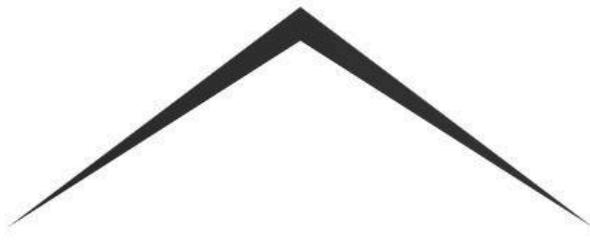


# **Gym4U**



# **GYM 4U**



Studenti:

Alessandro Strano 1000015308  
Francesco Romeo 1000067660

## **Prefazione**

Il documento qui presente fornisce una panoramica dettagliata dello sviluppo dell'applicazione Gym4U, realizzata nel contesto del corso di Ingegneria del Software. L'implementazione del software è avvenuta utilizzando il linguaggio Java all'interno dell'ambiente di sviluppo Visual Studio Code. La progettazione è stata condotta attraverso l'utilizzo del software Astah Professional, con l'obiettivo di presentare esclusivamente le versioni finali di ciascuna componente, elaborate al termine di tutte le fasi progettuali. Eventuali versioni intermedie sono consultabili nei documenti contenuti nelle corrispondenti sottocartelle della documentazione.

Il documento approfondisce inoltre le fasi conclusive del processo, focalizzandosi sul testing e sul refactoring del programma. Queste fasi hanno comportato miglioramenti significativi al codice, contribuendo a ottimizzare la qualità complessiva dell'applicazione.

# Indice

<b>Ideazione e analisi dei requisiti</b>	<b>6</b>
Introduzione	6
Obiettivi e requisiti	6
Obiettivo dei casi d'uso	7
Casi d'uso	9
1. Iscrizione ai corsi	9
2. Prenotazione lezione corso	11
3. Creazione nuovo corso	13
4. Registrazione nuovo cliente	15
5. Gestione abbonamento	17
6. Prenotazione Personal Trainer	19
7. Creazione di un'offerta promozionale	21
8. Creazione di una scheda personalizzata	23
9. Visualizza prenotati di una lezione	25
10. Ingresso in Palestra tramite badge	27
11. Visualizza Scheda Personalizzata	29
12. Visualizza Corsi Palestra	29
13. Modifica Account e Password	29
14. Modifica Cliente	29
15. Aggiungi nuovo Personal Trainer	29
Documento di Visione	30
Regole di business	30
Specifiche supplementari	32
Glossario	33
<b>Analisi orientata agli oggetti</b>	<b>34</b>
Introduzione	34
→ Iterazione 1:	34
→ Iterazione 2:	34
→ Iterazione 3:	34
→ Iterazione 4:	35
Modello di dominio	35
Modello degli oggetti	38
→ Iterazione 1:	38
→ Iterazione 2:	38
→ Iterazione 3:	39
→ Iterazione 4:	39
SSD e Contratti	40
→ Iterazione 1:	40
SSD Caso d'Uso 1	40
SSD Caso d'Uso 2	41
SSD Caso d'Uso 3	41
Contratti Operazioni UC1	42
Contratti Operazioni UC2	42

Contratti Operazioni UC3	43
→ Iterazione 2:	44
SSD Caso d'Uso 4	44
SSD Caso d'Uso 5	45
SSD Caso d'Uso 6	45
SSD Caso d'Uso 10	46
Contratti Operazioni UC4	46
Contratti Operazioni UC5	48
Contratti Operazioni UC6	49
Contratti Operazioni UC10	49
→ Iterazione 3:	50
SSD Caso d'Uso UC7	50
SSD Caso d'Uso UC8	50
SSD Caso d'Uso UC9	51
Contratti Operazioni UC7	51
Contratti Operazioni UC8	52
Contratti Operazioni UC9	53
→ Iterazione 4:	54
SSD Caso d'Uso UC3	54
SSD Caso d'Uso UC4	54
SSD Caso d'Uso UC5	55
SSD Caso d'Uso UC2	56
Contratti Operazioni UC5	56
Contratti Operazioni UC6	57
Contratti Operazioni UC8	57
→ Documentazione Completa:	58
SSD Caso d'Uso UC9	58
<b>Progettazione</b>	<b>59</b>
Diagramma delle classi	59
Diagramma di sequenza	61
→ Iterazione 1:	61
UC1	61
UC2	62
UC3	63
Caso d'Uso di Avviamento	63
→ Iterazione 2:	64
UC2	64
UC4	64
UC5	66
UC6	67
UC10	68
→ Iterazione 3:	69
UC7	69
UC8	69

UC9	70
→ Iterazione 4:	71
UC2	71
UC3	71
UC4 e UC5	71
<b>Testing</b>	<b>72</b>
Introduzione	72
Individuazione dei casi di test e testing unitario	72
→ Iterazione 1:	72
→ Iterazione 2:	76
→ Iterazione 3:	78
Test di sistema	80
<b>Refactoring e conclusioni</b>	<b>81</b>
Refactoring	81
Test di accettazione	81

# Ideazione e analisi dei requisiti

## Introduzione

**Scopo e Contesto:** Gym4U è pensato per essere un software innovativo dedicato alla gestione di palestre. Il suo scopo principale è automatizzare e ottimizzare le operazioni amministrative, migliorando al contempo l'esperienza degli utenti. Intende fornire una soluzione completa che faciliti la gestione delle iscrizioni, delle prenotazioni dei corsi, della comunicazione con gli utenti e della gestione finanziaria.

**Visione:** Gym4U si propone di essere un sistema integrato, user-friendly e multifunzionale, che soddisfi le esigenze sia dei gestori di palestre, dei personal trainer e dei loro clienti, rendendo la gestione delle palestre più efficiente e l'esperienza utente più soddisfacente.

## Obiettivi e requisiti

**Gestione Utenti:** Implementazione di un sistema per la registrazione, il monitoraggio e la gestione dei profili degli iscritti, con storico delle attività e gestione delle preferenze.

**Prenotazione Corsi:** Un sistema flessibile per prenotare, modificare e annullare la partecipazione ai corsi, con notifiche in tempo reale.

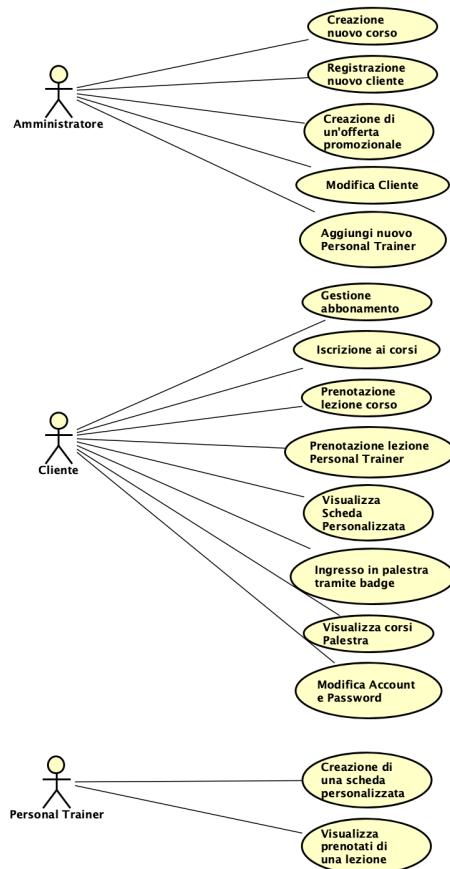
**Personal Trainer:** Funzionalità per la gestione degli orari di lavoro, delle competenze e delle prestazioni dei personal trainer.

**Integrazione con Servizi Esterni:** Connessione con sistemi di messaggistica email.

## Obiettivo dei casi d'uso

ATTORE PRINCIPALE	OBIETTIVO	CASO D'USO
Cliente	Consentire agli utenti autenticati di iscriversi ai corsi disponibili, con il sistema che registra l'iscrizione, aggiorna la disponibilità dei corsi.	<b>UC1: Iscrizione ai corsi</b>
Cliente	Fornire agli utenti iscritti la possibilità di prenotare le lezioni dei corsi desiderati attraverso Gym4U, con il sistema che registra la prenotazione e conferma all'utente via email.	<b>UC2: Prenotazione lezione corso</b>
Amministratore	Creare un nuovo corso nel sistema per ampliare l'offerta formativa della palestra, garantendo una gestione aggiornata e dettagliata dei corsi disponibili.	<b>UC3: Creazione nuovo corso</b>
Amministratore	Registrare un nuovo cliente nel sistema, mantenendo aggiornato l'elenco degli iscritti con informazioni complete.	<b>UC4: Registrazione nuovo cliente</b>
Cliente	Consentire agli utenti di gestire il proprio abbonamento, modificando le opzioni e il metodo di pagamento.	<b>UC5: Gestione abbonamento</b>
Cliente	Consentire agli utenti di prenotare sessioni individuali con un Personal Trainer attraverso Gym4U.	<b>UC6: Prenotazione Personal Trainer</b>
Amministratore	Consentire all'amministratore di creare offerte promozionali su Gym4U, specificando sconti, corsi inclusi e periodi di validità, per incentivare l'adesione e la partecipazione ai corsi.	<b>UC7: Creazione di un'offerta promozionale</b>
Personal Trainer	Consentire al Personal Trainer di creare una scheda personalizzata ed associarla ad un utente.	<b>UC8: Creazione di una scheda personalizzata</b>
Personal Trainer	Consentire al Personal Trainer di visualizzare gli iscritti prenotati per una lezione selezionata.	<b>UC9: Visualizza prenotati di una lezione</b>
Cliente	Consentire al Cliente l'accesso alla palestra passando il badge, consegnatogli durante l'iscrizione, e	<b>UC10: Ingresso in palestra tramite badge</b>

	confermare così la sua effettiva presenza.	
Cliente	Consentire al Cliente di visualizzare, se presente, la scheda personalizzata assegnatagli da un Personal Trainer	<b>UC11: Visualizza Scheda Personalizzata</b>
Cliente	Consentire al Cliente di visualizzare l'elenco dei corsi offerti dalla palestra su Gym4U, inclusi giorni, orari e dettagli.	<b>UC12: Visualizza Corsi Palestra</b>
Cliente	Consentire al Cliente di modificare i dati anagrafici, di contatto e la propria Password.	<b>UC13: Modifica Account e Password</b>
Amministratore	Consentire all'amministratore di modificare i dati anagrafici e di contatto associati a un cliente.	<b>UC14: Modifica Cliente</b>
Amministratore	Consentire all'amministratore di aggiungere un nuovo Personal Trainer al sistema Gym4U.	<b>UC15: Aggiungi nuovo Personal Trainer</b>



# Casi d'uso

## 1. Iscrizione ai corsi

<b>Nome</b>	UC1: Iscrizione ai corsi
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Cliente
<b>Parte interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera che gli utenti si possano iscrivere ai corsi</li> <li>• Palestra: Desidera mantenere un registro accurato delle iscrizioni ai corsi e aggiornare la disponibilità.</li> <li>• Personal Trainer: Vogliono ricevere notifiche sulle iscrizioni ai corsi.</li> <li>• Cliente: Desidera iscriversi ai corsi desiderati attraverso Gym4U.</li> </ul>
<b>Pre-condizioni</b>	Il Cliente è autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	<p>Il Cliente è iscritto correttamente al corso selezionato.      Il sistema registra l'iscrizione e aggiorna le informazioni sulla disponibilità dei corsi.      Personal Trainer notificati sull'iscrizione dell'utente al corso.</p>
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente accede a Gym4U e accede alla sezione "Iscrizione ai Corsi".</li> <li>2. Il Sistema verifica che il Cliente abbia un abbonamento attivo.</li> <li>3. Il Sistema verifica che il Cliente abbia un certificato medico valido.</li> <li>4. Il Cliente seleziona il corso desiderato.</li> <li>5. Il Sistema ritorna le informazioni del corso selezionato.</li> <li>6. Il Sistema registra l'iscrizione del Cliente al corso selezionato.</li> <li>7. Il Sistema aggiorna le informazioni sulla disponibilità del corso.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Cliente riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.           <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:               <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Cliente, registra l'errore, e passa in uno stato pulito.</li> <li>ii. Il Cliente prova nuovamente ad iscriversi ad un corso.</li> </ol> </li> </ol> </li> </ol> <p><b>2.a:</b> Il Cliente non ha un abbonamento attivo:</p> <ol style="list-style-type: none"> <li>1. Il Sistema invita il Cliente a sottoscrivere un abbonamento</li> </ol>

