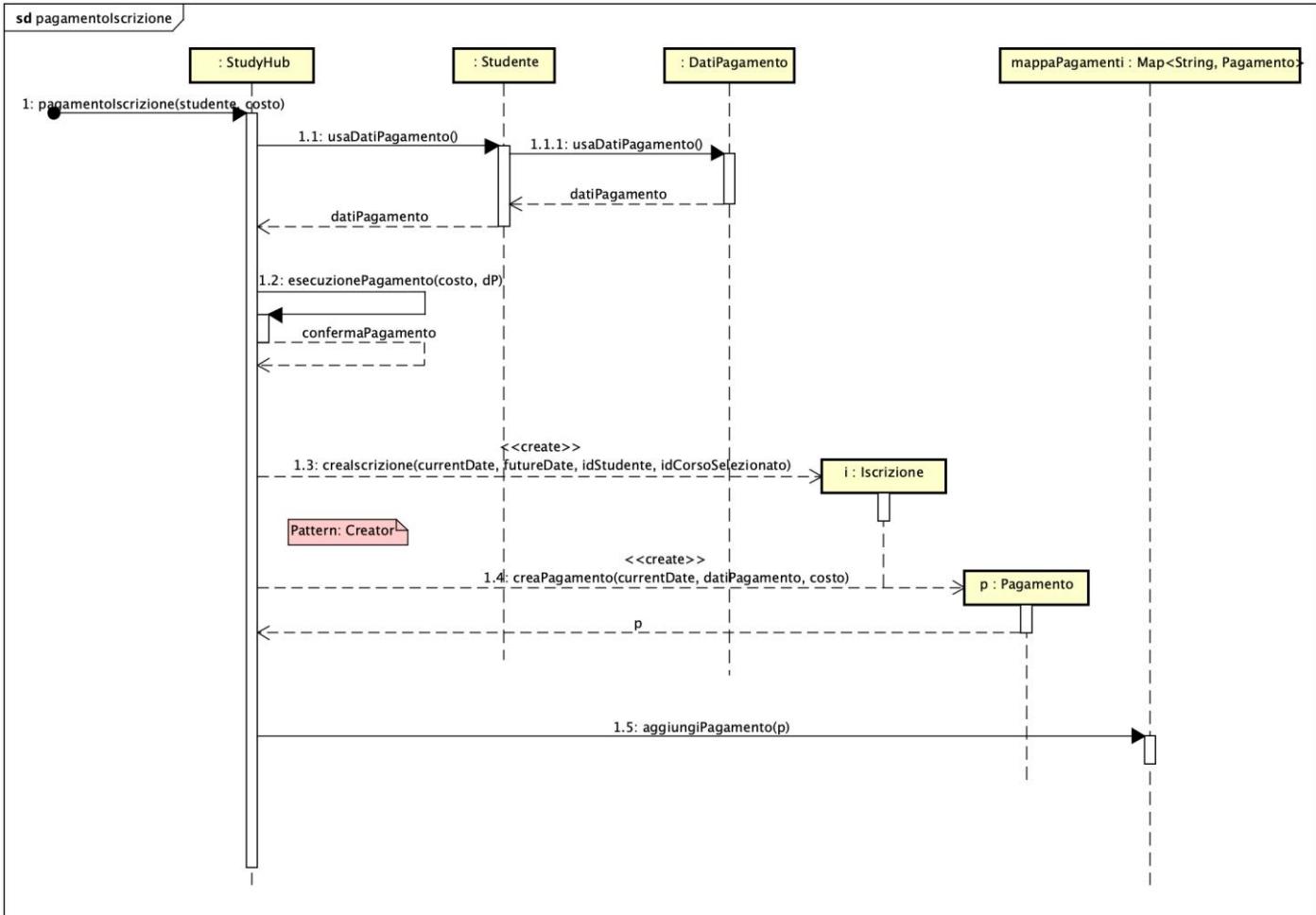
### -selezionaCorso



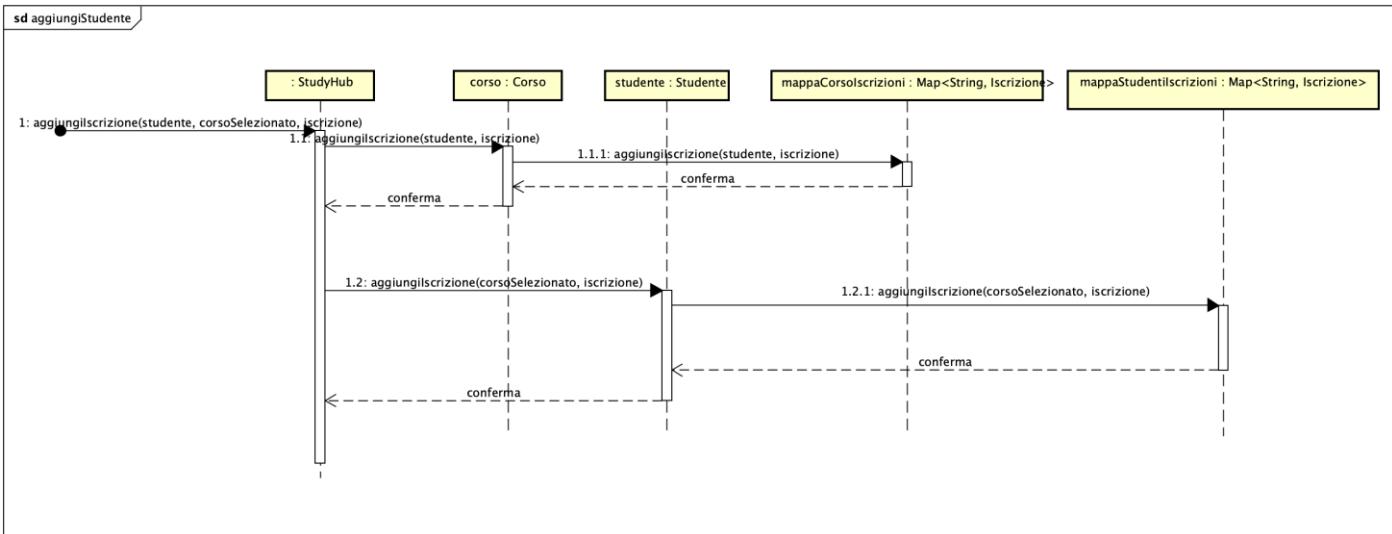
### -iscrizioneCorso



## -pagamentoIscrizione

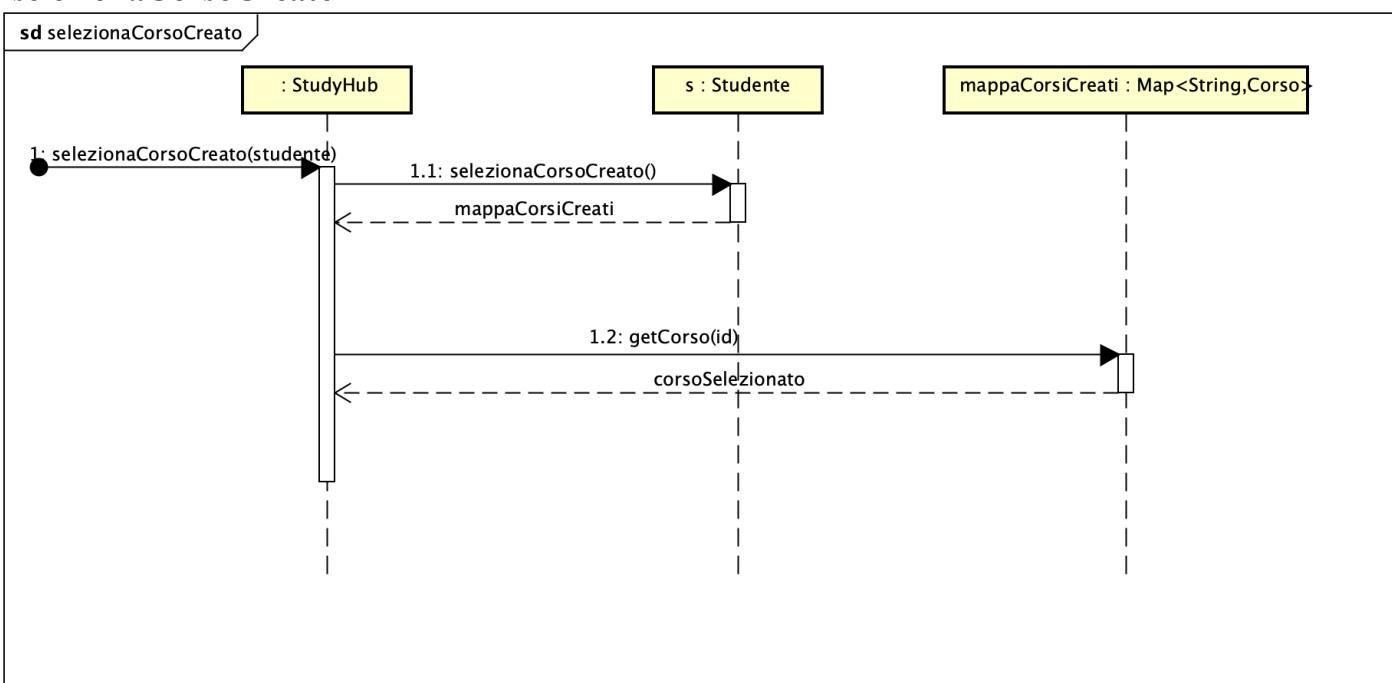


## -aggiungiStudente

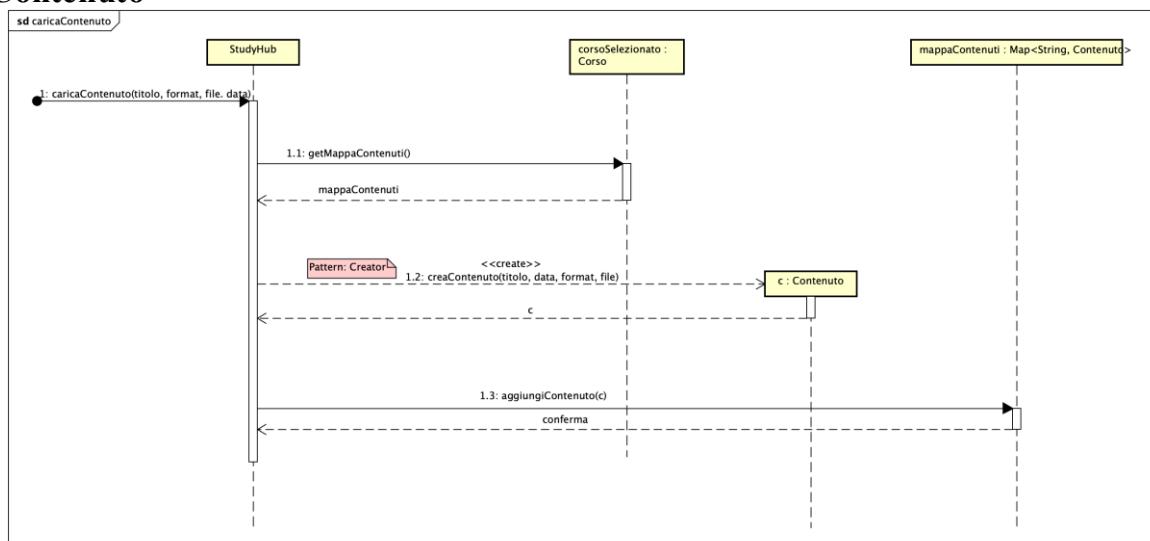


## UC5

### -selezionaCorsoCreato

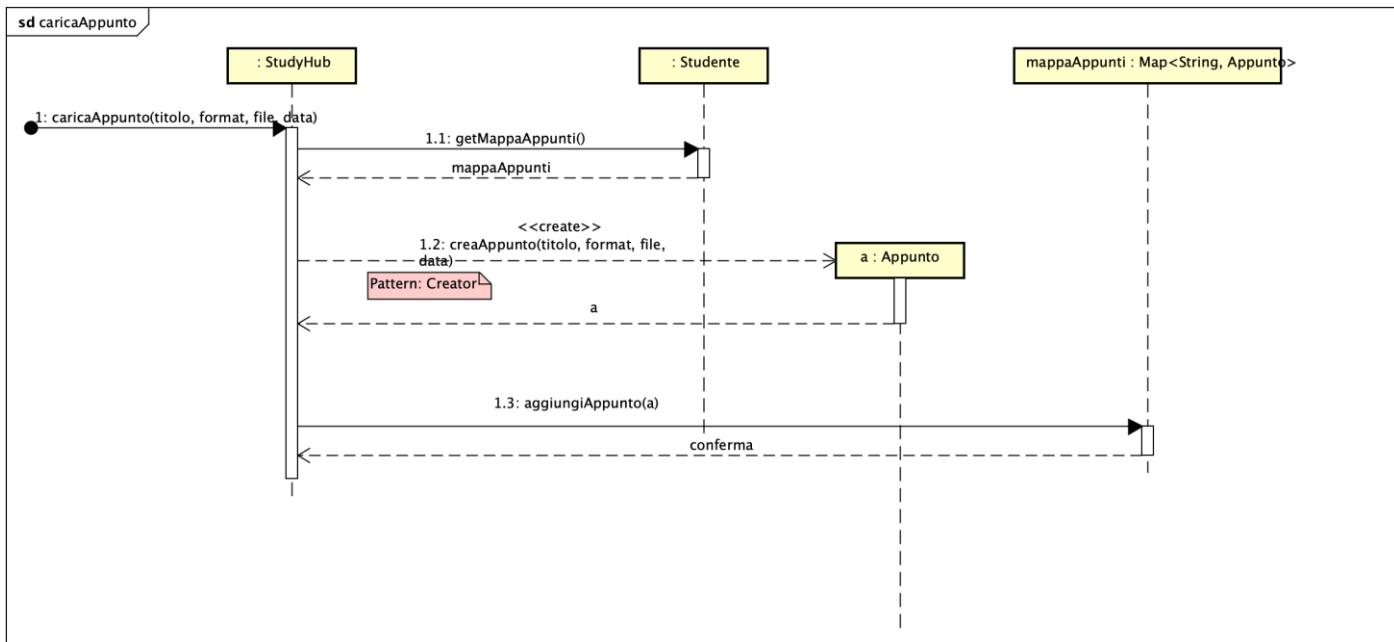