	<p><b>3.a:</b> Il Cliente non ha un certificato medico valido:</p> <ol style="list-style-type: none"> <li>1. Il Sistema invita il Cliente a presentare all'Amministrazione un nuovo certificato medico</li> </ol> <p><b>4.a:</b> Il Cliente tenta di iscriversi a un corso già pieno o non disponibile:</p> <ol style="list-style-type: none"> <li>1. Il sistema avvisa l'utente che il corso selezionato è già pieno o non disponibile.</li> <li>2. L'utente può scegliere un altro corso disponibile o tornare alla lista dei corsi.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per la selezione e l'iscrizione ai corsi. Aggiornamento istantaneo della disponibilità dei corsi.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Numerose volte al giorno
<b>Varie</b>	

## 2. Prenotazione lezione corso

<b>Nome</b>	UC2: Prenotazioni Lezione Corso
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Cliente
<b>Parte interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Personal Trainer: Desiderano avere informazioni aggiornate sulle prenotazioni degli orari corsi per garantire una corretta preparazione delle lezioni e la gestione degli iscritti.</li> <li>• Palestra: Desidera registrare e organizzare le prenotazioni degli iscritti per i corsi offerti.</li> <li>• Cliente: Desidera prenotare gli orari dei corsi desiderati attraverso Gym4U per partecipare alle lezioni.</li> </ul>
<b>Pre-condizioni</b>	Il Cliente è autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	<p>La prenotazione dell'orario corso desiderato è registrata nel sistema.</p> <p>Il Cliente riceve una conferma via email della prenotazione.</p>
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente accede alla sezione "Prenotazioni Corsi" su Gym4U.</li> <li>2. Il Sistema verifica che il Cliente abbia un abbonamento attivo.</li> <li>3. Il Sistema verifica che il Cliente abbia un certificato medico valido.</li> <li>4. Il Cliente visualizza l'elenco dei corsi in cui è iscritto.</li> <li>5. Il Cliente sceglie il corso per cui vuole prenotare una lezione.</li> <li>6. Il Sistema ritorna l'elenco delle lezioni del corso scelto.</li> <li>7. Il Cliente seleziona la lezione a cui vuole partecipare.</li> <li>8. Il Sistema ritorna il riepilogo.</li> <li>9. Il Cliente conferma la prenotazione.</li> <li>10. Il Sistema registra la prenotazione alla lezione.</li> <li>11. Il Sistema invia una conferma della prenotazione all'utente via email.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Cliente riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.           <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:               <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Cliente, registra l'errore, e passa in uno stato pulito.</li> <li>ii. Il Cliente inizia una nuova prenotazione dell'orario per un corso.</li> </ol> </li> </ol> </li> </ol> <p><b>2.a:</b> Il Cliente non ha un abbonamento attivo:</p> <ol style="list-style-type: none"> <li>1. Il Sistema invita il Cliente a sottoscrivere un abbonamento</li> </ol>

	<p><b>3.a:</b> Il Cliente non ha un certificato medico valido:</p> <ol style="list-style-type: none"> <li>1. Il Sistema invita il Cliente a presentare all'Amministrazione un nuovo certificato medico</li> </ol> <p><b>7.a:</b> Il Cliente tenta di prenotarsi per una lezione che è nello stesso giorno di un'altra sua prenotazione:</p> <ol style="list-style-type: none"> <li>1. Il Sistema rileva che il Cliente ha già una prenotazione per un'altra lezione in quel giorno.</li> <li>2. Il Sistema segnala all'utente che, per regola di business, è consentita solo una prenotazione di lezione al giorno.</li> <li>3. Il Sistema invita il Cliente ad iscrversi ad una lezione di un altro giorno o eliminare la prenotazione precedente</li> </ol> <p><b>11.a:</b> Il sistema non riesce a inviare la conferma via email:</p> <ol style="list-style-type: none"> <li>1. Il sistema segnala un errore nell'invio della conferma.</li> <li>2. Il sistema invita il Cliente a verificare la casella di posta spam o fornisce un messaggio per contattare il supporto tecnico in caso di problemi.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia utente intuitiva per la prenotazione degli orari corsi. Sistema di notifica email funzionante per la conferma delle prenotazioni.
<b>Elenco delle varianti tecnologiche e dei dati</b>	
<b>Frequenza di ripetizioni</b>	Molto frequente, a seconda della partecipazione agli orari dei corsi da parte degli iscritti.
<b>Varie</b>	

### 3. Creazione nuovo corso

<b>Nome</b>	UC3: Creazione nuovo corso
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Amministratore del Sistema
<b>Parte interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera creare un nuovo corso nel sistema.</li> <li>• Palestra: Desidera mantenere un elenco aggiornato dei corsi disponibili con relative informazioni.</li> <li>• Personal Trainer: Desiderano essere assegnati a nuovi corsi in modo da poter pianificare le sessioni di allenamento.</li> <li>• Cliente: Desidera avere ampia scelta tra i corsi a disposizione</li> </ul>
<b>Pre-condizioni</b>	L'Amministratore è identificato e autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	Il nuovo corso è creato nel sistema. Le informazioni sul corso, inclusi istruttore e disponibilità, sono registrate. L'inventario della palestra è aggiornato.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'Amministratore seleziona l'opzione "Creazione Nuovo Corso".</li> <li>2. L'Amministratore inserisce i dettagli del nuovo corso, come nome del corso, posti disponibili, orari, giorni, durata lezioni e Personal Trainer assegnati.</li> <li>3. Il Sistema ritorna le informazioni inserite.</li> <li>4. L'Amministratore conferma la registrazione del nuovo corso.</li> <li>5. Il Sistema associa i Personal Trainer al corso.</li> <li>6. Il Sistema crea le lezioni associate al corso sulla base delle informazioni inserite.</li> <li>7. Il Sistema registra il corso e le relative lezioni.</li> <li>8. Il Sistema ritorna tutte le informazioni del corso appena aggiunto.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. L'Amministratore del Sistema riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore all'Amministratore del Sistema, registra l'errore, e passa in uno stato pulito.</li> <li>ii. L'Amministratore del Sistema inizia una nuova registrazione del cliente.</li> </ol> </li> </ol> </li> </ol> <p><b>2.a:</b> L'Amministratore inserisce i dettagli del nuovo corso, ma il sistema segnala un errore di validazione (l'istruttore ha raggiunto il limite massimo di corsi).</p>

	<ol style="list-style-type: none"> <li>1. Il Sistema notifica all'Amministratore l'errore e chiede di correggere i dati.</li> <li>2. L'Amministratore corregge i dettagli del nuovo corso secondo le indicazioni del sistema.             <ol style="list-style-type: none"> <li>a. Il Sistema verifica nuovamente i dati inseriti e, se corretti, procede con la creazione del nuovo corso.</li> <li>b. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.</li> </ol> </li> </ol> <p><b>5.a:</b> Il Sistema fallisce durante l'associazione corso-istruttore:</p> <ol style="list-style-type: none"> <li>1. Il Sistema segnala l'errore.</li> <li>2. Il Sistema invita l'amministratore a riprovare</li> </ol>
<b>Requisiti speciali</b>	Interfaccia grafica intuitiva e di facile utilizzo. Il sistema deve essere in grado di gestire l'associazione dei Personal Trainer ai corsi. Il tempo di risposta del sistema deve essere entro pochi secondi.
<b>Elenco delle varianti tecnologiche e dei dati</b>	Il sistema deve essere in grado di gestire diverse tipologie di corsi e orari. Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande.
<b>Frequenza di ripetizioni</b>	Variabile, in base alle necessità della palestra per l'aggiunta di nuovi corsi.
<b>Varie</b>	

#### 4. Registrazione nuovo cliente

<b>Nome</b>	UC4: Registrazione nuovo cliente
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Amministratore del Sistema
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera registrare un nuovo cliente nel sistema.</li> <li>• Palestra: Desidera mantenere un elenco aggiornato degli iscritti con informazioni complete.</li> <li>• Cliente (Nuovo Iscritto): Desidera essere registrato correttamente nel sistema per accedere alle funzionalità della palestra.</li> </ul>
<b>Pre-condizioni</b>	L'Amministratore è identificato e autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	Il nuovo iscritto è registrato nel sistema. Vengono registrati i dettagli come nome, cognome, contatti e altri dettagli pertinenti. L'inventario della palestra è aggiornato.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'Amministratore seleziona l'opzione "Registrazione Nuovo Cliente".</li> <li>2. Inserisce i dettagli del nuovo cliente, come nome, cognome, data di nascita, indirizzo, e contatti.</li> <li>3. Il Sistema ritorna le informazioni inserite.</li> <li>4. L'amministratore verifica che il certificato medico fornito sia valido e lo registra.</li> <li>5. L'Amministratore associa l'iscritto all'abbonamento richiesto.</li> <li>6. L'Amministratore associa il Metodo di Pagamento con i dati comunicati dal Cliente.</li> <li>7. Il Sistema ritorna tutte le informazioni inserite.</li> <li>8. L'Amministratore conferma la registrazione del nuovo Cliente.</li> <li>9. Il Sistema registra le informazioni del nuovo Cliente.</li> <li>10. L'amministratore rilascia il badge univoco al Cliente.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. L'Amministratore del Sistema riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore all'Amministratore del Sistema, registra l'errore, e passa in uno stato pulito.</li> <li>ii. L'Amministratore del Sistema inizia una</li> </ol> </li> </ol> </li> </ol>

	<p>nuova registrazione del cliente.</p> <p><b>2.a:</b> L'Amministratore inserisce i dettagli del nuovo cliente, ma il sistema segnala un errore di validazione:</p> <ul style="list-style-type: none"> <li>a. Formato del numero di telefono o email non valido:           <ul style="list-style-type: none"> <li>1. Il sistema notifica all'Amministratore l'errore e chiede di correggere i dati.</li> <li>2. L'Amministratore corregge i dettagli del nuovo cliente secondo le indicazioni del sistema.</li> <li>3. Il sistema verifica nuovamente i dati inseriti e, se corretti, procede con la registrazione del cliente.</li> <li>4. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.</li> </ul> </li> <li>b. Data di nascita inserita in contrasto con la Regola di Business R7:           <ul style="list-style-type: none"> <li>1. Il sistema notifica all'Amministratore che il cliente non può essere registrato in quanto in contrasto con la Regola di Business R7</li> </ul> </li> </ul> <p><b>4.a:</b> Il Certificato Medico è scaduto:</p> <ul style="list-style-type: none"> <li>1. Il Sistema comunica che il Certificato Medico non è valido.</li> <li>2. Il Sistema interrompe la registrazione del Cliente.</li> </ul> <p><b>6.a:</b> Il Metodo di Pagamento è scaduto:</p> <ul style="list-style-type: none"> <li>1. Il Sistema comunica che il Metodo di Pagamento non è valido.</li> <li>2. Invita l'Amministratore a far comunicare al Cliente un Metodo di Pagamento valido.</li> </ul>
<b>Requisiti speciali</b>	<p>Interfaccia grafica intuitiva e di facile utilizzo.</p> <p>Il sistema deve gestire l'associazione degli iscritti agli abbonamenti in modo accurato.</p> <p>Il tempo di risposta del sistema deve essere entro pochi secondi.</p> <p>Lettore NFC compatibile con i badge.</p>
<b>Elenco delle varianti tecnologiche e dei dati</b>	<p>Il sistema deve essere in grado di gestire diverse tipologie di abbonamenti e dettagli degli iscritti.</p> <p>Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande.</p>
<b>Frequenza di ripetizioni</b>	Decine di volte al giorno, soprattutto nei periodi di nuove iscrizioni o promozioni.
<b>Varie</b>	

## 5. Gestione abbonamento

<b>Nome</b>	UC5: Gestione Abbonamento
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Cliente
<b>Parte interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera che il sistema di pagamento degli abbonamenti sia automatizzato</li> <li>• Palestra: Desidera mantenere aggiornate le informazioni sugli abbonamenti e i metodi di pagamento dei suoi iscritti.</li> <li>• Cliente: Desidera gestire il proprio abbonamento e il metodo di pagamento attraverso Gym4U.</li> </ul>
<b>Pre-condizioni</b>	L'Utente è autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	L'Utente modifica correttamente l'abbonamento e/o il metodo di pagamento. Le modifiche sono riflesse immediatamente nel sistema e nell'inventario della palestra.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'Utente accede alla sezione "Gestione Abbonamento" in Gym4U.</li> <li>2. L'Utente visualizza le attuali impostazioni di abbonamento e di pagamento.</li> <li>3. L'Utente seleziona quale impostazione vuole modificare.</li> <li>4. L'Utente modifica l'abbonamento o il metodo di pagamento.</li> <li>5. Il sistema ritorna il riepilogo delle impostazioni di abbonamento o del metodo di pagamento aggiornate.</li> <li>6. L'Utente conferma le modifiche.</li> <li>7. Il sistema salva le modifiche.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Cliente riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Cliente, registra l'errore, e passa in uno stato pulito.</li> <li>ii. Il Cliente prova nuovamente la procedura di gestione dell'abbonamento.</li> </ol> </li> </ol> </li> </ol> <p><b>4.a:</b> Il sistema rileva un problema nell'aggiornamento istantaneo delle modifiche nell'inventario:</p> <ol style="list-style-type: none"> <li>1. Il sistema tenta nuovamente l'aggiornamento:             <ol style="list-style-type: none"> <li>a. Il sistema fa un altro tentativo di aggiornare le modifiche nell'inventario.</li> <li>b. Se l'aggiornamento ha successo, il caso d'uso prosegue normalmente.</li> </ol> </li> <li>2. Il sistema rileva un problema persistente nell'aggiornamento:             <ol style="list-style-type: none"> <li>a. Il sistema registra l'errore nell'aggiornamento delle</li> </ol> </li> </ol>

	<p>modifiche.</p> <p>b. L'utente viene invitato a contattare l'assistenza.</p>
<b>Requisiti speciali</b>	Interfaccia utente intuitiva per la modifica dell'abbonamento e del metodo di pagamento. Aggiornamento istantaneo delle modifiche nell'inventario della palestra.
<b>Elenco delle varianti tecnologiche e dei dati</b>	Gestione dei diversi tipi di abbonamenti e dei relativi metodi di pagamento. Integrazione immediata e accurata delle modifiche nei dati dell'abbonamento.
<b>Frequenza di ripetizioni</b>	Relativo, con periodi di modifiche degli abbonamenti durante offerte speciali o cambiamenti nei requisiti dell'utente.
<b>Varie</b>	

## 6. Prenotazione Personal Trainer

<b>Nome</b>	UC6: Prenotazione Personal Trainer
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Cliente
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera gestire e monitorare le prenotazioni dei Personal Trainer.</li> <li>• Personal Trainer: Desidera ricevere notifiche sulle prenotazioni dei clienti.</li> <li>• Cliente: Desidera prenotare una sessione con un Personal Trainer.</li> </ul>
<b>Pre-condizioni</b>	L'Utente è identificato e autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	La prenotazione della sessione con il Personal Trainer è registrata nel sistema. L'Utente riceve la notifica.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'utente accede alla sezione "Prenotazioni Personal Trainer" su Gym4U.</li> <li>2. Il Sistema verifica che il Cliente abbia un abbonamento attivo.</li> <li>3. Il Sistema verifica che il Cliente abbia un certificato medico valido.</li> <li>4. Il Cliente visualizza l'elenco dei Personal Trainer.</li> <li>5. L'Utente seleziona il Personal Trainer desiderato e sceglie una data ed un orario per la sessione.</li> <li>6. Il Sistema verifica la disponibilità del Personal Trainer e ritorna le informazioni inserite.</li> <li>7. Il Cliente conferma la prenotazione.</li> <li>8. Il Sistema invia una notifica all'Utente.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. L'Utente riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore all'Utente, registra l'errore e passa in uno stato pulito.</li> <li>ii. L'Utente inizia una nuova prenotazione con il Personal Trainer.</li> </ol> </li> </ol> </li> </ol> <p><b>6.a:</b> Il Personal Trainer non è disponibile nella data/orario selezionati:</p> <ol style="list-style-type: none"> <li>1. Il Sistema notifica al Cliente e suggerisce alternative di data/orario.</li> </ol>

	<p>2. L'Utente seleziona una nuova data/orario o sceglie un altro Personal Trainer disponibile.</p> <p><b>6.b:</b> Il sistema segnala un errore durante la conferma della prenotazione:</p> <ol style="list-style-type: none"> <li>1. Il Sistema notifica all'Utente l'errore e chiede di riprovare.</li> <li>2. L'Utente corregge gli errori secondo le indicazioni del sistema.</li> <li>3. Il Sistema verifica nuovamente i dati inseriti e, se corretti, procede con la prenotazione.</li> <li>4. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.</li> </ol>
<b>Requisiti speciali</b>	<p>Interfaccia grafica intuitiva per la prenotazione.</p> <p>Sistema di notifiche efficiente per informare l'Utente e il Personal Trainer.</p> <p>Gestione accurata della disponibilità dei Personal Trainer.</p> <p>Tempo di risposta del sistema entro pochi secondi.</p>
<b>Elenco delle varianti tecnologiche e dei dati</b>	<p>Il sistema deve gestire diversi Personal Trainer e le relative disponibilità.</p> <p>Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande.</p>
<b>Frequenza di ripetizioni</b>	Abbastanza frequente, varia in base alle richieste degli Utenti e alla disponibilità dei Personal Trainer.
<b>Varie</b>	

## 7. Creazione di un'offerta promozionale

<b>Nome</b>	UC7: Creazione di un'offerta promozionale
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Amministratore del Sistema
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Amministratore del Sistema: Desidera creare offerte promozionali per incentivare l'adesione e la partecipazione ai corsi.</li> <li>• Cliente: Desidera beneficiare degli sconti e delle promozioni nelle offerte.</li> </ul>
<b>Pre-condizioni</b>	L'Amministratore è identificato e autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	L'Offerta Promozionale è creata nel sistema con successo e si applica automaticamente nel periodo stabilito.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. L'Amministratore accede all'opzione "Creazione Offerta Promozionale".</li> <li>2. L'Amministratore specifica i dettagli dell'offerta, inclusi sconti percentuali e il periodo di validità.</li> <li>3. Il sistema registra e salva l'Offerta Promozionale.</li> <li>4. Durante il periodo di validità, il sistema applica automaticamente gli sconti e le promozioni agli Utenti iscritti.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. L'Amministratore riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore all'Amministratore, registra l'errore e passa in uno stato pulito.</li> <li>ii. L'Amministratore inizia una nuova creazione dell'Offerta Promozionale.</li> </ol> </li> </ol> </li> <p><b>2.a:</b> L'Amministratore non specifica correttamente i dettagli dell'offerta:</p> <ol style="list-style-type: none"> <li>1. Il sistema notifica all'Amministratore e chiede di correggere o fornire informazioni mancanti.</li> <li>2. L'Amministratore apporta le correzioni richieste.</li> <li>3. Il sistema verifica nuovamente i dati inseriti e, se corretti, procede con la creazione dell'offerta.</li> <li>4. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.</li> </ol> </ol>

	<p><b>4.a:</b> Durante il periodo di validità, l'Amministratore inserisce un'offerta promozionale in un periodo già esistente:</p> <ol style="list-style-type: none"> <li>1. Il sistema rileva l'errore e notifica all'Amministratore che l'offerta configge con un periodo già programmato.</li> <li>2. Il sistema fornisce all'Amministratore la possibilità di selezionare un periodo diverso o di modificare l'offerta esistente.</li> <li>3. L'Amministratore apporta le correzioni richieste.</li> <li>4. Il sistema verifica nuovamente i dati inseriti e, se corretti, procede con la creazione dell'offerta.</li> <li>5. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.</li> </ol>
<b>Requisiti speciali</b>	<p>Interfaccia grafica intuitiva per la creazione delle offerte promozionali.</p> <p>Sistema automatico di applicazione delle offerte durante il periodo di validità.</p> <p>Registro dettagliato delle Offerte Promozionali e delle applicazioni agli Utenti.</p>
<b>Elenco delle varianti tecnologiche e dei dati</b>	<p>Il sistema deve gestire diverse tipologie di sconti e promozioni.</p> <p>Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande.</p> <p>Integrazione con il sistema di gestione dei corsi per associare automaticamente i corsi inclusi nell'offerta.</p>
<b>Frequenza di ripetizioni</b>	Generalmente a intervalli trimestrali o semestrali.
<b>Varie</b>	

## 8. Creazione di una scheda personalizzata

<b>Nome</b>	UC8: Creazione di una scheda personalizzata
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Personal Trainer
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Personal Trainer: Desidera creare schede personalizzate per gli utenti al fine di adattare gli allenamenti alle loro esigenze e monitorarne i progressi.</li> <li>• Cliente: Beneficia della personalizzazione degli allenamenti, adatti alle proprie esigenze e obiettivi di fitness.</li> </ul>
<b>Pre-condizioni</b>	Il Personal Trainer è autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	Il Personal Trainer crea con successo una scheda personalizzata e la associa all'utente desiderato.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Il Personal Trainer accede all'opzione "Creazione di una Scheda Personalizzata" nel sistema Gym4U.</li> <li>2. Il Personal Trainer inserisce gli esercizi e i dettagli specifici della scheda personalizzata.</li> <li>3. Il Sistema mostra gli utenti disponibili al Personal Trainer.</li> <li>4. Il Personal Trainer seleziona l'utente a cui associare la scheda.</li> <li>5. Il Sistema ritorna le informazioni inserite.</li> <li>6. Il Personal Trainer conferma l'inserimento.</li> <li>7. Il Sistema registra le informazioni inserite.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Personal Trainer riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.             <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:                     <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Personal Trainer, registra l'errore e passa in uno stato pulito.</li> <li>ii. Il Personal Trainer inizia una creazione di una scheda personalizzata.</li> </ol> </li> </ol> </li> </ol> <p><b>4.a:</b> Il Personal Trainer riscontra un errore nell'associazione della scheda personalizzata all'utente:</p> <ol style="list-style-type: none"> <li>1. Il sistema segnala un errore durante il processo di selezione dell'utente per l'associazione.</li> <li>2. Il Personal Trainer riceve un avviso e può correggere l'errore di associazione selezionando nuovamente l'utente.</li> </ol>

	<p><b>6.a:</b> Il Personal Trainer non conferma l'inserimento:</p> <ol style="list-style-type: none"> <li>1. Il Personal Trainer annulla la scheda inserita.</li> <li>2. Il Sistema ritorna al menù iniziale.</li> </ol> <p><b>7.a:</b> Il Sistema rileva un errore di registrazione dei dati:</p> <ol style="list-style-type: none"> <li>1. Il Sistema segnala un errore durante il processo di registrazione delle informazioni inserite.</li> <li>2. Il Personal Trainer riceve una notifica sull'errore di registrazione.</li> <li>3. Il Personal Trainer può rientrare la registrazione o chiedere assistenza al supporto tecnico.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia intuitiva per la creazione della scheda personalizzata. Capacità di associazione della scheda all'utente desiderato. Possibilità di aggiunta di dettagli specifici sull'utente o sulla scheda in fase di creazione.
<b>Elenco delle varianti tecnologiche e dei dati</b>	Memorizzazione delle schede personalizzate associate agli utenti nel database del sistema. Interfaccia user-friendly per l'inserimento e la modifica delle schede personalizzate. Connessione stabile per garantire il salvataggio corretto delle informazioni.
<b>Frequenza di ripetizioni</b>	Frequente, in base alle necessità di creazione di nuove schede personalizzate per gli utenti.
<b>Varie</b>	

## 9. Visualizza prenotati di una lezione

<b>Nome</b>	UC9: Visualizza prenotati di una lezione
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Personal Trainer
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Personal Trainer: Desidera accedere a Gym4U per visualizzare l'elenco degli iscritti prenotati per una specifica lezione, in modo da prepararsi adeguatamente prima dell'inizio della lezione.</li> <li>• Palestra: Desidera fornire ai Personal Trainer la possibilità di accedere facilmente alle informazioni sui partecipanti prenotati per le lezioni da loro supervisionate.</li> <li>• Cliente: Desidera che il Personal Trainer sia preparato e consapevole delle caratteristiche della classe, numero ed età, a cui terrà lezioni per ricevere un'esperienza di allenamento personalizzata e soddisfacente.</li> </ul>
<b>Pre-condizioni</b>	Il Personal Trainer è autenticato nel sistema Gym4U.
<b>Garanzia di successo</b>	Il Personal Trainer visualizza correttamente l'elenco degli iscritti prenotati per una specifica lezione.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Il Personal Trainer accede alla sezione "Visualizza Prenotati di una Lezione" su Gym4U.</li> <li>2. Il Sistema mostra l'elenco delle prossime lezioni assegnate al Personal Trainer.</li> <li>3. Il Personal Trainer seleziona la lezione di interesse.</li> <li>4. Il Sistema mostra un elenco dettagliato degli iscritti prenotati per la lezione selezionata.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Personal Trainer riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.           <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:               <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Personal Trainer, registra l'errore e passa in uno stato pulito.</li> <li>ii. Il Personal Trainer inizia una nuova visualizzazione dei prenotati di una lezione.</li> </ol> </li> </ol> </li> </ol> <p><b>2.a:</b> Non viene mostrato nessun corso o lezione:</p> <p>1a: Il Sistema segnala un errore nella visualizzazione dei corsi:</p>