### -caricaContenuto



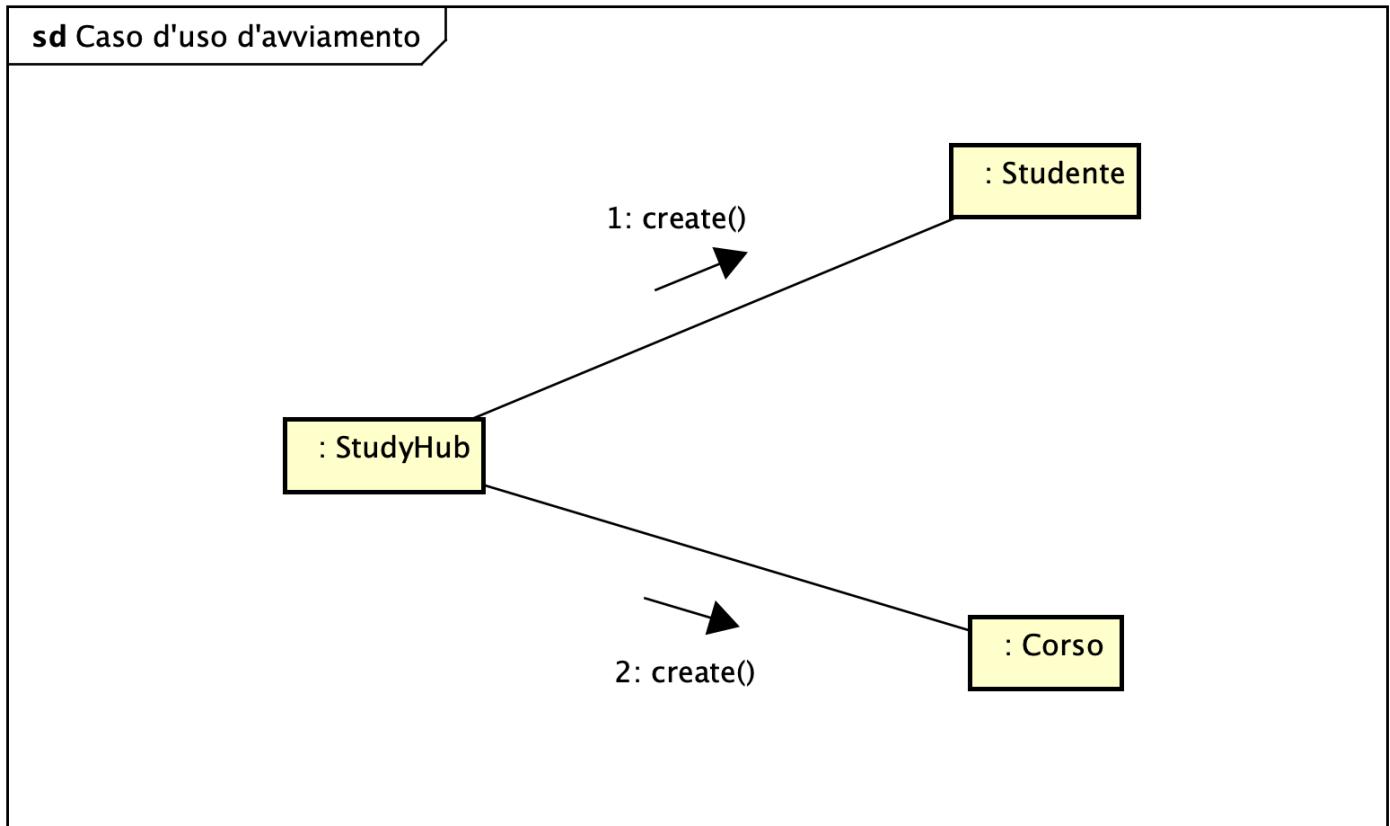
## UC6

### -caricaAppunto



### Caso d'uso di avviamento

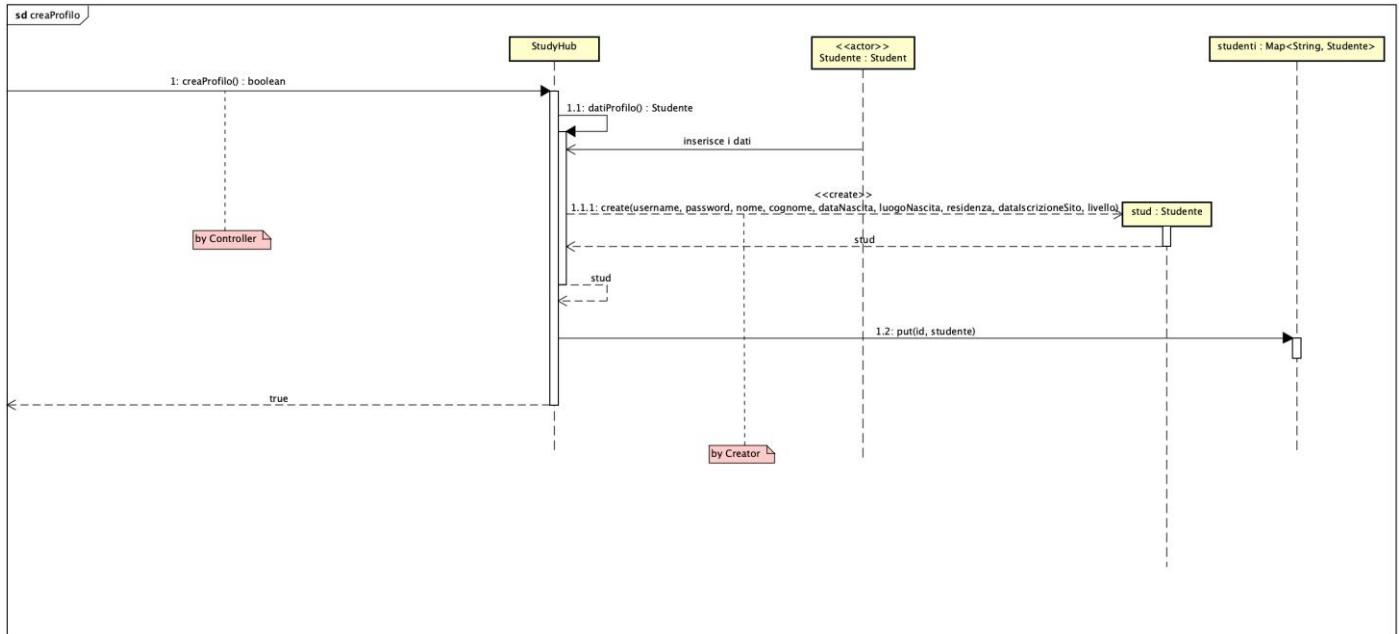
Per fare partire il sistema all'inizio del suo ciclo vitale è stato ideato un caso d'uso d'avviamento per creare degli studenti e dei corsi.