	<p>a. Il Sistema invita il Personal Trainer a riprovare.</p> <p>1b: Il Sistema ritorna un elenco vuoto perché il Personal Trainer non ha nessun corso o lezione a suo carico:</p> <ul style="list-style-type: none"> <li>a. Il Sistema comunica al Personal Trainer che non ha corsi o lezioni a suo carico.</li> </ul> <p><b>4.a:</b> Il Sistema non mostra correttamente l'elenco dei prenotati:</p> <ol style="list-style-type: none"> <li>1. Il Sistema segnala un errore nella visualizzazione delle informazioni.</li> <li>2. Il Sistema invita il Personal Trainer a riprovare.</li> </ol>
<b>Requisiti speciali</b>	Interfaccia utente chiara e dettagliata per la consultazione delle informazioni sugli iscritti prenotati.
<b>Elenco delle varianti tecnologiche e dei dati</b>	Archiviazione delle informazioni sugli iscritti prenotati per le lezioni.
<b>Frequenza di ripetizioni</b>	Variabile, dipende dalla frequenza delle lezioni tenute dal Personal Trainer e dalla necessità di prepararsi prima dell'inizio delle lezioni.
<b>Varie</b>	

## 10. Ingresso in Palestra tramite badge

<b>Nome</b>	UC10: Accesso in Palestra tramite Badge
<b>Portata</b>	Gestione sistema Gym4U
<b>Livello</b>	Obiettivo utente
<b>Attore primario</b>	Cliente
<b>Parti interessate e interessi</b>	<ul style="list-style-type: none"> <li>• Cliente: Desidera accedere alla palestra tramite il proprio badge e confermare la propria presenza.</li> <li>• Palestra: Desidera tenere traccia dei clienti che accedono all'istituzione per motivi di sicurezza, controllo degli accessi e gestione degli utenti all'interno delle strutture.</li> </ul>
<b>Pre-condizioni</b>	Il Cliente ha prenotato un'attività o una sessione in palestra tramite Gym4U.
<b>Garanzia di successo</b>	Il Cliente ottiene l'accesso alla palestra passando il proprio badge univoco.
<b>Scenario principale di successo</b>	<ol style="list-style-type: none"> <li>1. Il Cliente si presenta al tornello d'ingresso della palestra.</li> <li>2. Il Cliente mostra il proprio badge all'apposito lettore.</li> <li>3. Il Sistema verifica la validità del badge.</li> <li>4. Il Sistema verifica che il Cliente abbia una prenotazione valida.</li> <li>5. Il Sistema conferma la presenza del Cliente.</li> <li>6. Il tornello è libero di poter girare per far entrare il Cliente.</li> </ol>
<b>Estensioni</b>	<p><b>*a:</b> In qualsiasi momento, il Sistema fallisce:</p> <ol style="list-style-type: none"> <li>1. Il Cliente riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.</li> <li>2. Il Sistema ricostruisce lo stato precedente.           <ol style="list-style-type: none"> <li>a. Il Sistema rileva delle anomalie che impediscono il ripristino:               <ol style="list-style-type: none"> <li>i. Il Sistema segnala un errore al Cliente, registra l'errore e passa in uno stato pulito.</li> <li>ii. Il Cliente inizia un nuovo accesso in palestra.</li> </ol> </li> </ol> </li> </ol> <p><b>3.a:</b> Il badge non è valido:</p> <ol style="list-style-type: none"> <li>1a: Il Sistema segnala che il badge non è stato riconosciuto:       <ol style="list-style-type: none"> <li>a. Il Sistema invita il Cliente a riprovare.</li> </ol> </li> <li>1b: Il Sistema segnala che il badge è stato già usato:       <ol style="list-style-type: none"> <li>a. Il Sistema invita il Cliente a contattare l'assistenza.</li> </ol> </li> </ol> <p><b>4.a:</b> Il Cliente non ha una prenotazione valida:</p> <ol style="list-style-type: none"> <li>1a: Il Cliente non ha una lezione prenotata per quel giorno:       <ol style="list-style-type: none"> <li>a. Il Sistema invita al Cliente a prenotarsi alla lezione desiderata.</li> </ol> </li> <li>1b: Il Cliente sta violando la Regola di Business R4:</li> </ol>

	<p>a. Il Sistema invita il Cliente a presentarsi al più mezz'ora prima dell'inizio della lezione per la quale è prenotato.</p> <p><b>6.a:</b> Il tornello ha un malfunzionamento:</p> <ol style="list-style-type: none"> <li>1. Il Cliente constata che il tornello non si è sbloccato.</li> <li>2. Il Cliente contatta l'assistenza.</li> </ol>
<b>Requisiti speciali</b>	Lettore NFC installato ai tornelli d'ingresso. Interfaccia veloce e affidabile per la scansione e la verifica dei badge.
<b>Elenco delle varianti tecnologiche e dei dati</b>	Utilizzo di tecnologie di scansione NFC integrate nei tornelli d'ingresso della palestra. Archiviazione delle informazioni di prenotazione dei clienti per la verifica dell'accesso. Connessione dati sicura per garantire l'affidabilità dell'accesso tramite badge.
<b>Frequenza di ripetizioni</b>	Frequente, ogni volta che i Clienti devono accedere alle attività della palestra tramite Gym4U.
<b>Varie</b>	

## **11. Visualizza Scheda Personalizzata**

Il Cliente, attraverso Gym4U, visualizza, se presente, la scheda personalizzata assegnatagli da un Personal Trainer. Questa visualizzazione include informazioni dettagliate come esercizi, giorno di inizio e giorno di fine.

## **12. Visualizza Corsi Palestra**

Il Cliente, attraverso Gym4U, visualizza l'elenco completo dei corsi offerti dalla palestra. Questa visualizzazione include informazioni dettagliate come giorni, orari e eventuali dettagli specifici relativi a ciascun corso.

## **13. Modifica Account e Password**

Il Cliente, attraverso Gym4U, può modificare i dati anagrafici e di contatto associati al proprio account e la propria Password.

## **14. Modifica Cliente**

L'amministratore, tramite Gym4U, ha la possibilità di modificare i dati anagrafici e di contatto associati a un Cliente. Ciò include informazioni come nome, cognome, email e numero di telefono.

## **15. Aggiungi nuovo Personal Trainer**

L'amministratore accede a Gym4U per aggiungere un nuovo Personal Trainer al sistema, permettendo l'espansione del team di Personal Trainer all'interno della palestra.

## Documento di Visione

Gym4U si propone di diventare la soluzione di riferimento nel settore della gestione di una palestra, offrendo un sistema completo che semplifica le attività amministrative e ottimizza l'interazione tra la struttura e i suoi iscritti. La visione del progetto è quella di fornire una piattaforma versatile, user-friendly e altamente efficiente, contribuendo al successo e alla crescita della struttura che adotterà questa soluzione.

Per il documento di visione approfondito vedere documento specifico.

## Regole di business

Per il corretto utilizzo del sistema devono essere rispettate le seguenti regole di dominio:

ID	Regola	Modificabilità	Sorgente
R1	Per accedere al sistema, gli iscritti devono essere in possesso di un badge valido.	Bassa.	Il badge è un requisito obbligatorio per politica interna della palestra.
R2	Per iscriversi ai corsi, gli iscritti devono aver sottoscritto un abbonamento che include la tipologia di corso desiderata ed aver caricato il certificato medico.	Bassa.	Politica interna della palestra e normativa europea.
R3	Acquistando un Abbonamento Semestrale il prezzo mensile è scontato di 5€	Bassa.	Politica interna alla palestra.
R4	Acquistando un Abbonamento Annuale il prezzo mensile è scontato di 10€	Bassa.	Politica interna alla palestra.
R5	Un iscritto può prenotare una sola lezione al giorno	Media.	Politica interna alla palestra.
R6	Il Cliente può accedere alla struttura solo se è prenotato e se mancano meno di 30 minuti	Bassa.	Politica interna alla palestra.

	all'inizio della lezione per la quale è prenotato .		
R7	Un cliente può registrarsi e accedere al sistema Gym4U, nonché alla struttura fisica della palestra, solo se ha un'età superiore a 18 anni.	Alta.	Politica interna alla palestra.
R8	Il Cliente può prenotare una Lezione privata con un Personal Trainer per una durata di un'ora.	Media.	Politica interna alla palestra.
R9	Un Personal Trainer può gestire al più 5 corsi.	Alta	Politica interna alla palestra.

## Specifiche supplementari

1. Usabilità
  - a. L'interfaccia grafica deve essere intuitiva e semplice anche per un utente non esperto.
  - b. L'interazione con il sistema non deve presentare un elevato grado di complessità.
2. Affidabilità e performance
  - a. Il software sviluppato deve essere affidabile e deve poter mantenere i propri dati anche in caso di guasti (guasti elettrici, usura dell'hardware).
  - b. Assicurare alta disponibilità del sistema, tempi di risposta rapidi e riduzione dei tempi di inattività.
3. Vincoli hardware e software
  - a. Per eseguire il software non ci sono particolari requisiti per il sistema operativo purché sia presente la Java Virtual Machine.
  - b. Per inviare le comunicazioni agli iscritti è indispensabile avere una connessione a Internet attiva.
  - c. Il software deve assicurare la sicurezza e la privacy dei dati personali e sensibili dei clienti e del personale della palestra, e deve rispettare le normative vigenti in materia di protezione dei dati e di certificazione medica.
  - d. Il software deve essere scalabile e flessibile, in modo da poter adattarsi alle diverse esigenze e dimensioni delle palestre che lo adottano, e deve consentire l'aggiunta di nuove funzionalità e moduli in base alle richieste del mercato e dei clienti.
4. Vincoli di sviluppo del software
  - a. Tutto il software è scritto usando Java e sfrutta un database per gestire la persistenza dei dati.

## Glossario

- **Gym4U:** Software dedicato alla gestione e all'ottimizzazione delle attività di una palestra, che semplifica l'amministrazione e migliora l'esperienza degli utenti.
- **Amministratore di Sistema:** Gestore delle attività quotidiane della palestra, utilizzatori principali della componente amministrativa del software Gym4U.
- **Cliente:** Cliente della palestra che utilizza l'applicativo Gym4U per gestire le proprie iscrizioni, prenotazioni, e partecipare alle attività.
- **Personal Trainer:** Professionista coinvolto nell'erogazione di corsi e sessioni personalizzate, utilizza Gym4U per la pianificazione e la comunicazione con gli iscritti.
- **Corso:** Un programma strutturato offerto dalla palestra, che può comprendere diverse sessioni o lezioni. I corsi sono generalmente focalizzati su specifiche attività fisiche o obiettivi di fitness e sono gestiti dai personal trainer.
- **Lezione:** Una singola sessione di allenamento o attività all'interno di un corso. Le lezioni possono essere prenotate individualmente dagli utenti tramite l'applicazione Gym4U e sono spesso parte di un corso più ampio.
- **Abbonamento:** Un accordo contrattuale tra l'utente e la palestra, gestito attraverso Gym4U, che dà accesso agli utenti a corsi, lezioni e altre risorse della palestra per un periodo di tempo definito. Gli abbonamenti possono variare in termini di durata e costo.
- **Offerta:** Promozioni e incentivi personalizzati forniti dal sistema Gym4U per aumentare la partecipazione e la fedeltà degli utenti.
- **Scheda:** Un piano personalizzato di esercizi fornito da un personal trainer a un cliente. La scheda include una serie di esercizi specifici al fine di raggiungere gli obiettivi di fitness del cliente. Attraverso Gym4U, i personal trainer possono creare e gestire le schede dei loro clienti.
- **Badge:** Un dispositivo fisico utilizzato per autenticare e monitorare l'accesso degli iscritti alla palestra. Il badge può essere dotato di tecnologie NFC per facilitare l'ingresso automatico e registrare la presenza degli utenti.
- **Team di Sviluppo e IT:** Responsabili dello sviluppo e della manutenzione del software Gym4U, lavorano sulla progettazione, implementazione di funzionalità avanzate e manutenzione del sistema.

# Analisi orientata agli oggetti

## Introduzione

L'implementazione dell'applicazione ha seguito l'approccio iterativo ed evolutivo proposto da UP (Unified Process), articolandosi in quattro iterazioni. Questo approccio ha consentito di sviluppare in modo progressivo il nucleo dell'architettura del software. Durante ogni iterazione, sono stati affrontati e risolti iterativamente i problemi legati ai rischi più significativi. Inoltre, è stata condotta un'analisi graduale dei requisiti, limitando così al minimo il potenziale danno derivante da eventuali errori di progettazione e implementazione.

Ciascuna iterazione ha affrontato specifiche problematiche, gestendo in particolare:

→ Iterazione 1:

- ◆ Implementare lo scenario previsto dai casi d'uso UC1: "Iscrizione ai corsi", UC2: "Prenotazione lezione corso", a meno del passo 11 e dello scenario alternativo 7.a, e lo UC3: "Creazione nuovo corso".
- ◆ Implementare i casi d'uso di Start Up necessari per gestire le esigenze di inizializzazione per questa iterazione Documento

→ Iterazione 2:

- ◆ Implementare lo scenario alternativo 7.a del caso d'uso UC2: "Prenotazione lezione corso" dove il cliente tenta di prenotare una lezione in conflitto con la Regola di Business 5, poiché è già prenotato per un'altra lezione nello stesso giorno.
- ◆ Implementare lo scenario previsto dai casi d'uso UC4: "Registrazione nuovo cliente", UC5: "Gestione abbonamento", UC6: "Prenotazione Personal Trainer" ed UC10: "Ingresso in palestra tramite badge".

L'approccio di analizzare insieme questi è mirata ad agevolare una visione globale del percorso del cliente all'interno del sistema, fornendo una comprensione più completa e continua delle attività correlate che il cliente può svolgere da registrato a partecipante alle sessioni in palestra.

→ Iterazione 3:

- ◆ Implementare lo scenario previsto dai casi d'uso UC7: "Creazione di un'offerta promozionale", UC8: "Creazione di una scheda personalizzata" ed UC9: "Visualizza lezioni".

L'approccio di analizzare insieme questi casi d'uso è mirata ad agevolare una visione globale del percorso del Personal Trainer all'interno del sistema, fornendo una comprensione più completa e continua delle attività correlate che il Personal Trainer può svolgere all'interno della palestra.

#### → Iterazione 4:

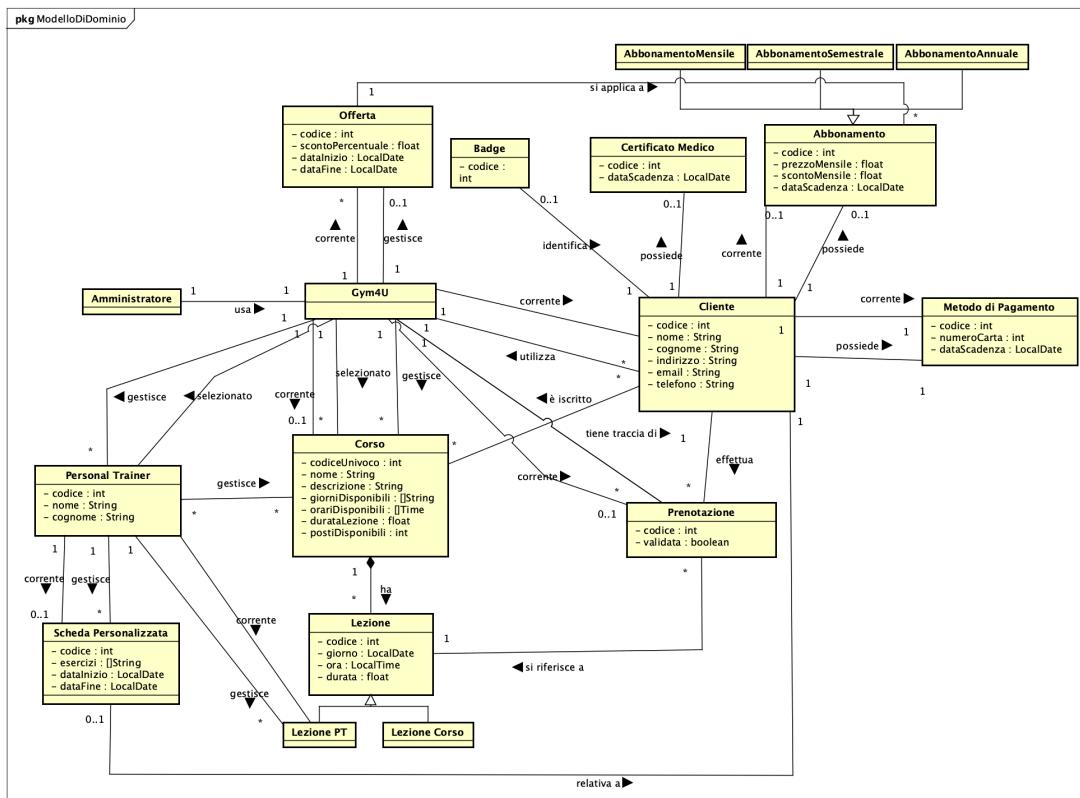
- ◆ Implementazione scenario alternativo UC3 2.a facendo valere la Regola di Business R9.
- ◆ Implementazione modifiche collaterali dell'UC7 agli UC4 e UC5.
- ◆ Implementazione passo 11 UC2.
- ◆ Modifiche del Modello di Dominio in ottica Low Coupling ed High Cohesion.
- ◆ Gestione della schermata di accesso e relative modifiche.

Questa iterazione si propone di consolidare e arricchire il sistema, affrontando aspetti chiave che contribuiranno a definire una versione sempre più completa e performante dell'applicazione Gym4U.

All'inizio di ogni iterazione, è stato condotto un processo di analisi dei requisiti orientato agli oggetti. Questo tipo di analisi si basa sulla creazione di una descrizione del dominio da una prospettiva orientata agli oggetti. Per fornire tale descrizione, sono stati impiegati diversi strumenti, tra cui il Modello di Dominio, il Diagramma di Sequenza di Sistema (SSD) e i Contratti delle Operazioni.

## Modello di dominio

La disciplina all'interno di UP che si dedica a fornire dettagli sul dominio è la Modellazione del Business. Questa fase, in particolare, incorpora la creazione del Modello di Dominio, un elaborato grafico in cui vengono identificati i concetti, gli attributi e le associazioni considerati significativi nell'ambito del dominio di riferimento. Alla luce dei contributi derivanti da ciascuna iterazione, il modello di dominio si configura come segue:



- **Gym4U:** Rappresenta l'applicazione o la piattaforma stessa, il sistema software che gestisce l'intero ecosistema di servizi offerti dalla palestra.
- **Amministratore:** è l'utente con privilegi speciali e di gestione all'interno della piattaforma Gym4U. Gestisce l'iscrizione dei clienti, i corsi e altri aspetti amministrativi della palestra;
- **Personal Trainer:** rappresenta il Personal Trainer che gestisce i corsi della palestra;
- **Cliente:** cliente della palestra che effettua iscrizione ai corsi e prenotazione alle lezioni;
- **Abbonamento:** Rappresenta i diversi tipi di abbonamenti disponibili per i clienti, come mensili, trimestrali o annuali, offrendo accesso a determinati servizi o corsi specifici;
- **Abbonamento Mensile:** Rappresenta una sottoclasse di "Abbonamento" specificamente per la durata mensile.
- **Abbonamento Semestrale:** Rappresenta una sottoclasse di "Abbonamento" specificamente per la durata semestrale.
- **Abbonamento Annuale:** Rappresenta una sottoclasse di "Abbonamento" specificamente per la durata annuale.
- **Certificato medico:** Rappresenta il certificato medico posseduto dal cliente, avere il certificato medico in corso di validità permette al cliente di accedere ai corsi e alle lezioni della palestra;
- **Badge:** Rappresenta un dispositivo identificativo fornito ai clienti per l'accesso fisico alla palestra. È utilizzato per sbloccare i tornelli o l'ingresso alla struttura ed è associato all'identità del cliente tramite Gym4U.
- **Metodo di Pagamento:** Rappresenta le modalità e le informazioni utilizzate per effettuare transazioni finanziarie all'interno della piattaforma Gym4U, come carta di credito, carta di debito, PayPal, etc.
- **Corso:** Rappresenta le diverse attività formative o di fitness offerte dalla palestra attraverso Gym4U, come lezioni di gruppo, sessioni di allenamento specifiche o programmi specializzati;
- **Lezione:** Indica le singole sessioni all'interno di un corso a cui i clienti possono prenotarsi. Ogni lezione ha una data, un orario e una capacità massima di partecipanti;
- **Lezione PT:** Indica le sessioni specifiche di addestramento o consulenza offerte da un Personal Trainer. Queste lezioni potrebbero differire da quelle dei corsi standard e possono essere prenotate per sessioni individuali.
- **Lezione Corso:** Rappresenta le sessioni standard offerte all'interno dei corsi generici della palestra. Queste lezioni sono parte integrante dei corsi regolari offerti dalla palestra e sono accessibili ai clienti iscritti.
- **Prenotazione:** Rappresenta l'atto di prenotare una lezione specifica all'interno di un corso per un cliente registrato su Gym4U.
- **Scheda Personalizzata:** Rappresenta una scheda personalizzata, composta da esercizi, una data di inizio e una data di fine, viene creata da un personal trainer e viene assegnata ad un cliente.
- **Offerta:** Rappresenta una promozione o uno sconto temporaneo (dataInizio e dataFine) applicato agli abbonamenti. Ogni offerta è caratterizzata da uno scontoPercentuale che indica la percentuale di sconto applicata.

In particolare, nella quarta iterazione vengono effettuate delle modifiche molto importanti riguardo il Modello di Dominio che mirano ad un consolidamento della piattaforma grazie all'eliminazione dell'associazione "corrente" tra Gym4U e alcune classi, seguendo il principio di progettazione orientata agli oggetti di Low Coupling e High Cohesion.

- **Low Coupling:**

Prima della modifica, l'associazione "corrente" tra Gym4U e altre classi indicava una dipendenza diretta, creando un alto livello di accoppiamento tra Gym4U e le altre classi coinvolte.

Rimuovendo questa associazione, si riduce la dipendenza diretta tra Gym4U e le classi correlate, migliorando la modularità del sistema. Ora, Gym4U non è direttamente coinvolto nella gestione di oggetti specifici come SchedaPersonalizzata, Lezione, Abbonamento, e MetodoDiPagamento.

- **High Cohesion:**

Prima della modifica, l'associazione "corrente" poteva suggerire una responsabilità eccessiva a Gym4U, coinvolgendo direttamente nella gestione di diverse classi. Separando le responsabilità, le classi come PersonalTrainer, Cliente, SchedaPersonalizzata, Lezione, Abbonamento, e MetodoDiPagamento sono più coese e focalizzate sulle loro specifiche responsabilità. Ad esempio, la gestione di una SchedaPersonalizzata è ora responsabilità del PersonalTrainer, mentre un Abbonamento è gestito dal Cliente.

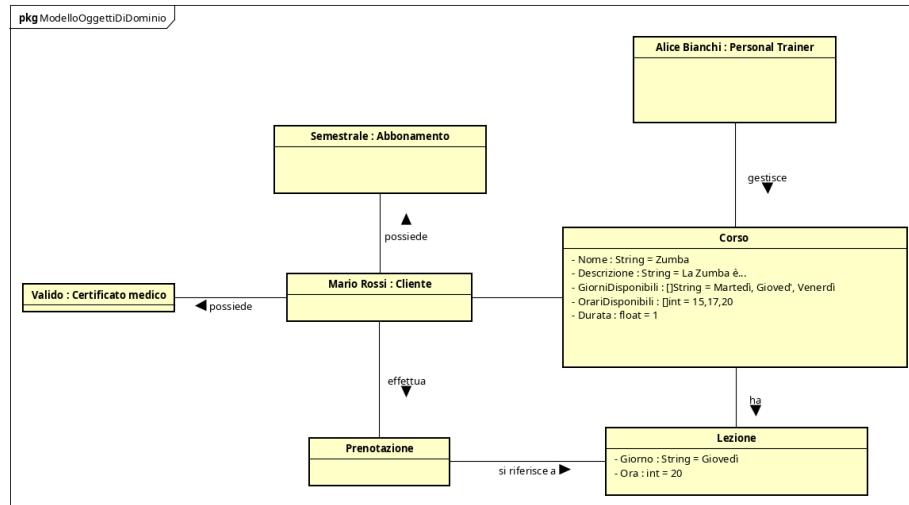
In questo modo, la progettazione diventa più flessibile e manutenibile, consentendo futuri cambiamenti o estensioni in modo più agevole senza influenzare negativamente altre parti del sistema. La separazione delle responsabilità migliora la chiarezza del codice e facilita la comprensione e la manutenzione a lungo termine.

## Modello degli oggetti

Il modello degli oggetti di dominio offre una visione schematica e strutturata degli oggetti chiave, delle loro relazioni e degli attributi all'interno di un sistema. Questo modello aiuta a visualizzare e comprendere meglio la struttura e l'organizzazione dei dati nel dominio di interesse, fornendo una panoramica chiara delle entità coinvolte e delle loro interazioni.