### Iterazione 2

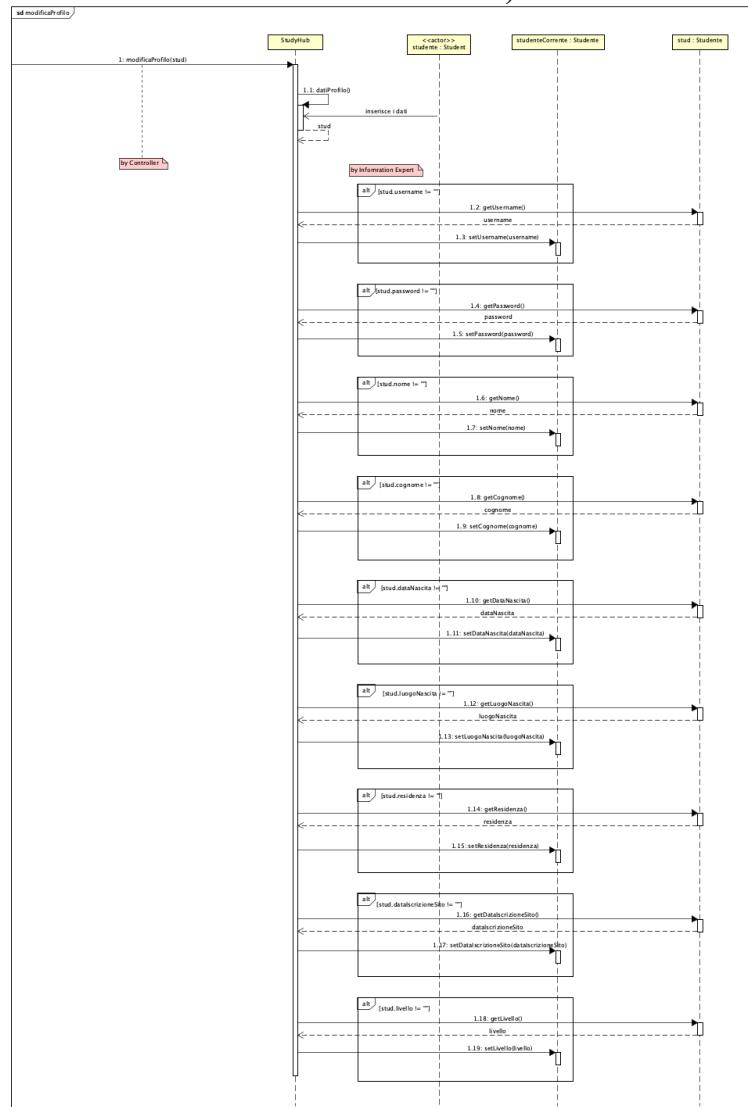
UC1

## -creaProfilo



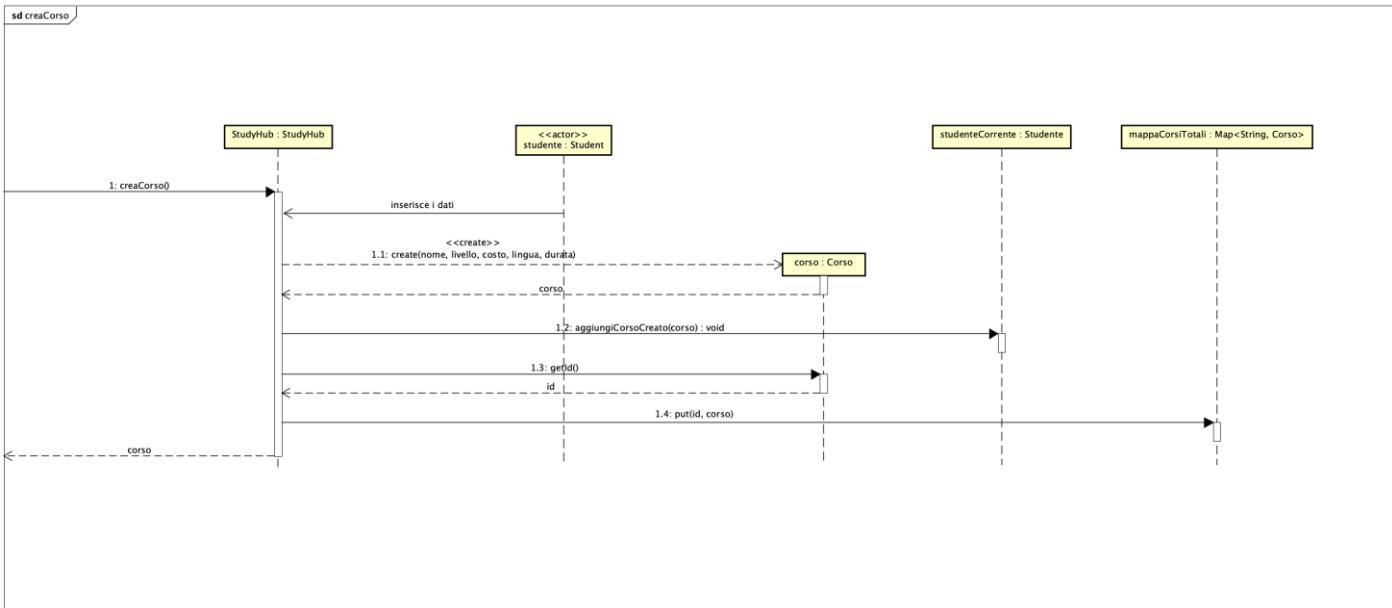
## UC2

### -modificaProfilo (è stato modificato l'ordine dei casi d'uso)



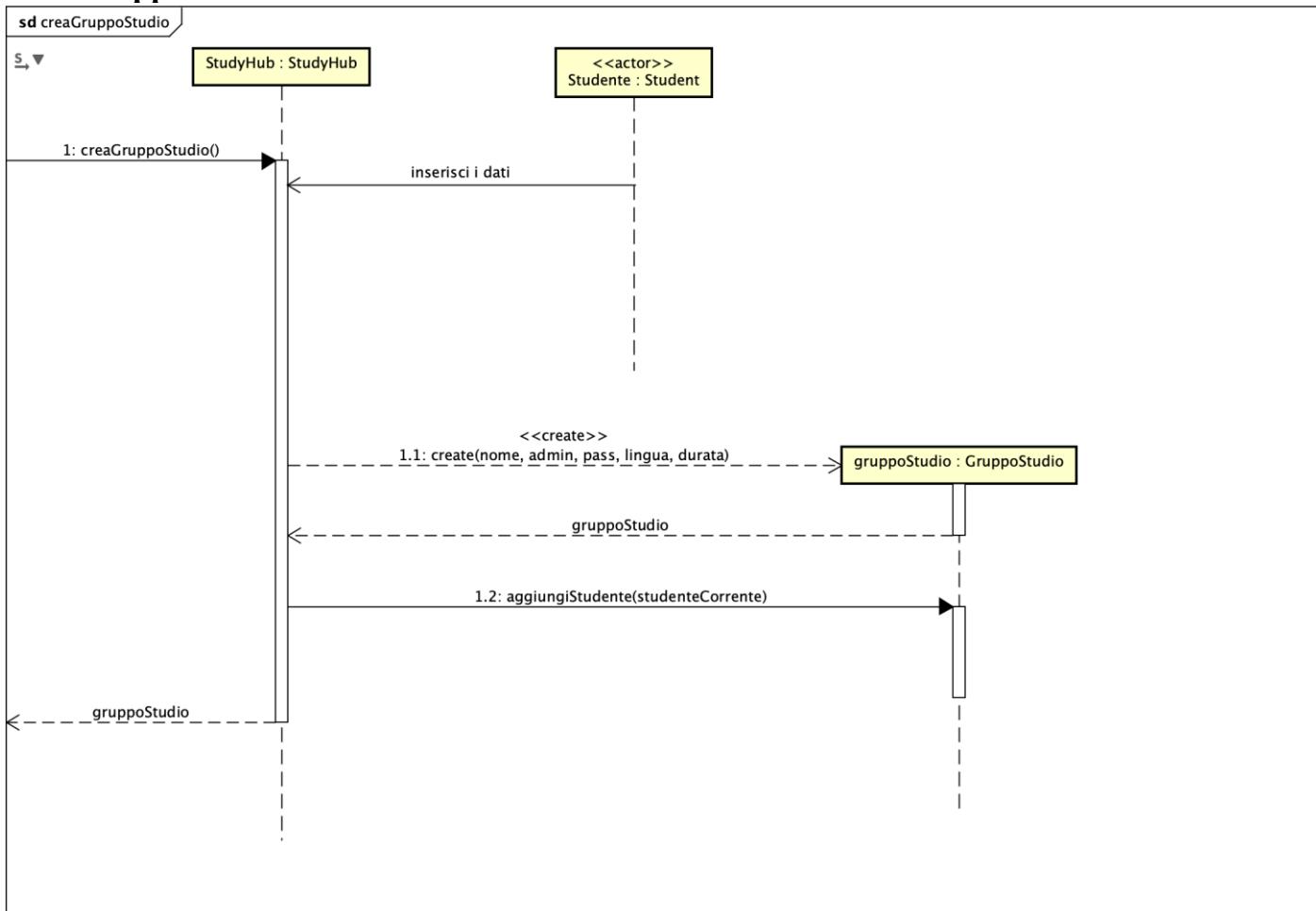
## UC3

### -creaCorso



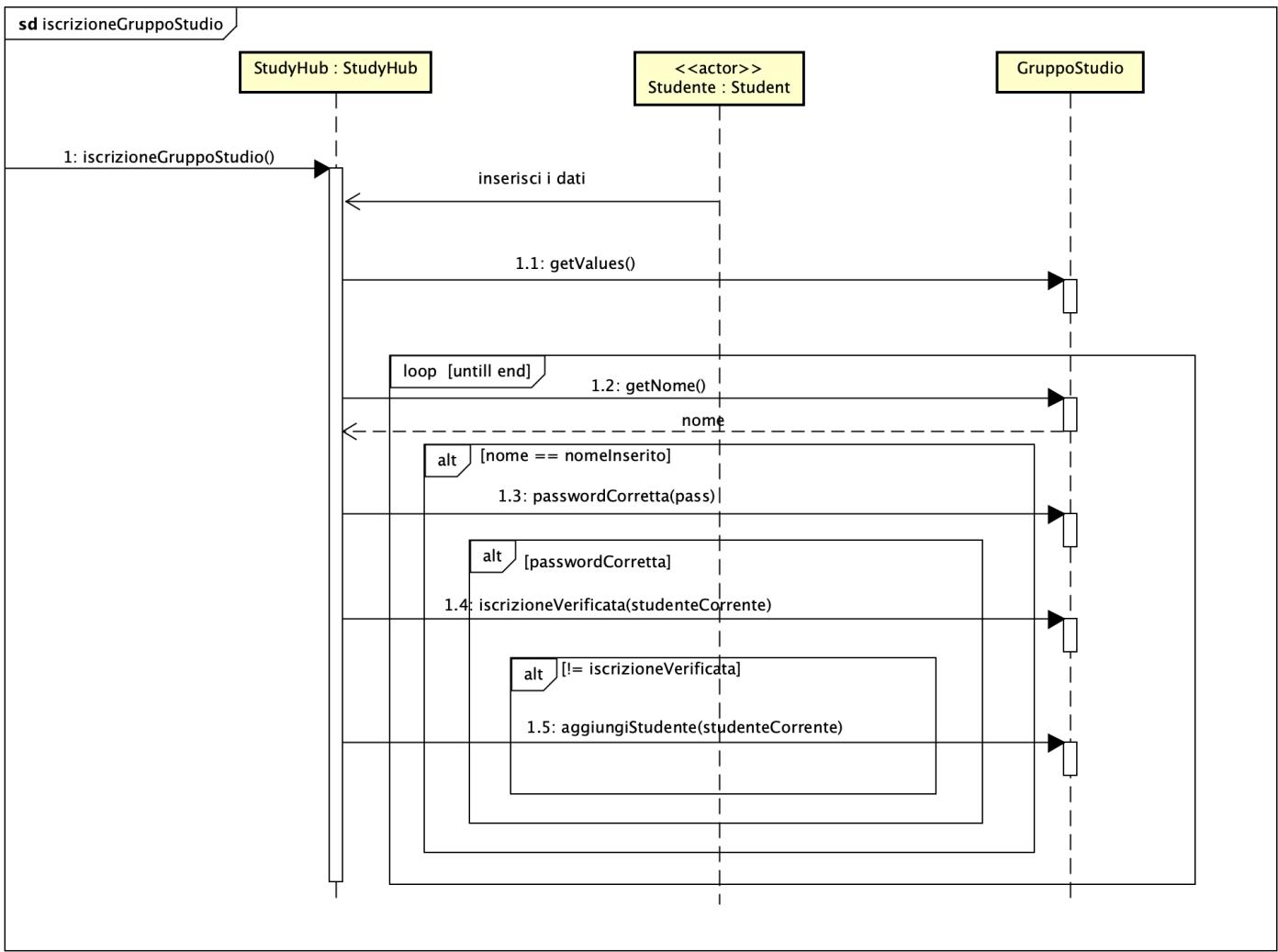
## UC5

-creaGruppoStudio



## UC6

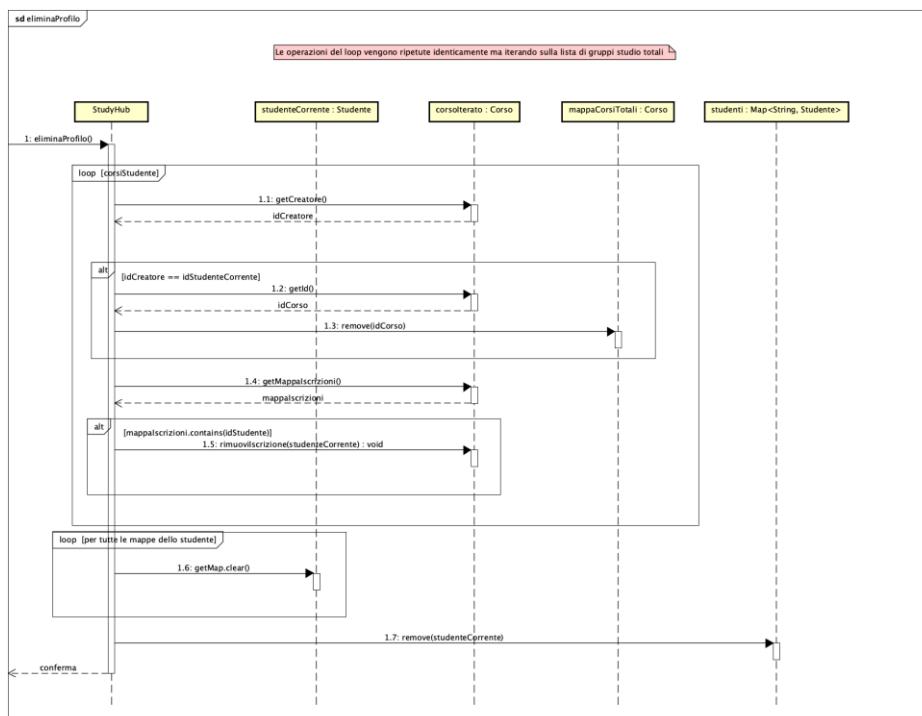
-iscrizioneGruppoStudio



## Iterazione 3

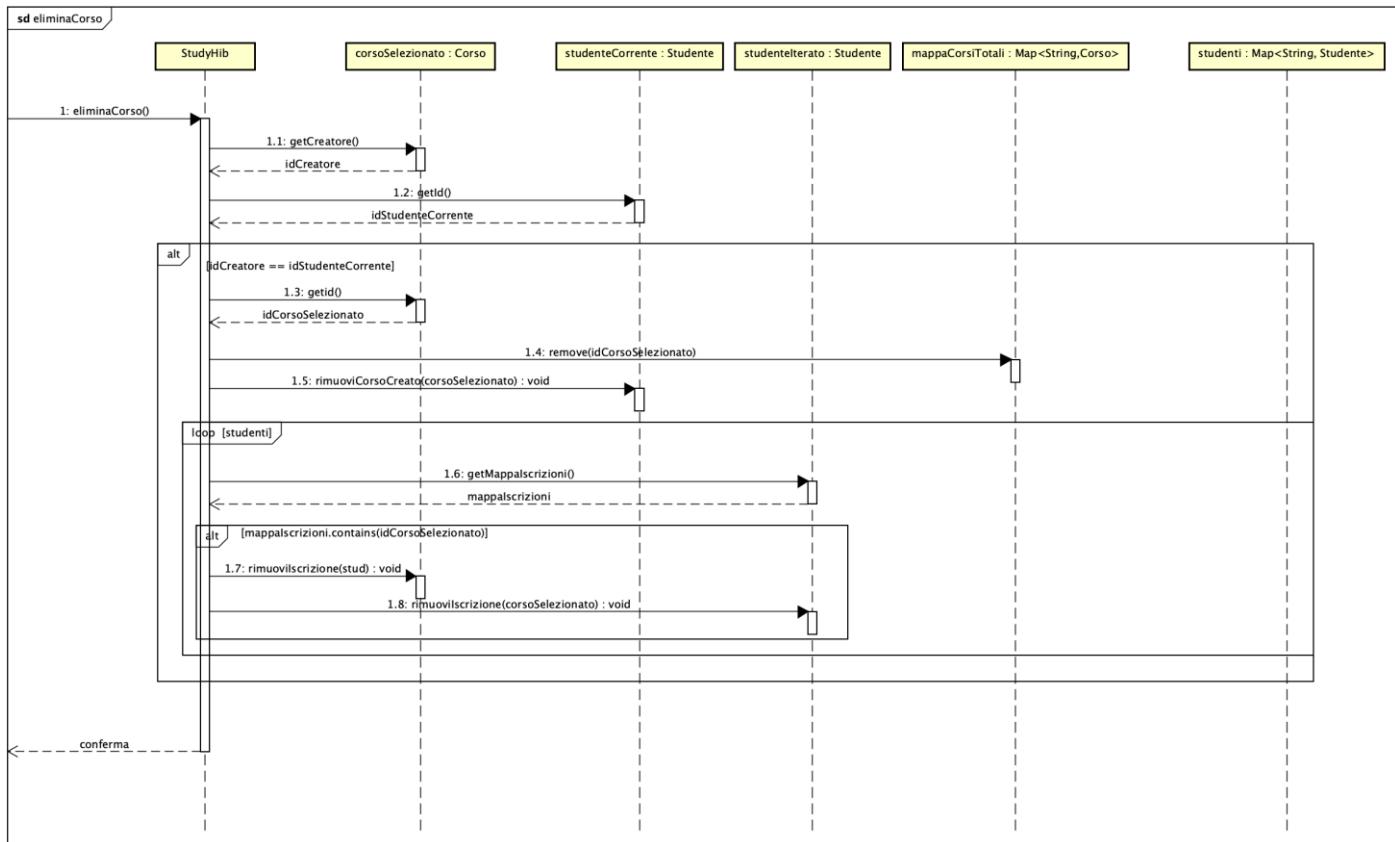
### UC1

-eliminaProfilo



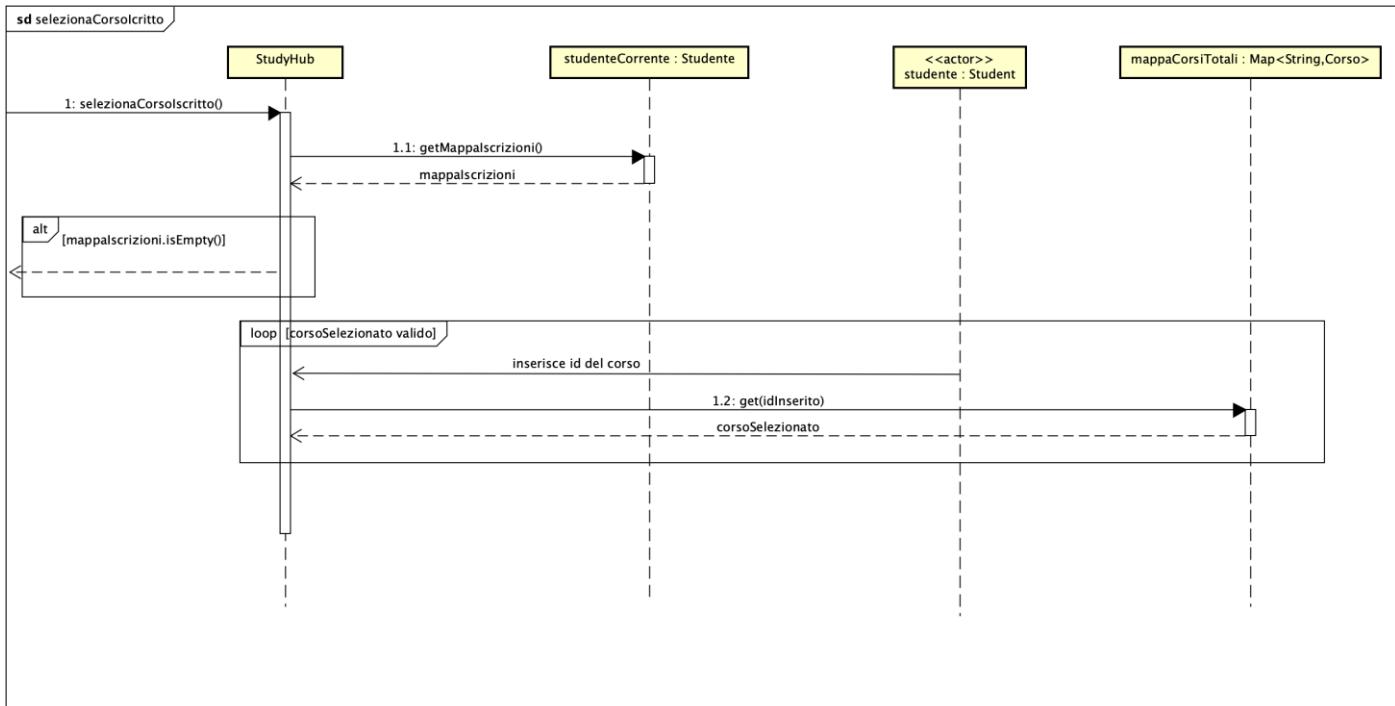
### UC3

-eliminaCorso

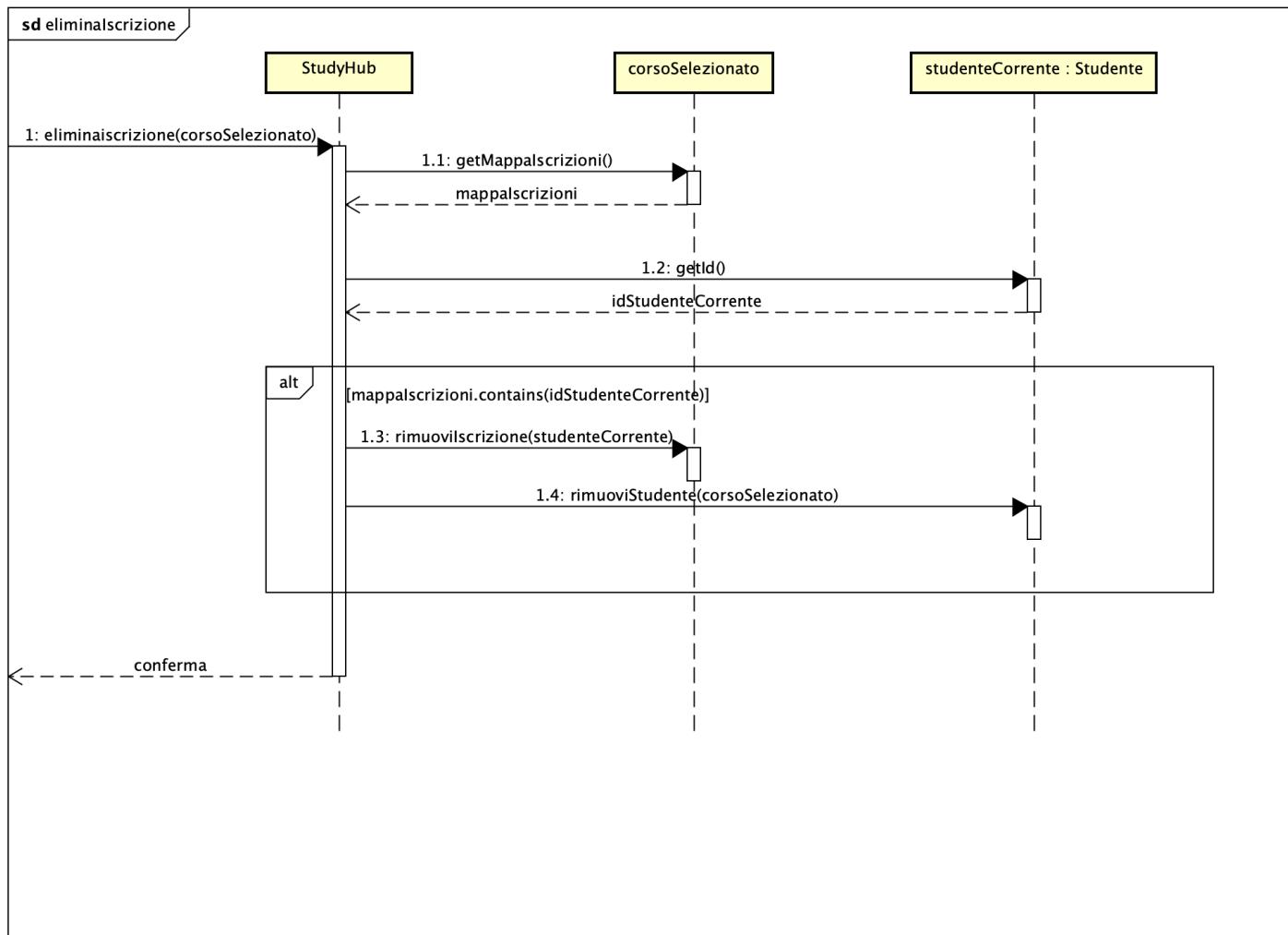


## UC4

### -selezionaCorsoliscritto

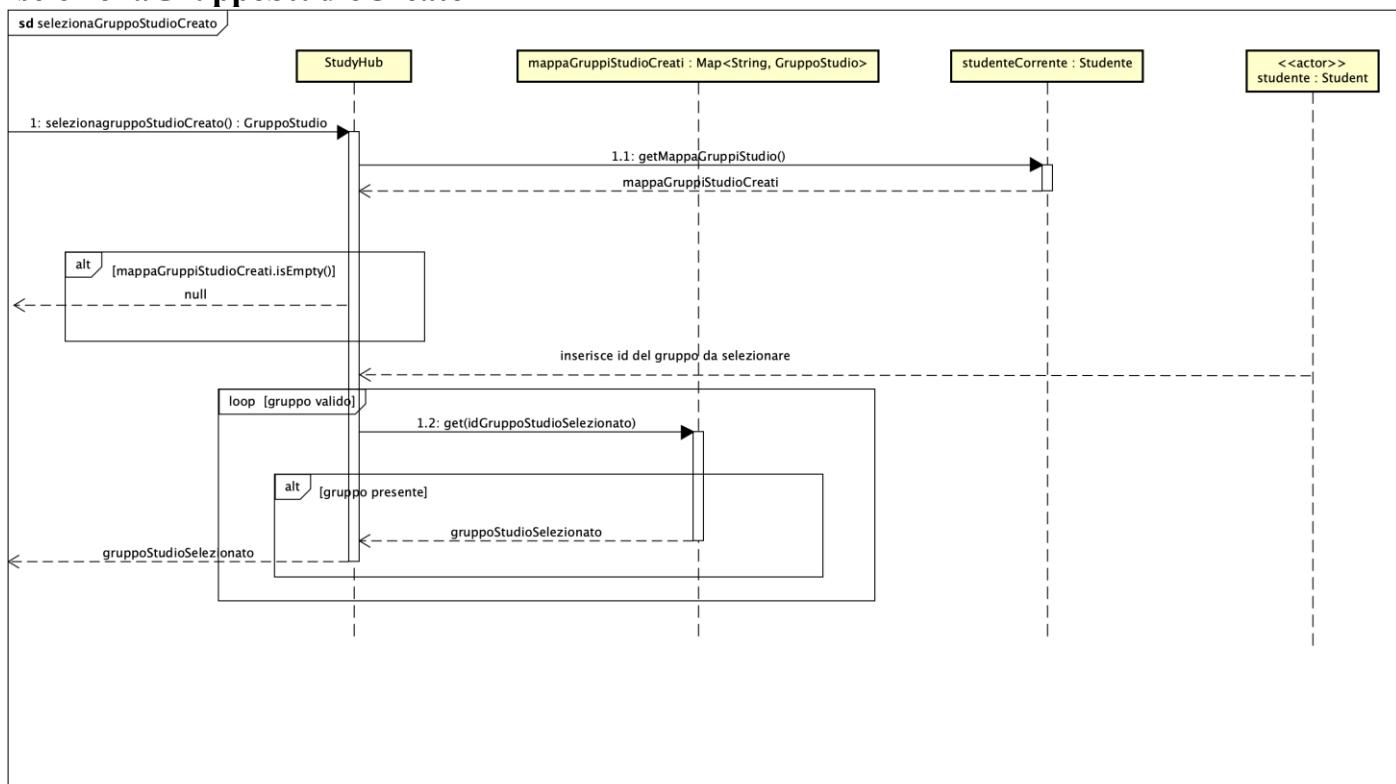


### -eliminalIscrizione

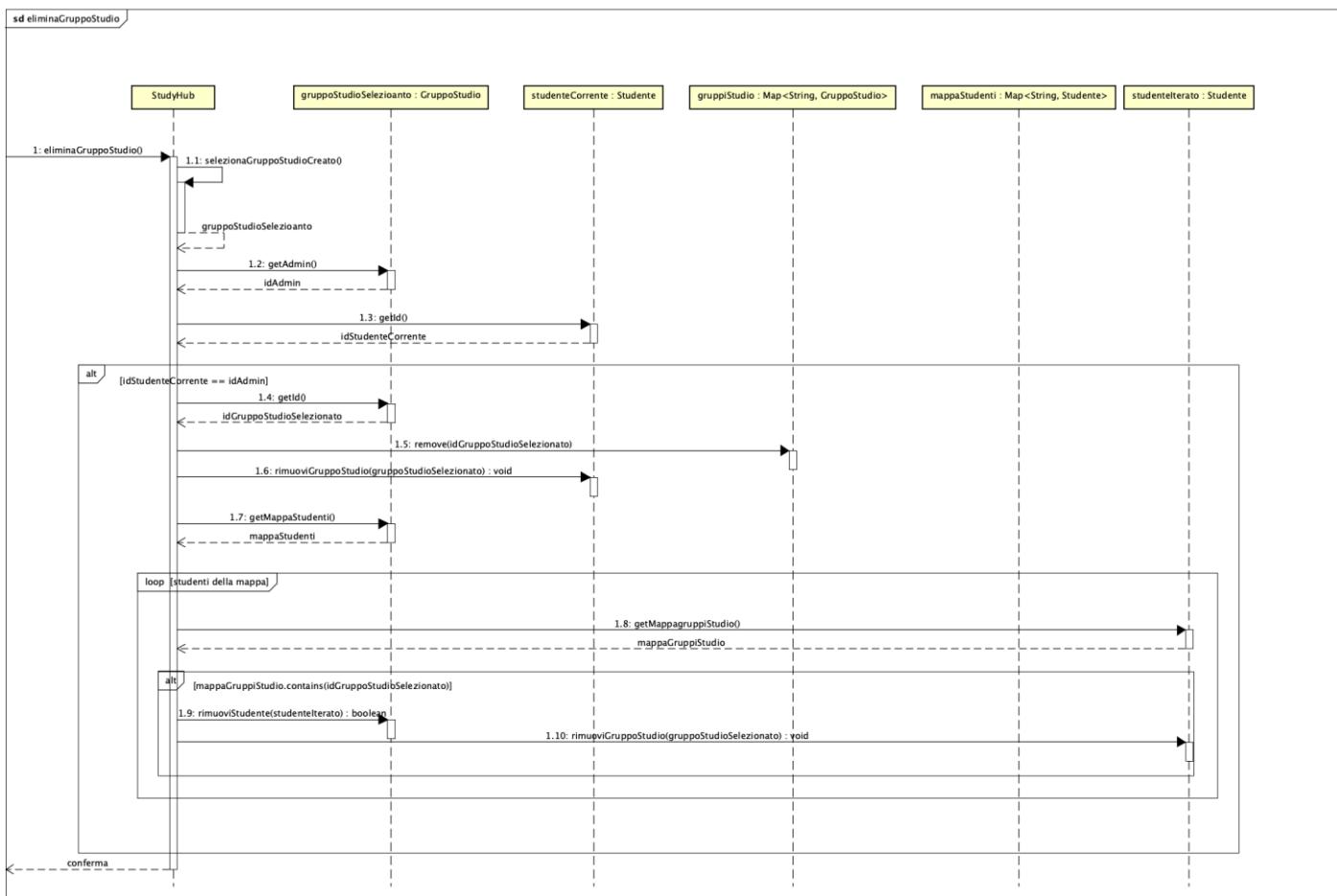


## UC5

### -selezionaGruppoStudioCreato

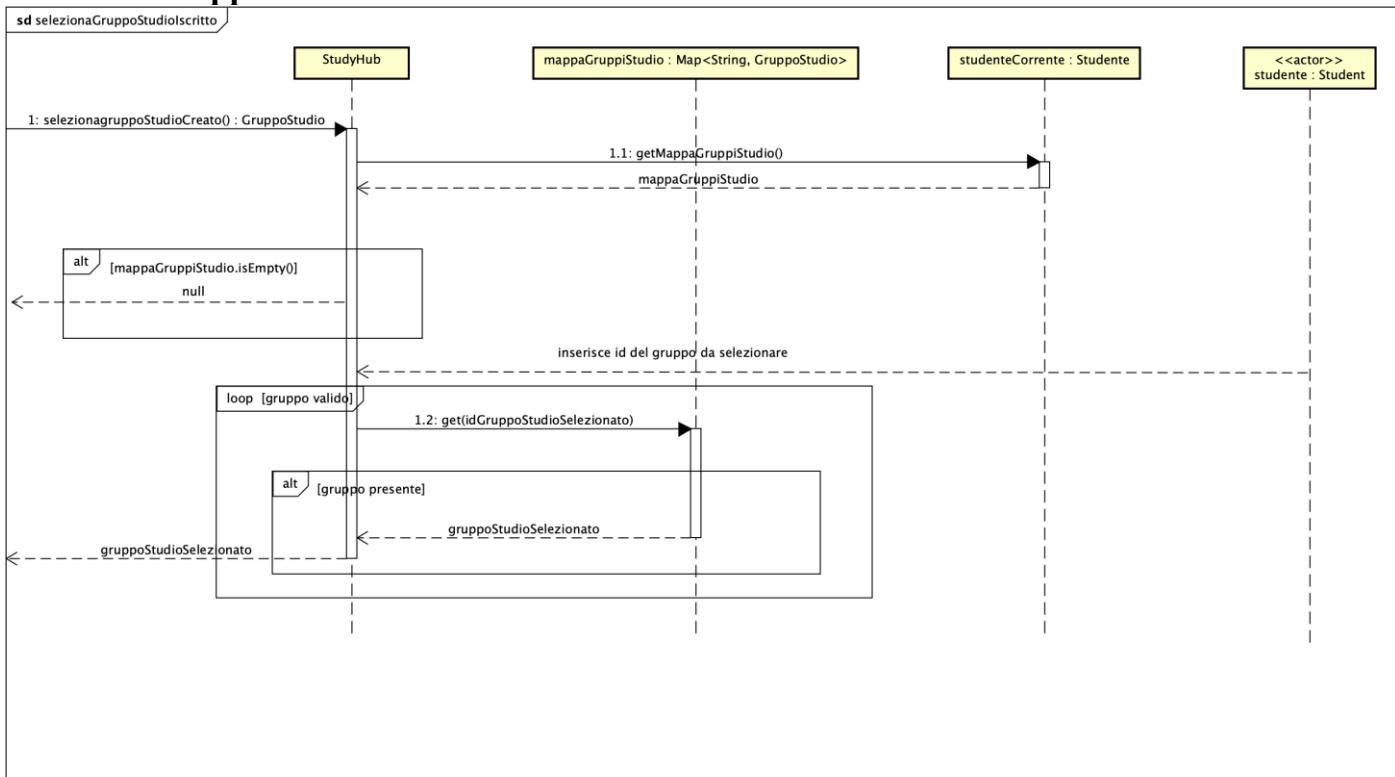


### -eliminaGruppoStudio

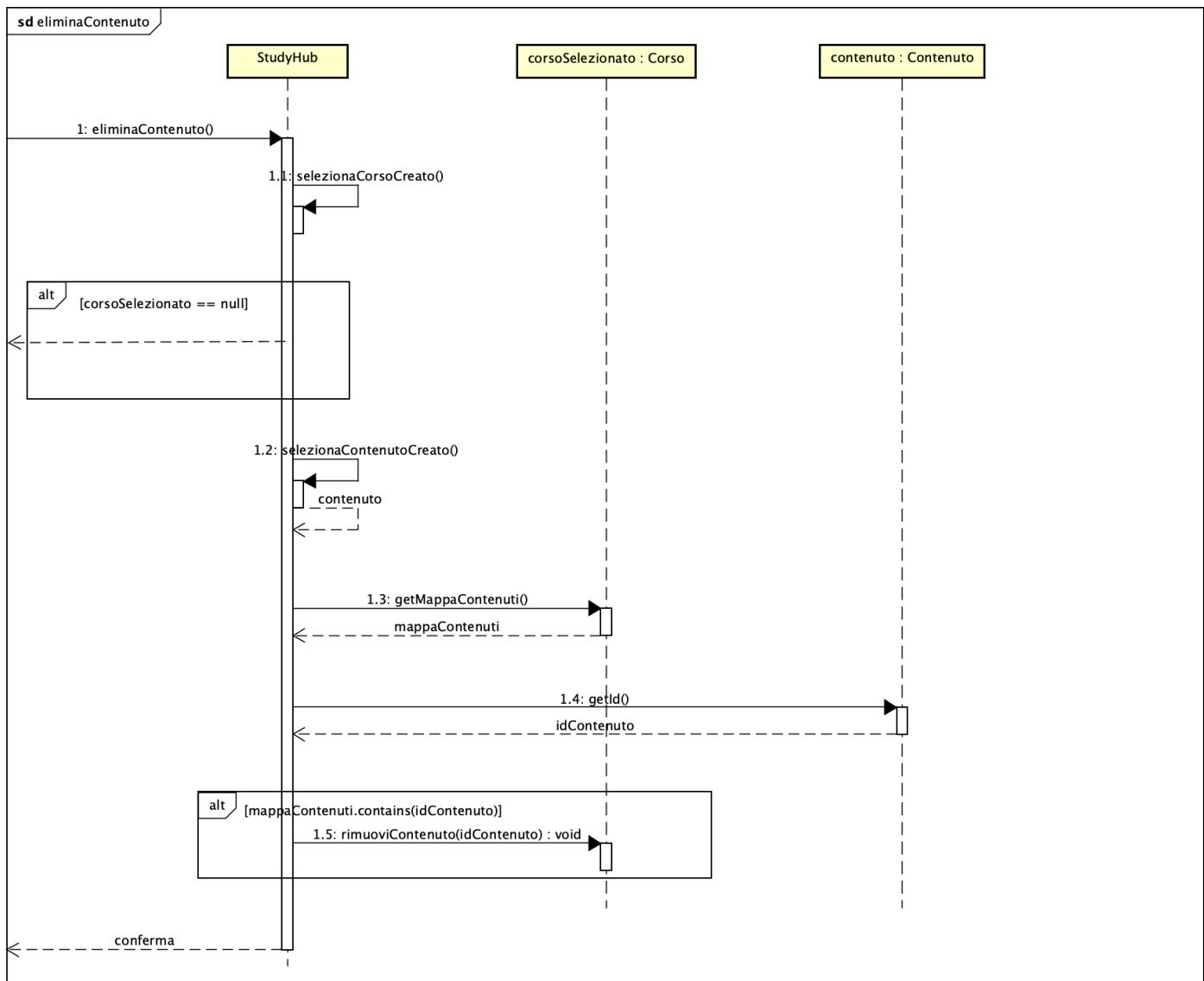


## UC6

### -selezionaGruppoStudioIscritto

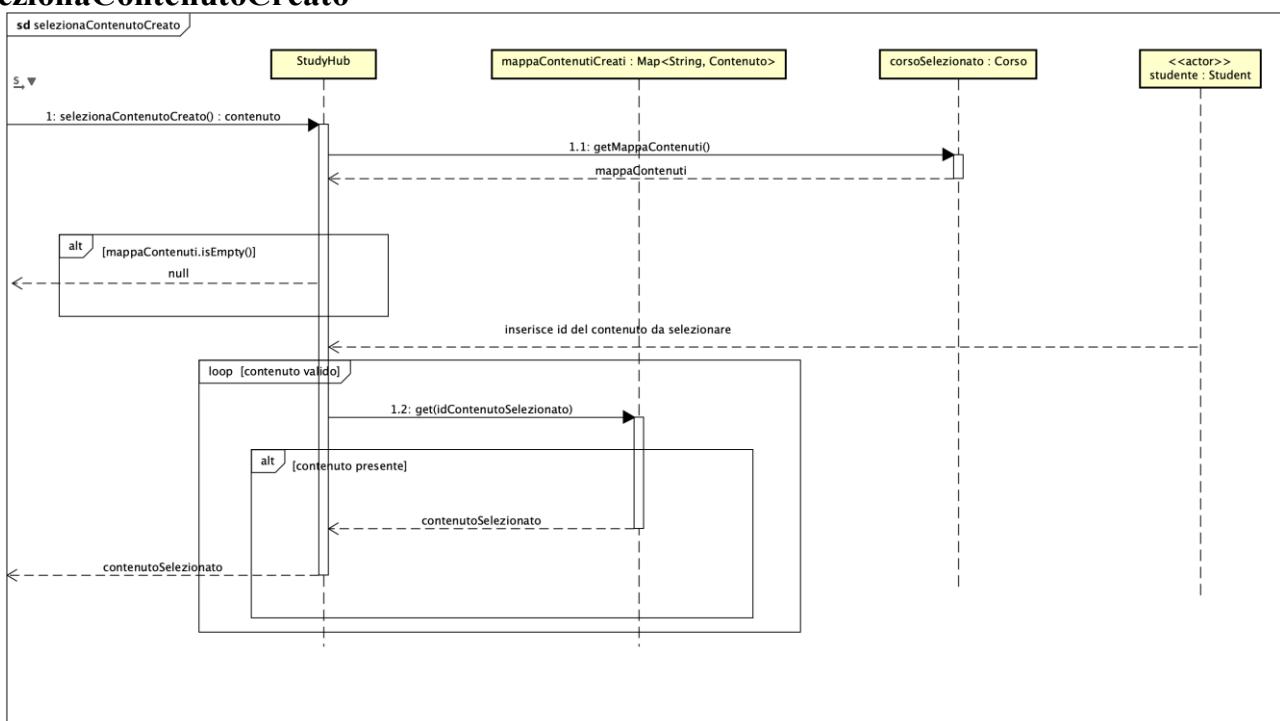


### -eliminaIscrizioneGruppoStudio

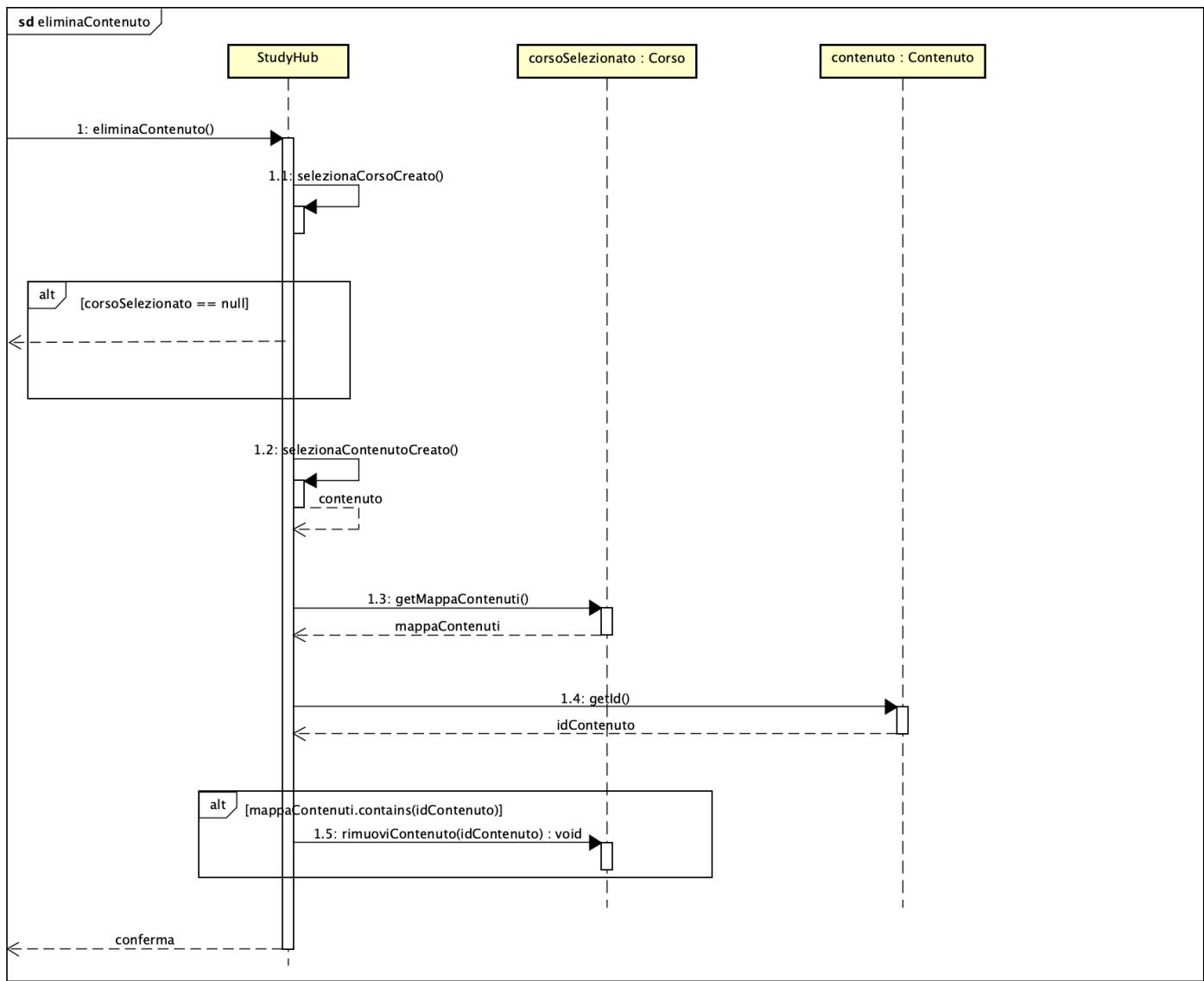


## UC7

### -selezionaContenutoCreato

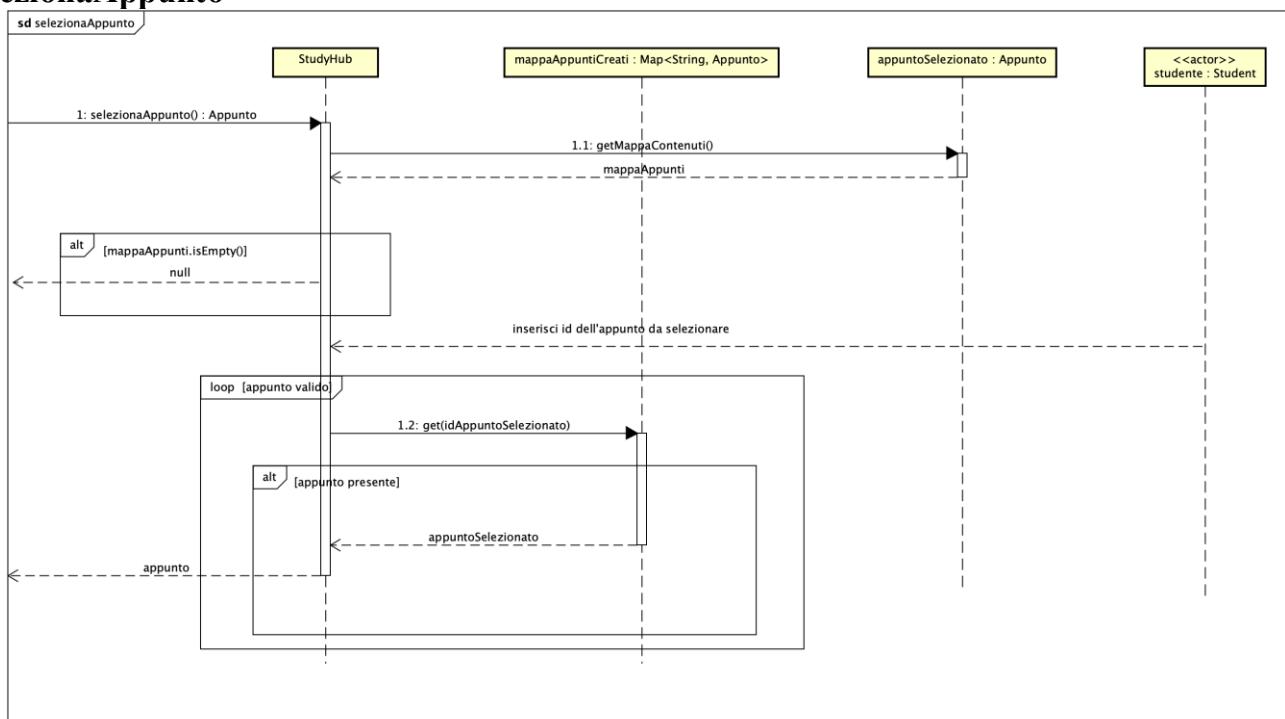


### -eliminaContenuto

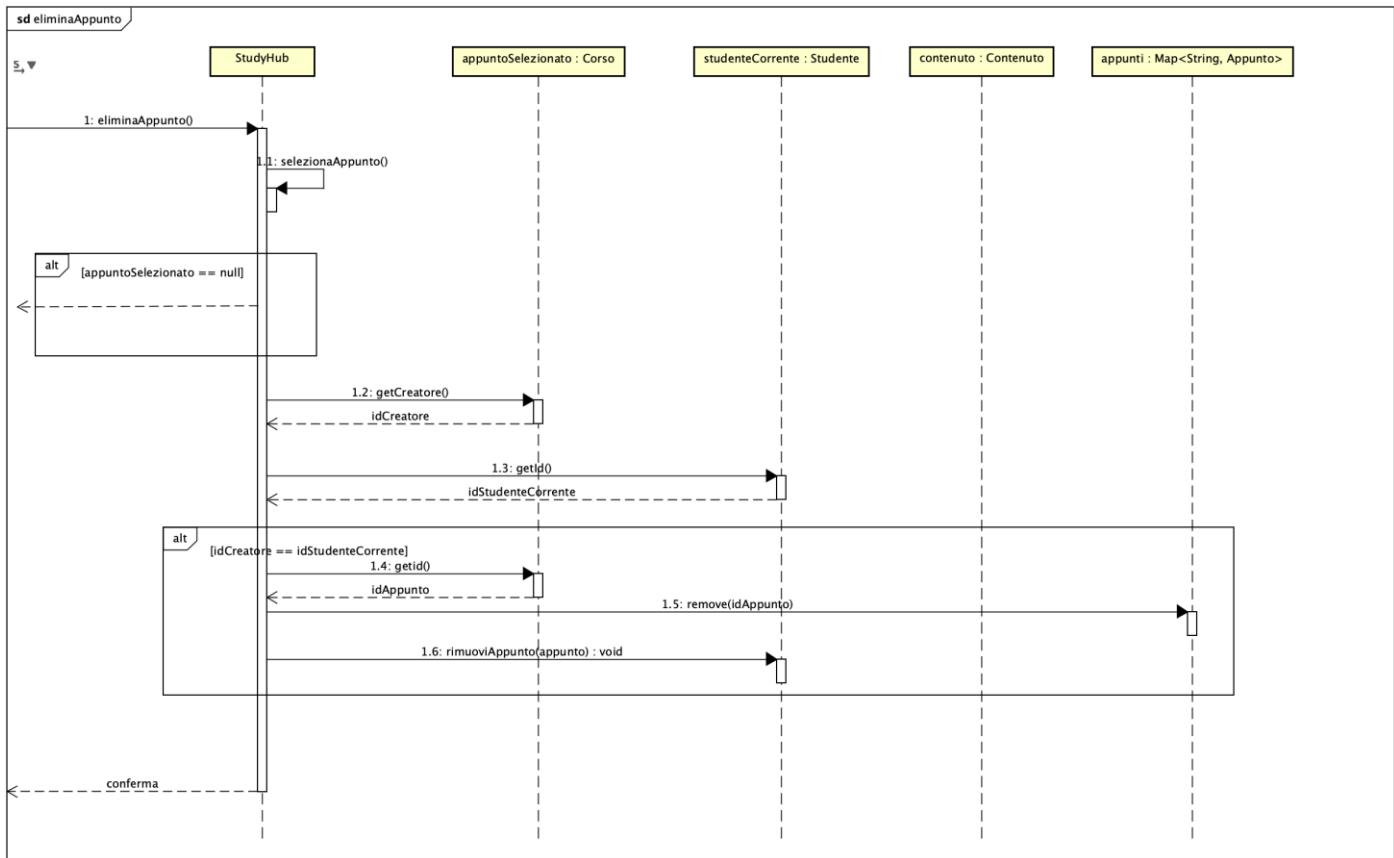


## UC8

### -selezionaAppunto



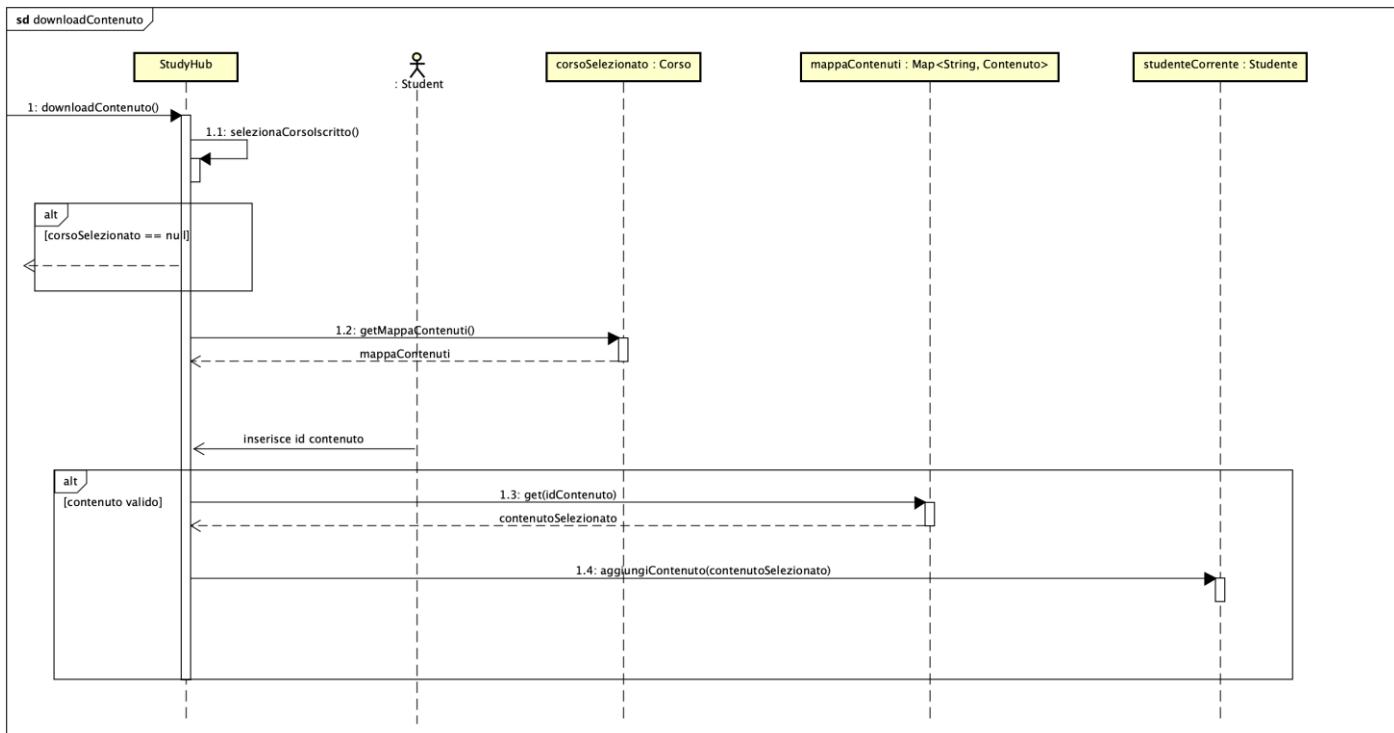
### -eliminaAppunto



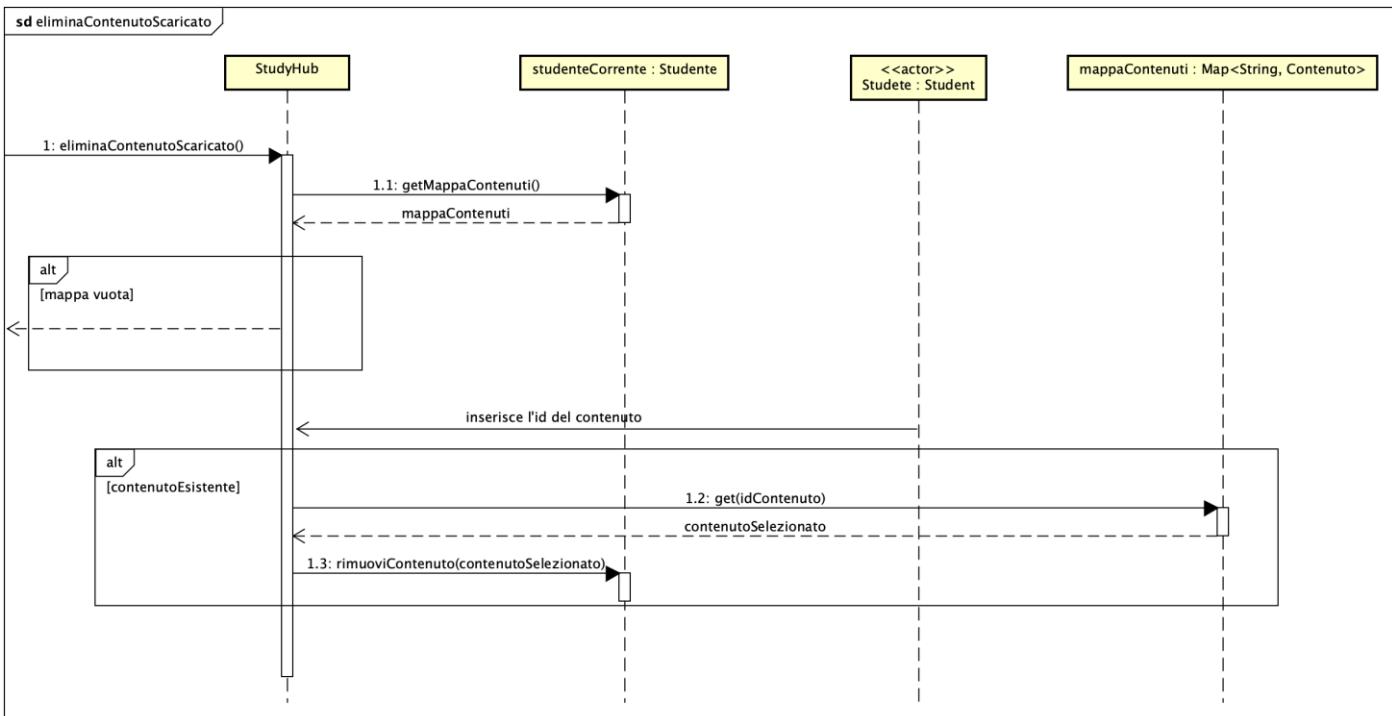
## Iterazione 4

UC9

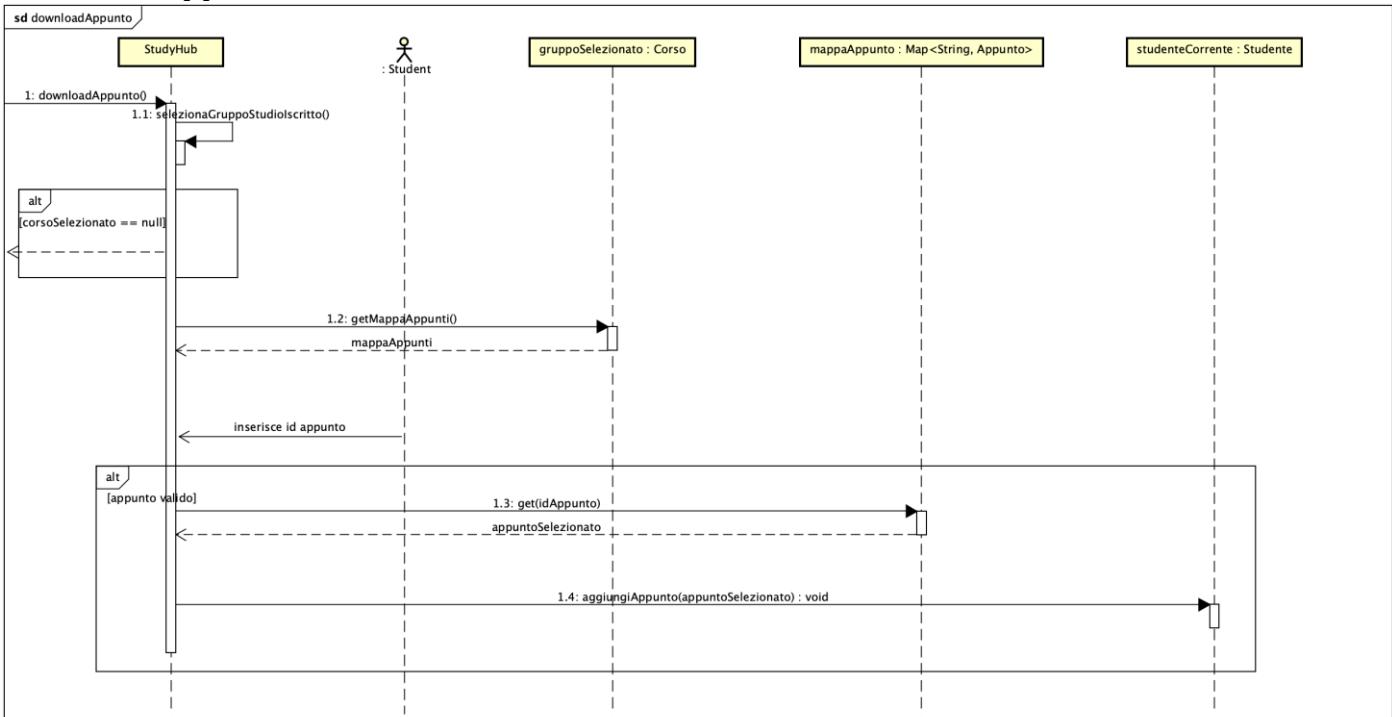
-downloadContenuto



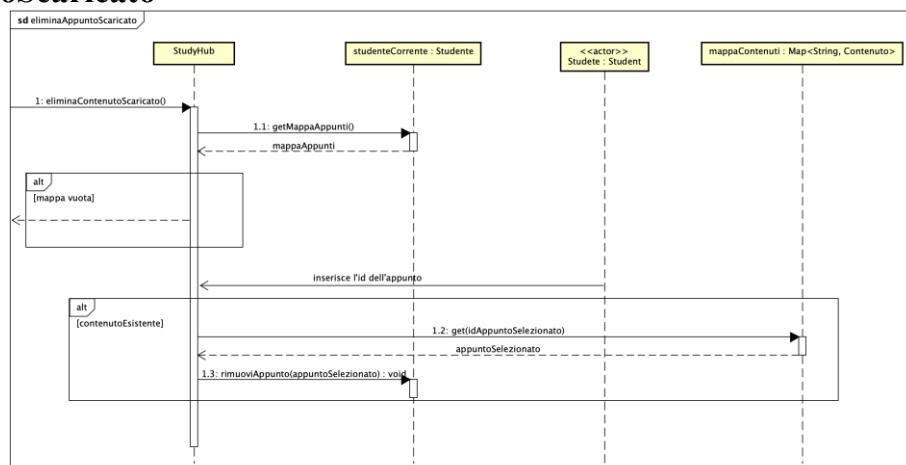
-eliminaContenutoScaricato