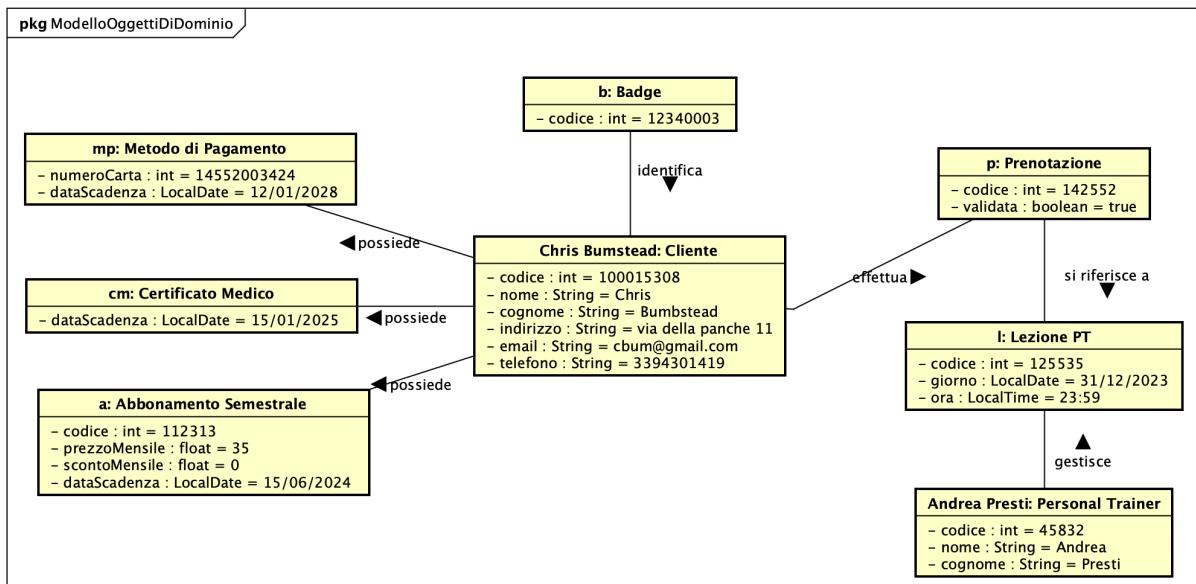
### → Iterazione 1:

“Il cliente Mario Rossi, con un abbonamento semestrale, già iscritto al corso di Zumba tenuto dal Personal Trainer Alice Bianchi, si è appena prenotato alla lezione di giovedì alle ore 20.”



### → Iterazione 2:

“Il Cliente Chris Bumbstead, che possiede un Abbonamento Semestrale attivo, un Certificato Medico valido ed un Metodo di Pagamento, ha validato grazie al suo Badge la Prenotazione riferita alla Lezione del 31/12/2023 alle 23:59 con il Personal Trainer Andrea Presti”



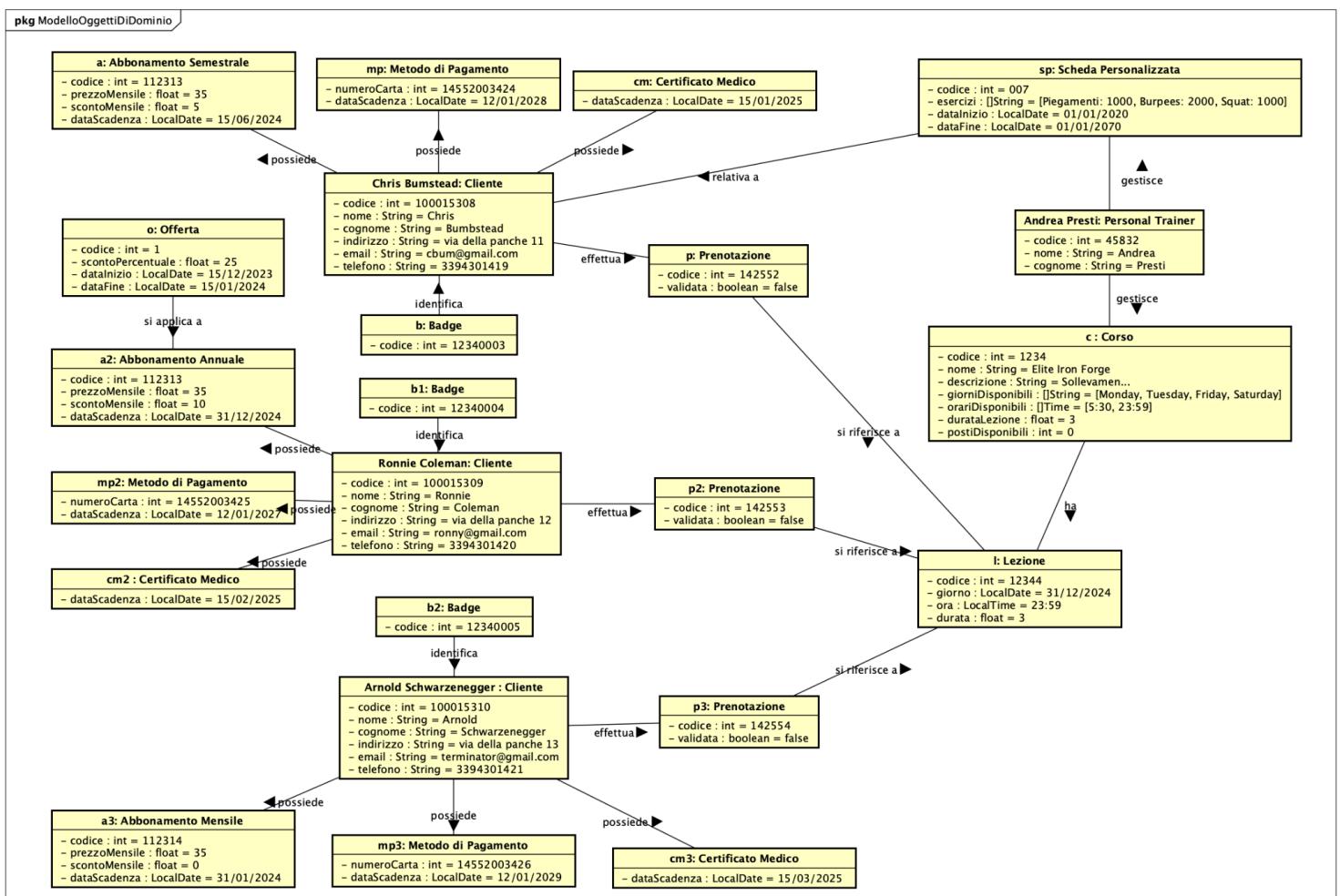
## → Iterazione 3:

“Il Personal Trainer Andrea Presti ha visualizzato tra gli iscritti di una sua Lezione di giorno 31/12/2024 alle 23:59 del Corso di Elite Iron Forge:

il Cliente Chris Bumstead, che possiede un Abbonamento Semestrale attivo, un Certificato Medico valido ed un Metodo di Pagamento, a cui ha in precedenza assegnato una Scheda Personalizzata;

il Cliente Ronnie Coleman, che possiede un Abbonamento Annuale attivo, a cui è stato applicata un'Offerta, un Certificato Medico valido ed un Metodo di Pagamento;

il Cliente Arnold Schwarzenegger, che possiede un Abbonamento Mensile Attivo, un Certificato Medico valido ed un Metodo di Pagamento”



## → Iterazione 4:

Non vi è alcuna modifica rispetto all'iterazione precedente.

## SSD e Contratti

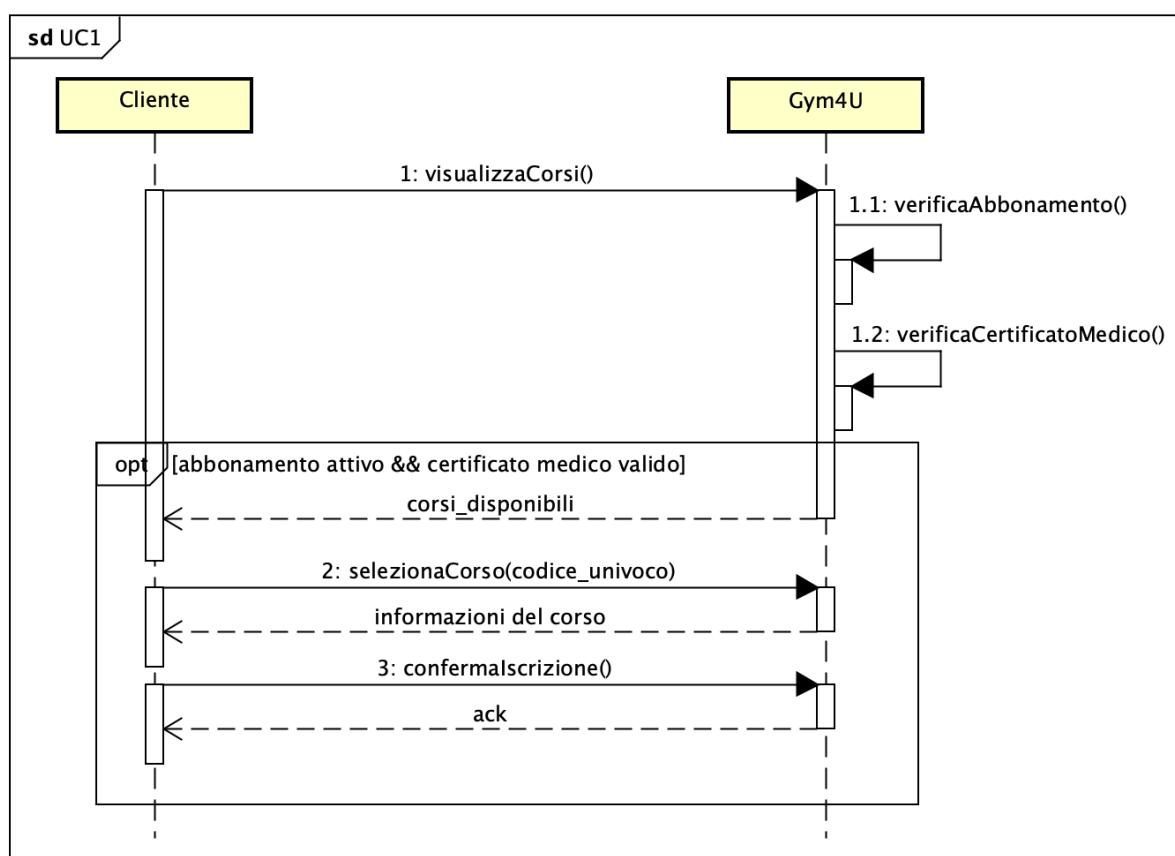
Nel contesto dell'analisi orientata agli oggetti, la successiva tappa consiste nella creazione dei Diagrammi di Sequenza di Sistema (SSD) al fine di delineare il flusso degli eventi di input e output relativi ai vari casi d'uso analizzati in ciascuna iterazione. Questi diagrammi forniscono una rappresentazione 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.

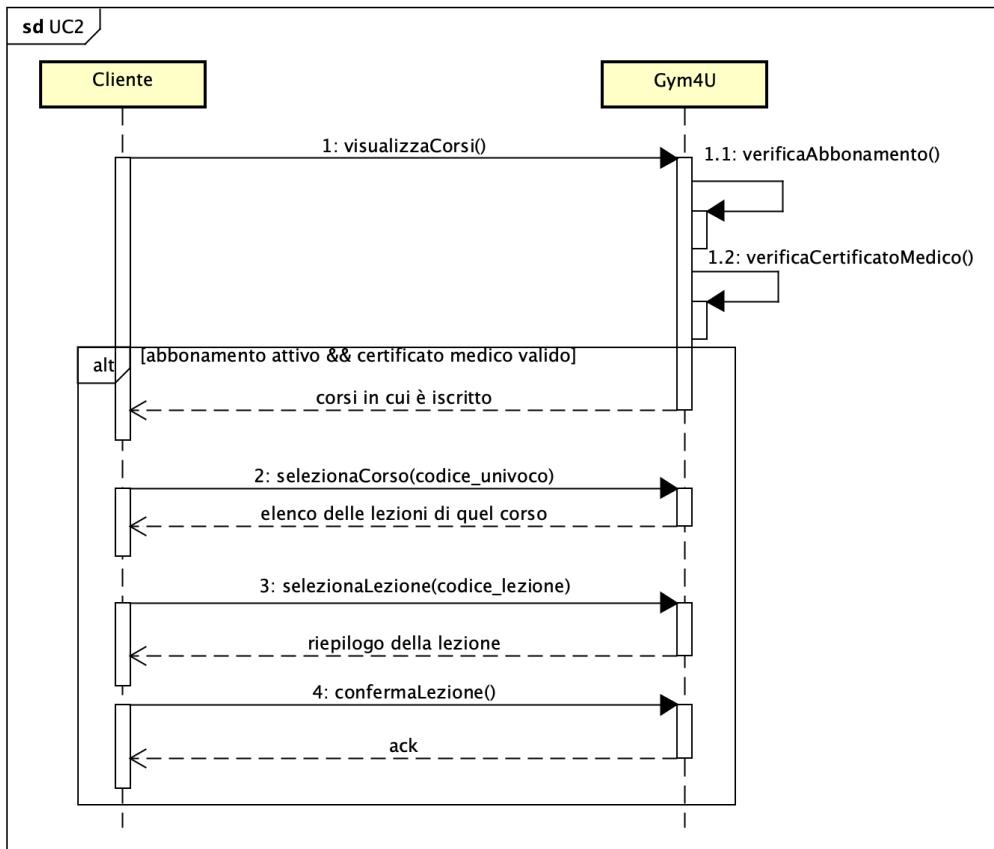
In questo modo, si ottiene una visione dettagliata e chiara del funzionamento del sistema, consentendo una migliore comprensione delle interazioni e delle responsabilità delle componenti coinvolte. Tale approccio facilita anche la fase successiva di progettazione, fornendo una base solida per l'implementazione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