### -downloadAppunto



### -eliminaAppuntoScaricato



# **13. Testing**

## **Introduzione**

Il testing rappresenta un elemento fondamentale nello sviluppo di software affidabile ed efficiente. Non solo consente di ridurre i costi di manutenzione, ma garantisce anche che l'applicazione rispetti gli standard funzionali richiesti dal committente. Infatti, la probabilità di individuare eventuali malfunzionamenti cresce all'aumentare del numero di test eseguiti. Tuttavia, testare in maniera esaustiva un programma può risultare complesso, poiché le possibili combinazioni di input sono estremamente numerose e non sempre riproducibili in tempi brevi. Un processo di testing ben strutturato, dunque, diventa indispensabile per identificare comportamenti anomali e assicurare il corretto funzionamento del software.

I test possono essere suddivisi principalmente in due categorie:

- **Test Unitari (Unit Test):** Questi test verificano la correttezza delle singole componenti del codice, valutando ogni metodo rispetto ai risultati attesi. In ambiente Java, l'utilizzo di framework come JUnit permette di automatizzare e rendere efficiente questa fase di verifica.
- **Test Funzionali:** Questi test valutano il comportamento complessivo del sistema trattandolo come una “scatola nera”. In questo modo, si analizzano gli input e si controlla la correttezza degli output, assicurando che l'intero sistema funzioni come previsto.

Nell'ambito dell'applicazione sviluppata, si è scelto di concentrarsi principalmente sui test unitari, adottando un approccio Bottom-Up. Tale strategia prevede di testare prima le singole unità di codice, le quali, una volta integrate, costituiranno l'intero programma. Questo metodo garantisce un controllo dettagliato e accurato della correttezza del codice a ogni livello di sviluppo.

## **Individuazione dei casi di test e testing unitario**

### **TestStudyHub**

#### **-Test generali**

`testGeneralId()`:

- Scopo: Verificare la generazione di un identificativo univoco per il sistema.
- Realizzazione: Si invoca il metodo statico che genera l'ID e si asserisce che il valore restituito non sia nullo.

`testMenuOptionX()`:

- Scopo: Verificare che il metodo del menu riconosca e restituisca correttamente l'opzione X (un numero da 1 a 13).
- Realizzazione: Si simula l'input X e si confronta il valore di ritorno del metodo menu() con il numero X.

`testLogin()`:

- Scopo: Verificare che il processo di login funzioni correttamente con credenziali valide.
- Realizzazione: Si simula l'inserimento di username e password (“Mazzoldi” e “1111”) e si asserisce che il login restituisca true.

#### **-Test UC1**

`testCreaProfilo()`:

- Scopo: Verificare che la creazione di un nuovo profilo utente avvenga correttamente.

- Realizzazione: Si simulano i vari input per i dati del profilo; successivamente si controlla che il profilo creato venga registrato nel sistema e che il numero totale di studenti aumenti.

testControllaUsername():

- Scopo: Verificare il controllo sull'unicità dello username durante la creazione del profilo.
- Realizzazione: Simulando l'inserimento di dati con uno username già esistente, si asserisce che il metodo di creazione del profilo restituisca false, impedendo duplicazioni.

testEliminaProfilo():

- Scopo: Verificare che la cancellazione del profilo utente rimuova tutti i dati associati.
- Realizzazione: Dopo aver impostato un profilo utente con corsi, gruppi, appunti, iscrizioni e pagamenti, si esegue l'eliminazione e si controlla la rimozione di tutte le associazioni relative.

## -Test UC2

testModificaProfilo():

- Scopo: Assicurare che le modifiche ai dati del profilo vengano applicate correttamente.