→ Iterazione 1:

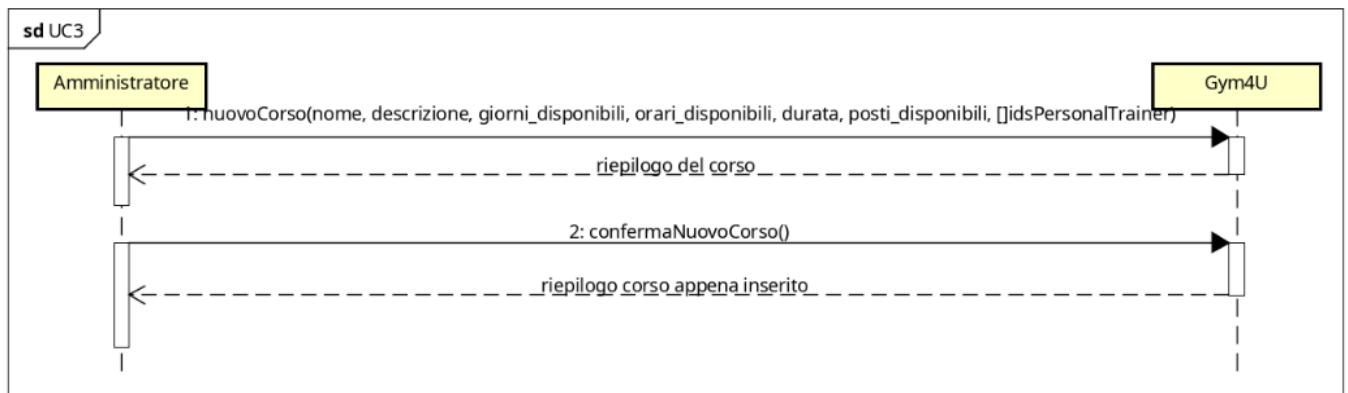
SSD Caso d'Uso 1



## SSD Caso d'Uso 2



## SSD Caso d'Uso 3



## Contratti Operazioni UC1

### Contratto CO1: *selezionaCorso*

<b>Operazione</b>	selezionaCorso(codice_univoco: int)
<b>Riferimenti</b>	Caso d'Uso UC1: Iscrizione ai corsi
<b>Pre-Condizioni</b>	Esiste una lista di Corsi disponibili cd
<b>Post-Condizioni</b>	È stata selezionata un'istanza c dalla lista dei Corsi disponibili cd

### Contratto CO2: *confermalscrizione*

<b>Operazione</b>	confermalscrizione()
<b>Riferimenti</b>	Caso d'Uso UC1: Iscrizione ai corsi
<b>Pre-Condizioni</b>	Esiste un'istanza corrente c di Corso
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• c è stato associato a cliente tramite l'associazione "è iscritto"</li> <li>• È stata aggiornata la disponibilità dei posti di c</li> <li>• È stata eliminata l'associazione "selezionato" tra c e Gym4U</li> </ul>

## Contratti Operazioni UC2

### Contratto CO1: *selezionaCorso*

<b>Operazione</b>	selezionaCorso(codice_univoco: int)
<b>Riferimenti</b>	Caso d'Uso UC2: Prenotazione lezione corsi
<b>Pre-Condizioni</b>	Esiste una lista di Corsi del Cliente cc
<b>Post-Condizioni</b>	È stata selezionata l'istanza di corso c dalla lista dei Corsi del Cliente cc

### Contratto CO2: *selezionaLezione*

<b>Operazione</b>	selezionaLezione(codice: int)
<b>Riferimenti</b>	Caso d'Uso UC2: Prenotazione lezione corsi
<b>Pre-Condizioni</b>	Il cliente è autenticato ed iscritto al corso
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza p di Prenotazione</li> <li>• È stata selezionata l'istanza l di Lezione</li> </ul>

Contratto CO3: *confermaLezione*

<b>Operazione</b>	confermaLezione()
<b>Riferimenti</b>	Caso d'Uso UC2: Prenotazione lezione corsi
<b>Pre-Condizioni</b>	È stata creata un'istanza p di Prenotazione
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza p è stata associata alla Lezione l tramite l'associazione "si riferisce a"</li> <li>• L'istanza p è stata associata al Cliente cliente tramite l'associazione "effettua"</li> <li>• L'istanza p viene associata a Gym4U tramite l'associazione "tiene traccia di"</li> <li>• È stata eliminata l'associazione corrente tra p e Gym4U</li> <li>• È stata eliminata l'associazione "selezionata" tra c e Gym4U</li> </ul>

Contratti Operazioni UC3

Contratto CO1: *nuovoCorso*

<b>Operazione</b>	nuovoCorso(nome:String, descrizione:String, giorni_disponibili: []String, orari_disponibili: []int, durata: float, posti_disponibili: int, idsPersonalTrainer: []int)
<b>Riferimenti</b>	Caso d'Uso UC3: Creazione nuovo corso
<b>Pre-Condizioni</b>	L'amministratore è autenticato in Gym4U
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata l'istanza c di Corso</li> <li>• c è stata inizializzata con le informazioni inserite</li> <li>• c è stata associata a Gym4U tramite l'associazione "corrente"</li> </ul>

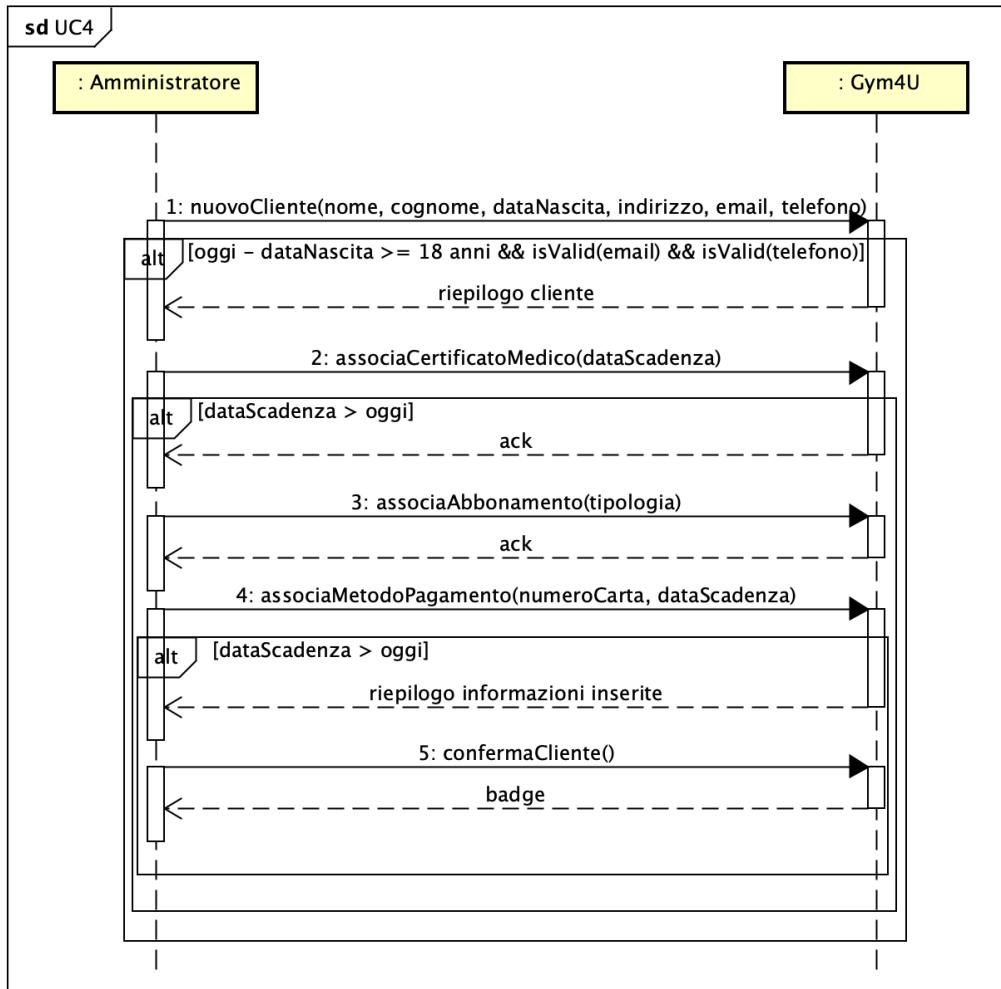
Contratto CO3: *confermaNuovoCorso*

<b>Operazione</b>	confermaNuovoCorso()
<b>Riferimenti</b>	Caso d'Uso UC3: Creazione nuovo corso
<b>Pre-Condizioni</b>	È in corso la creazione di un nuovo Corso c
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza corrente c è stata associata a uno o più Personal Trainer tramite l'associazione "gestisce"</li> <li>• L'istanza c è stata associata a Gym4U tramite l'associazione "gestisce"</li> <li>• È stata eliminata l'associazione "corrente" tra Gym4U e l'istanza c</li> </ul>

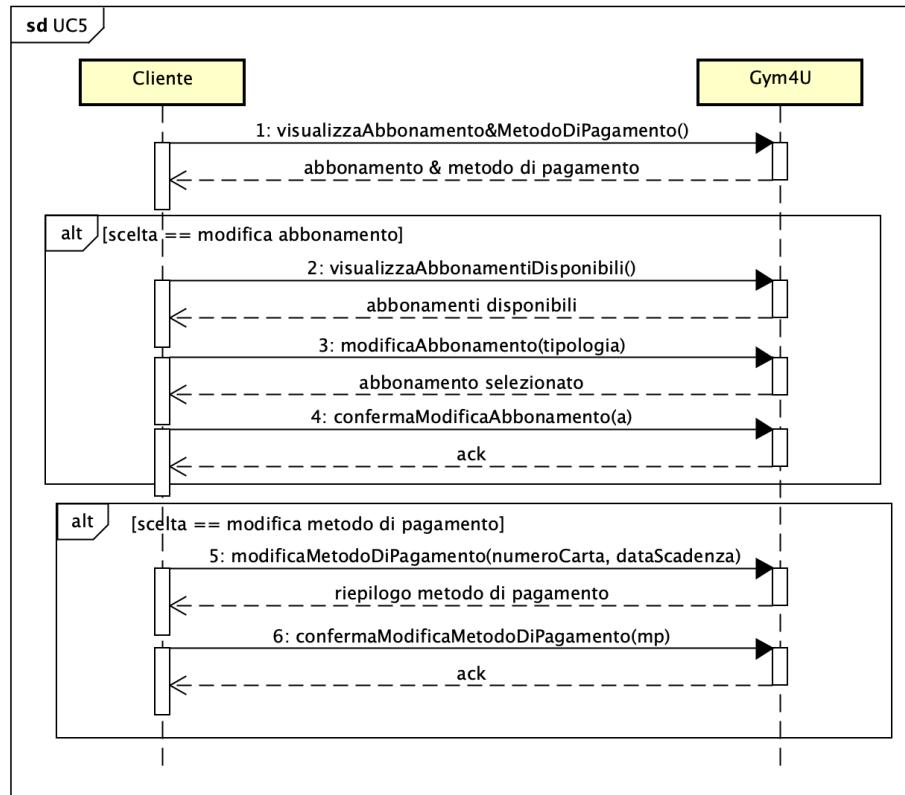
→ Iterazione 2:

SSD Caso d'Uso 4

Con il relativo alt: "oggi - dataNascita >= 18 anni && isValid(email) && isValid(telefono)" viene fatta valere la Regola di Business R7, viene previsto lo scenario 2a; Vengono previsti anche gli scenari 4.a e 6.a con i relativi alt: dataScadenza > oggi per controllare che i dati inseriti siano validi.