- Realizzazione: Si simula l'inserimento di nuovi dati (username, nome, cognome, ecc.) e si verifica che il profilo aggiornato rispecchi le modifiche apportate.

## -Test UC3

testCreaCorso():

- Scopo: Verificare che il processo di creazione di un nuovo corso avvenga correttamente.
- Realizzazione: Simulando l'inserimento dei dati del corso, si verifica che il corso venga aggiunto sia nella mappa dei corsi creati dallo studente sia in quella dei corsi totali.

testControllaNomeCorso():

- Scopo: Verificare il controllo sull'unicità e la validità del nome del corso.
- Realizzazione: Si simulano tentativi di creazione di un corso con un nome già esistente e con caratteri non validi, verificando che il corso non venga creato e il numero totale dei corsi non cambi.

testEliminaCorso():

- Scopo: Verificare che l'eliminazione di un corso rimuova il corso e tutti i relativi dati (contenuti e iscrizioni).
- Realizzazione: Impostando un corso selezionato, si simula la sua eliminazione e si controlla che venga rimosso dalla mappa dei corsi dello studente e del sistema, insieme a tutte le associazioni.

## -Test UC4

testCercaCorso():

- Scopo: Verificare la funzionalità di ricerca dei corsi in base a criteri multipli.
- Realizzazione: Si crea una mappa con diversi corsi e si simulano vari input (nome, livello, creatore, lingua, id) per filtrare la ricerca, verificando che i corsi restituiti corrispondano ai criteri inseriti.

testSelezionaCorso():

- Scopo: Verificare che dalla lista dei corsi trovati sia possibile selezionare correttamente un corso.
- Realizzazione: Simulando l'input dell'ID di un corso all'interno della mappa dei corsi cercati, si asserisce che il corso selezionato corrisponda a quello atteso.

testIscrizioneCorsoGratuito():

- Scopo: Verificare che l'iscrizione a un corso gratuito avvenga correttamente.

- Realizzazione: Simulando l'inserimento dei dati per l'iscrizione a un corso gratuito, si controlla che il corso venga associato allo studente e che i contatori (numero di studenti e corsi) vengano aggiornati.

testControllaIscrizione():

- Scopo: Assicurare che il sistema non permetta iscrizioni duplicate a un corso.
- Realizzazione: Dopo aver verificato l'iscrizione già esistente, si controlla che il numero di iscrizioni rimanga invariato.

testPagamentoIscrizione():

- Scopo: Verificare il corretto funzionamento del pagamento per l'iscrizione a un corso a pagamento.
- Realizzazione: Si crea un dato di pagamento per lo studente, si simula l'inserimento del numero di carta e si verifica che l'iscrizione e il pagamento vengano registrati correttamente.

testAggiungiIscrizione():

- Scopo: Verificare l'aggiunta manuale di un'iscrizione a un corso.
- Realizzazione: Creando un'istanza di iscrizione e aggiungendola tramite il metodo apposito, si asserisce che la stessa venga registrata nelle mappe dello studente e del corso, con l'aggiornamento dei relativi conteggi.

testSelezionaCorsoIscritto():

- Scopo: Verificare che lo studente possa selezionare correttamente un corso a cui è già iscritto.
- Realizzazione: Simulando l'input dell'ID di un corso a cui è già iscritto, si verifica che il corso selezionato corrisponda a quello previsto.

testEliminaIscrizione():

- Scopo: Verificare l'annullamento dell'iscrizione a un corso.
- Realizzazione: Simulando l'eliminazione dell'iscrizione (tramite input dell'ID del corso), si controlla che il corso venga rimosso dalle mappe di iscrizione dello studente e del corso, con decremento dei contatori.

## -Test UC5

testCreaGruppoStudio():

- Scopo: Verificare la creazione di un nuovo gruppo di studio.
- Realizzazione: Simulando l'inserimento dei dati del gruppo (nome, password, lingua, capacità), si verifica che il gruppo venga registrato sia nel sistema che nel profilo dello studente, con lo studente impostato come amministratore.

testControllaNomeGruppo():

- Scopo: Assicurare che non sia possibile creare gruppi di studio con nomi duplicati.
- Realizzazione: Si simula la creazione di un gruppo con un nome già esistente e si verifica che il metodo restituisca null, mantenendo invariato il numero dei gruppi.

testSelezionaGruppoStudioCreato():

- Scopo: Verificare che lo studente possa selezionare correttamente un gruppo di studio da quelli creati da lui.
- Realizzazione: Si individua un gruppo (ad esempio "Gruppo1") e, simulando l'input del suo ID, si asserisce che il gruppo selezionato corrisponda a quello atteso.

testEliminaGruppoStudio():

- Scopo: Verificare la rimozione completa di un gruppo di studio dal sistema.
- Realizzazione: Dopo aver individuato un gruppo, si simula la sua eliminazione e si controlla che venga rimosso sia dalla mappa dei gruppi del sistema sia da quella dello studente e dei relativi membri.

### **-Test UC6**

testIscrizioneGruppoStudio():

- Scopo: Verificare che lo studente possa iscriversi a un gruppo di studio esistente.
- Realizzazione: Simulando l'inserimento dei dati richiesti per l'iscrizione, si verifica che lo studente venga aggiunto alla mappa del gruppo e che il conteggio degli iscritti aumenti.

testSelezionaGruppoStudioIscritto():

- Scopo: Verificare che lo studente possa selezionare correttamente un gruppo di studio a cui è già iscritto.
- Realizzazione: Individuando un gruppo (ad esempio "Gruppo4") e simulando l'input del suo ID, si controlla che il gruppo selezionato sia quello previsto.

testSelezionaGruppoStudio():

- Scopo: Verificare la corretta selezione di un gruppo di studio, sia creato che iscritto.
- Realizzazione: Simulando l'input dell'ID di un gruppo, in due casi differenti (ad es. "Gruppo4" e "Gruppo3"), si asserisce che il gruppo selezionato corrisponda a quello atteso.

testEliminaIscrizioneGruppoStudio():

- Scopo: Verificare l'annullamento dell'iscrizione ad un gruppo di studio.
- Realizzazione: Simulando l'eliminazione (tramite input dell'ID del gruppo), si verifica che lo studente venga rimosso dalla mappa degli iscritti del gruppo e che i contatori vengano aggiornati correttamente.

### **-Test UC7**

testSelezionaCorsoCreato():

- Scopo: Verificare che lo studente possa selezionare un corso da quelli da lui creati.
- Realizzazione: Simulando l'input dell'ID di un corso creato, si controlla che il corso selezionato corrisponda a quello atteso.

testCaricaContenuto():

- Scopo: Verificare il caricamento di un nuovo contenuto in un corso.
- Realizzazione: Simulando l'inserimento dei dati (titolo, tipo, file, durata) per un contenuto, si asserisce che esso venga aggiunto alla mappa dei contenuti del corso selezionato.

testControllaContenuto():

- Scopo: Assicurare che vengano accettati solo contenuti con dati validi.
- Realizzazione: Si simulano tentativi di caricamento con tipo di file non consentito o dimensione non corretta, verificando che il metodo restituisca null e non modifichi la mappa dei contenuti.

testEliminaContenuto():

- Scopo: Verificare che un contenuto possa essere eliminato correttamente da un corso.
- Realizzazione: Simulando l'eliminazione (tramite input dell'ID del corso e del contenuto), si controlla che il contenuto venga rimosso dalla mappa e che le eventuali associazioni (come iscrizioni) vengano aggiornate.

### **-Test UC8**

#### **testCaricaAppunto():**

- Scopo: Verificare il caricamento di un nuovo appunto da parte dello studente.
- Realizzazione: Simulando l'inserimento dei dati (titolo, tipo, file, durata) per un appunto, si asserisce che esso venga aggiunto alla mappa degli appunti dello studente.

#### **testControllaAppunto():**

- Scopo: Assicurare che il caricamento di un appunto avvenga solo con dati validi.
- Realizzazione: Si simula un tentativo di caricamento con dati non validi (ad esempio tipo errato) e si verifica che il metodo restituisca null senza modificare la mappa degli appunti.

#### **testEliminaAppunto():**

- Scopo: Verificare la rimozione di un appunto dal profilo dello studente.
- Realizzazione: Simulando l'input dell'ID di un appunto, si controlla che questo venga eliminato dalla mappa degli appunti dello studente.

#### **testAppuntoGruppoStudio():**

- Scopo: Verificare il caricamento e la successiva eliminazione di appunti all'interno di un gruppo di studio.
- Realizzazione: Si simulano due casi: l'aggiunta di un appunto esistente e la creazione di un nuovo appunto nel gruppo, per poi eliminarli (in modalità diverse) e verificare l'aggiornamento delle mappe sia nel gruppo che dello studente.

### **-Test UC9**

#### **testDownloadContenuto():**

- Scopo: Verificare il download di un contenuto da un corso e la sua successiva rimozione.
- Realizzazione: Simulando il download (tramite input dell'ID del corso e del contenuto), si asserisce che il contenuto venga aggiunto alla mappa dei contenuti scaricati dello studente e, dopo la simulazione dell'eliminazione, che venga rimosso.

#### **testDownloadAppunto():**

- Scopo: Verificare il download di un appunto da un gruppo di studio e la corretta gestione della sua rimozione.
- Realizzazione: Si simula il download di un appunto da un gruppo (con preliminare caricamento nel gruppo) da parte di un secondo studente, verificando l'aggiunta alla mappa degli scaricati e successivamente la sua rimozione.

## **TestStudente**

#### **testCreaDatiPagamento():**

- Scopo: Verificare la creazione e l'associazione dei dati di pagamento a uno studente.
- Realizzazione: Simulando l'inserimento di metodo, numero carta, nome e cognome, si controlla che i dati di pagamento vengano creati e registrati nella mappa dello studente.

#### **testUsaDatiPagamento():**

- Scopo: Verificare il corretto recupero dei dati di pagamento già registrati.
- Realizzazione: Dopo aver creato i dati di pagamento, si simula l'inserimento del numero di carta e si verifica che il metodo restituisca i dati associati correttamente.

#### **testAggiungiAppunto():**

- Scopo: Verificare l'aggiunta di un appunto alla raccolta degli appunti dello studente.

- Realizzazione: Creando un appunto e aggiungendolo alla mappa, si asserisce che il numero di appunti aumenti e che l'appunto sia registrato.

testRimuoviAppunto():

- Scopo: Assicurare che un appunto possa essere rimosso correttamente dalla mappa dello studente.
- Realizzazione: Dopo aver aggiunto un appunto, si esegue la rimozione e si verifica che la mappa venga aggiornata e che l'appunto non sia più presente.

testAggiungiAppuntoScaricato():

- Scopo: Verificare l'aggiunta di un appunto scaricato alla relativa mappa dello studente.
- Realizzazione: Creando un appunto scaricato e aggiungendolo, si controlla che questo venga correttamente registrato nella mappa degli appunti scaricati.

testRimuoviAppuntoScaricato():

- Scopo: Verificare la rimozione di un appunto scaricato dalla mappa dello studente.
- Realizzazione: Dopo aver aggiunto un appunto scaricato, si procede alla sua rimozione e si asserisce che la mappa venga aggiornata.

testVerificaPassword():

- Scopo: Controllare che il sistema verifichi correttamente la corrispondenza della password dello studente.
- Realizzazione: Si testa il metodo di verifica con la password corretta ed errata, asserendo il risultato true per la corretta corrispondenza e false altrimenti.

testAggiungiContenuto():

- Scopo: Verificare che l'aggiunta di un contenuto alla raccolta dello studente avvenga correttamente.
- Realizzazione: Creando un contenuto e aggiungendolo alla mappa, si controlla l'incremento del conteggio e la corretta registrazione.

testRimuoviContenuto():

- Scopo: Verificare la corretta rimozione di un contenuto dalla raccolta dello studente.
- Realizzazione: Si aggiunge un contenuto, lo si rimuove e si asserisce che la mappa venga aggiornata, non contenendo più il contenuto.

testSetNome():

- Scopo: Verificare l'aggiornamento del nome dello studente e la propagazione del cambiamento ai dati di pagamento associati.
- Realizzazione: Si modifica il nome dello studente e si verifica che il nuovo nome sia aggiornato sia nel profilo che in ogni dato di pagamento registrato.

testSetCognome():

- Scopo: Assicurare che la modifica del cognome dello studente venga propagata correttamente.
- Realizzazione: Dopo aver impostato un nuovo cognome, si controlla che tale modifica sia riflessa nel profilo e nei relativi dati di pagamento.

## TestGruppoStudio

testVerificaIscrizione():

- Scopo: Verificare che uno studente venga correttamente iscritto a un gruppo di studio.

- Realizzazione: Creando un gruppo e aggiungendo uno studente, si asserisce che lo studente sia iscritto (con incremento del conteggio) e riconosciuto dal gruppo.

testPasswordCorretta():

- Scopo: Verificare la correttezza del controllo della password di un gruppo di studio.
- Realizzazione: Si testa il metodo del gruppo che controlla la password, asserendo il risultato true se la password inserita corrisponde a quella impostata.

testAggiungiStudente():

- Scopo: Verificare l'aggiunta di uno studente a un gruppo di studio.
- Realizzazione: Si aggiunge uno studente al gruppo e si verifica che il gruppo riconosca correttamente l'iscrizione e che il numero di studenti aumenti.

testRimuoviStudente():

- Scopo: Verificare la rimozione di uno studente da un gruppo di studio.
- Realizzazione: Dopo aver aggiunto uno studente, si procede alla sua rimozione e si controlla che il gruppo non riconosca più lo studente e che il conteggio diminuisca.

## TestAppunto

testModificaAppunto():

- Scopo: Verificare che le modifiche apportate a un appunto (titolo e file) vengano applicate correttamente.
- Realizzazione: Si modifica un appunto esistente e si asserisce che il titolo e il file siano aggiornati secondo i nuovi valori.

## Test di sistema

Alla conclusione della fase di integrazione e dopo l'esecuzione di un ampio insieme di test unitari (tramite JUnit) per verificare ogni componente del sistema StudyHub, è stata condotta una fase di test manuale finalizzata a convalidare l'esperienza utente e l'integrazione complessiva delle funzionalità principali. Questa fase ha rappresentato un ulteriore livello di validazione, volto a garantire che l'applicazione non solo fosse corretta dal punto di vista del codice, ma anche intuitiva, coerente e performante per l'utente finale. Di seguito, la sintesi dei principali aspetti testati manualmente, adattata al contesto dei test precedentemente eseguiti:

Verifica dell'Interfaccia Utente:

- Obiettivo: Assicurare che l'interfaccia offra un'esperienza intuitiva e coerente, rispondendo alle aspettative degli studenti, sia riguardo la creazione di corsi e gruppi studio che riguardo le iscrizioni e i download di materiale.
- Attività: Navigazione attraverso le diverse schermate e menu, verificando la chiarezza dell'organizzazione e la facilità di utilizzo. In particolare, sono stati esaminati gli scenari relativi alla gestione dei profili (creazione, modifica ed eliminazione profilo), la creazione e gestione dei corsi e dei gruppi di studio, le iscrizioni ad entrambi e il download ed eliminazione di contenuti e appunti.

Test delle Funzionalità Principali:

- Obiettivo: Verificare che tutte le operazioni essenziali – quali la creazione e la modifica di profili, corsi, iscrizioni, gruppi di studio, contenuti e appunti – siano implementate correttamente e rispondano ai requisiti specificati.
- Attività: Esecuzione di scenari di utilizzo realistici che simulano le interazioni tipiche degli utenti con il proprio profilo, con i materiali e i canali (corsi e gruppi studio).

#### **Analisi della Coerenza dei Dati:**

- **Obiettivo:** Verificare che i dati visualizzati nelle interfacce utente siano coerenti con quelli memorizzati nel sistema.
- **Attività:** Confronto dei dati registrati a seguito delle operazioni eseguite (come la creazione, modifica ed eliminazione di profili, corsi e gruppi) con i dati effettivamente presenti nel database, assicurando così l'integrità e la consistenza del sistema.

#### **Valutazione delle Prestazioni:**

- **Obiettivo:** Garantire che l'applicazione mantenga elevate prestazioni e reattività, anche in condizioni di utilizzo intensivo.
- **Attività:** Esecuzione di test di carico simulato e monitoraggio delle risposte del sistema durante interazioni ripetute, verificando che le operazioni – già validate a livello unitario – continuino a essere performanti in uno scenario integrato.

In sintesi, la fase di test manuale ha confermato l'esito positivo dei test unitari, dimostrando che il sistema StudyHub è conforme alle specifiche progettuali. L'interfaccia utente si è rivelata intuitiva e ben strutturata, i dati sono risultati coerenti e le prestazioni dell'applicazione soddisfacenti, rendendola pronta per il rilascio e l'adozione da parte degli utenti.