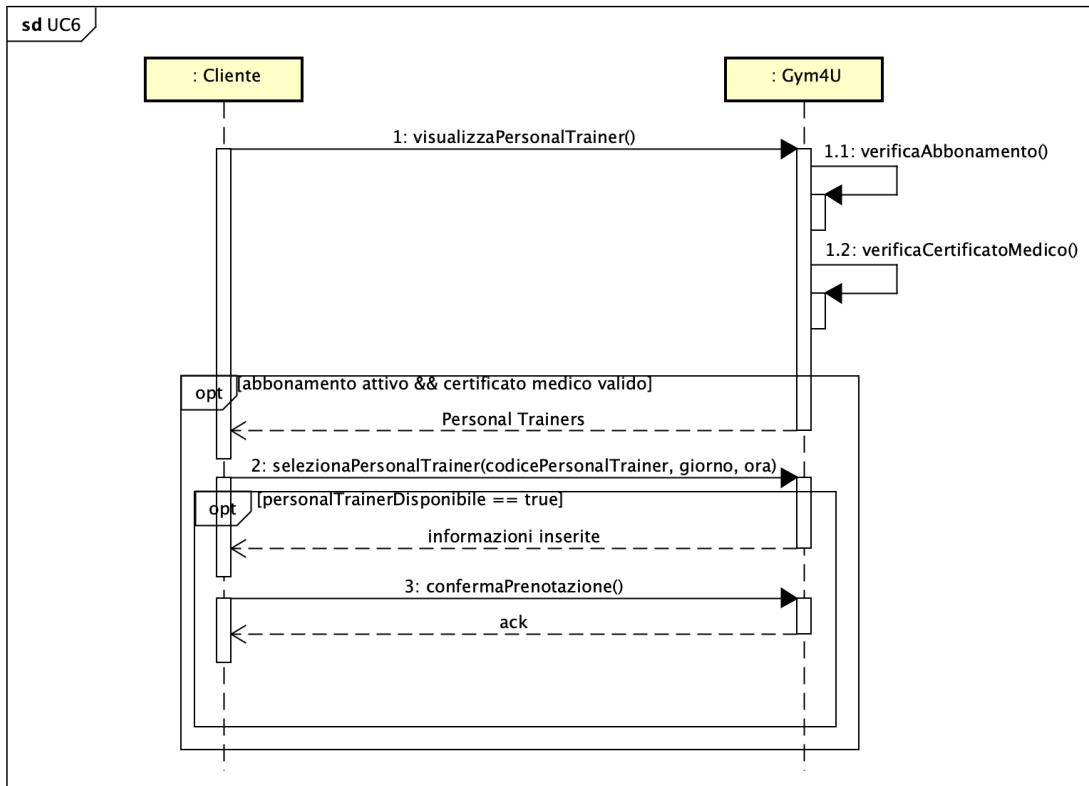


## SSD Caso d'Uso 5

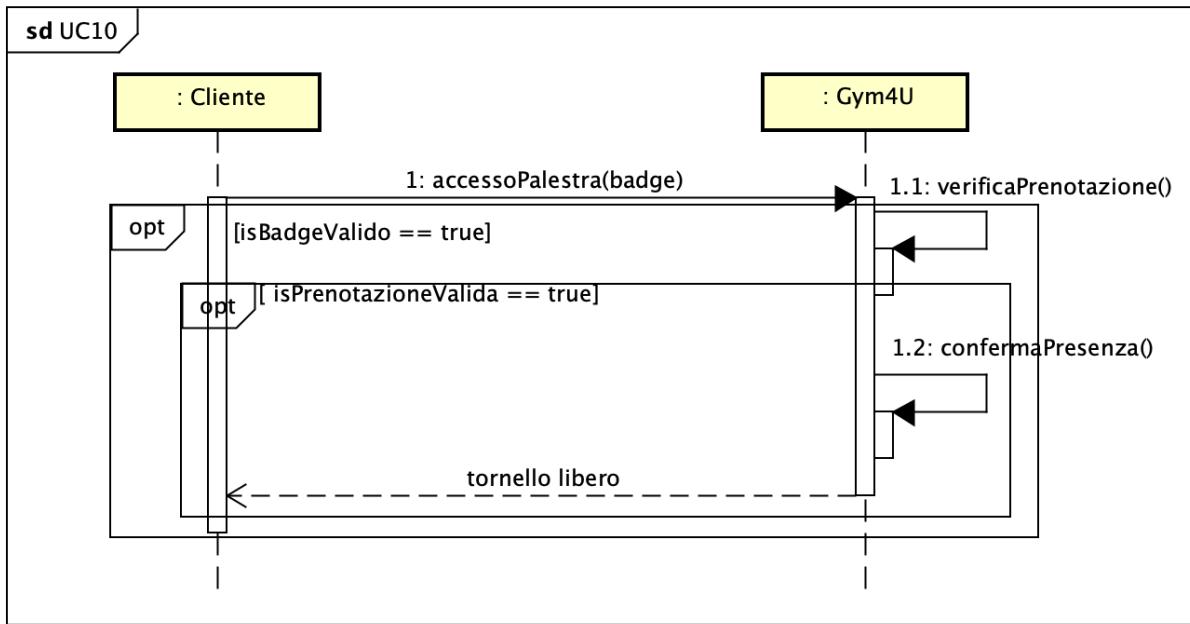


## SSD Caso d'Uso 6

Con il relativo alt: "personalTrainerDisponibile == true" viene fatta valere la Regola di Business R8, viene previsto anche lo scenario 6a.



## SSD Caso d'Uso 10



### Contratti Operazioni UC4

#### Contratto CO1: *nuovoCliente*

<b>Operazione</b>	<code>nuovoCliente(nome: String, cognome: String, indirizzo: String, email: String, telefono: String)</code>
<b>Riferimenti</b>	Caso d'Uso UC4: Registrazione nuovo Cliente
<b>Pre-Condizioni</b>	L'amministratore è autenticato in Gym4U
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza cliente di Cliente</li> <li>• cliente è stato inizializzato con le informazioni inserite</li> <li>• cliente è stato associato a Gym4U tramite l'associazione "corrente"</li> </ul>

#### Contratto CO2: *associaCertificatoMedico*

<b>Operazione</b>	<code>associaCertificatoMedico(dataScadenza: LocalDate)</code>
<b>Riferimenti</b>	Caso d'Uso UC4: Registrazione nuovo Cliente
<b>Pre-Condizioni</b>	È in corso la registrazione di un nuovo Cliente cliente
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza cm di Certificato Medico</li> <li>• cm è stato inizializzato con le informazioni inserite</li> <li>• cm è stato associato a Cliente tramite l'associazione "possiede"</li> </ul>

Contratto CO3: *associaAbbonamento*

<b>Operazione</b>	associaAbbonamento(tipologia: String)
<b>Riferimenti</b>	Caso d'Uso UC4: Registrazione nuovo Cliente
<b>Pre-Condizioni</b>	È in corso la registrazione di un nuovo Cliente cliente
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza a di Abbonamento</li> <li>• a è stato inizializzato con le informazioni inserite</li> <li>• a è stato associato a Cliente tramite l'associazione "possiede"</li> </ul>

Contratto CO4: *associaMetodoPagamento*

<b>Operazione</b>	associaMetodoPagamento(numeroCarta: String, dataScadenza: LocalDate)
<b>Riferimenti</b>	Caso d'Uso UC4: Registrazione nuovo Cliente
<b>Pre-Condizioni</b>	È in corso la registrazione di un nuovo Cliente cliente
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza mp di Metodo di Pagamento</li> <li>• mp è stato inizializzato con le informazioni inserite</li> <li>• mp è stato associato a Cliente tramite l'associazione "possiede"</li> </ul>

Contratto CO4: *confermaCliente*

<b>Operazione</b>	confermaCliente()
<b>Riferimenti</b>	Caso d'Uso UC4: Registrazione nuovo Cliente
<b>Pre-Condizioni</b>	È in corso la registrazione di un nuovo Cliente cliente
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza b di Badge</li> <li>• b è stato associato a cliente tramite l'associazione "identifica"</li> <li>• L'istanza corrente cliente è stata associata a Gym4U tramite l'associazione "utilizza"</li> <li>• È stata eliminata l'associazione "corrente" tra Gym4U e l'istanza cliente</li> </ul>

## Contratti Operazioni UC5

### CO1: *modificaAbbonamento*

<b>Operazione</b>	modificaAbbonamento(abbonamento: String)
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	L'Utente è autenticato ed iscritto a Gym4U.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente a di Abbonamento tramite l'associazione "corrente"</li> <li>• a è stata inizializzata con le informazioni inserite</li> </ul>

### CO2: *confermaModificaAbbonamento*

<b>Operazione</b>	confermaAbbonamento()
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	Esiste un'istanza a di Abbonamento
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza a di Abbonamento viene associata al Cliente c tramite l'associazione "possiede"</li> <li>• È stata eliminata l'associazione "corrente" tra a e c</li> </ul>

### CO3: *modificaMetodoDiPagamento*

<b>Operazione</b>	modificaMetodoDiPagamento(numeroCarta: int, dataScadenza: LocalDate)
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	L'Utente è autenticato ed iscritto a Gym4U.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente mp di MetodoDiPagamento</li> <li>• mp è stata inizializzata con le informazioni inserite</li> </ul>

### CO4: *confermaModificaMetodoDiPagamento*

<b>Operazione</b>	confermaMetodoDiPagamento()
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	Esiste una lista di Personal Trainer personalTrainers
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza mp di MetodoDiPagamento viene associata al Cliente c tramite l'associazione "possiede"</li> <li>• È stata eliminata l'associazione "corrente" tra mp e c</li> </ul>

<b>Operazione</b>	selezionaPersonalTrainer(codicePersonalTrainer: int, giorno: LocalDate, ora: LocalTime)
<b>Riferimenti</b>	Caso d'Uso UC6: Prenotazione Personal Trainer
<b>Pre-Condizioni</b>	Esiste una lista di Personal Trainer personalTrainers
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente l di LezionePT</li> <li>• l è stata inizializzata con le informazioni inserite</li> <li>• È stata selezionata l'istanza pt di Personal Trainer</li> </ul>

#### Contratti Operazioni UC6

##### Contratto CO2: *confermaPrenotazione*

<b>Operazione</b>	confermaPrenotazione()
<b>Riferimenti</b>	Caso d'Uso UC6: Prenotazione Personal Trainer
<b>Pre-Condizioni</b>	<ul style="list-style-type: none"> <li>• È in corso la prenotazione di una Lezione Personal Trainer</li> <li>• Esiste un'istanza l di LezionePT</li> </ul>
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza p di Prenotazione</li> <li>• L'istanza p è stata associata alla Lezione l</li> <li>• L'istanza p è stata associata al Cliente cliente</li> <li>• L'istanza p viene associata a Gym4U tramite l'associazione "tiene traccia di"</li> <li>• È stata eliminata l'associazione "corrente" tra l e pt</li> <li>• L'istanza l è stato associata a pt tramite l'associazione "gestisce"</li> </ul>

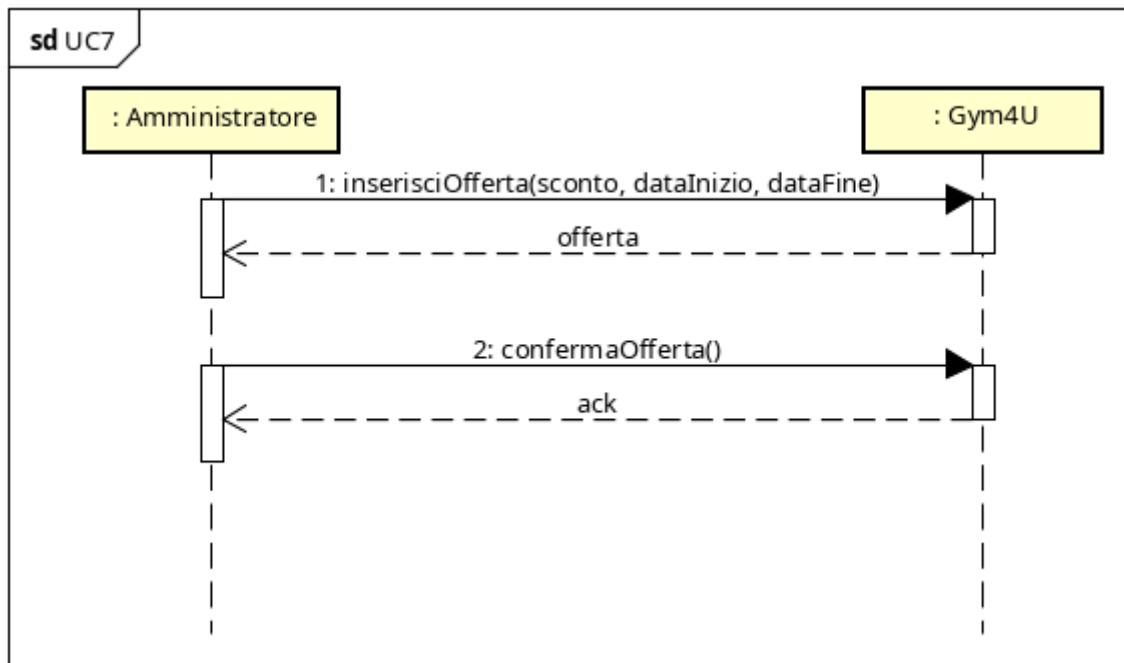
#### Contratti Operazioni UC10

##### Contratto CO1: *confermaPresenza*

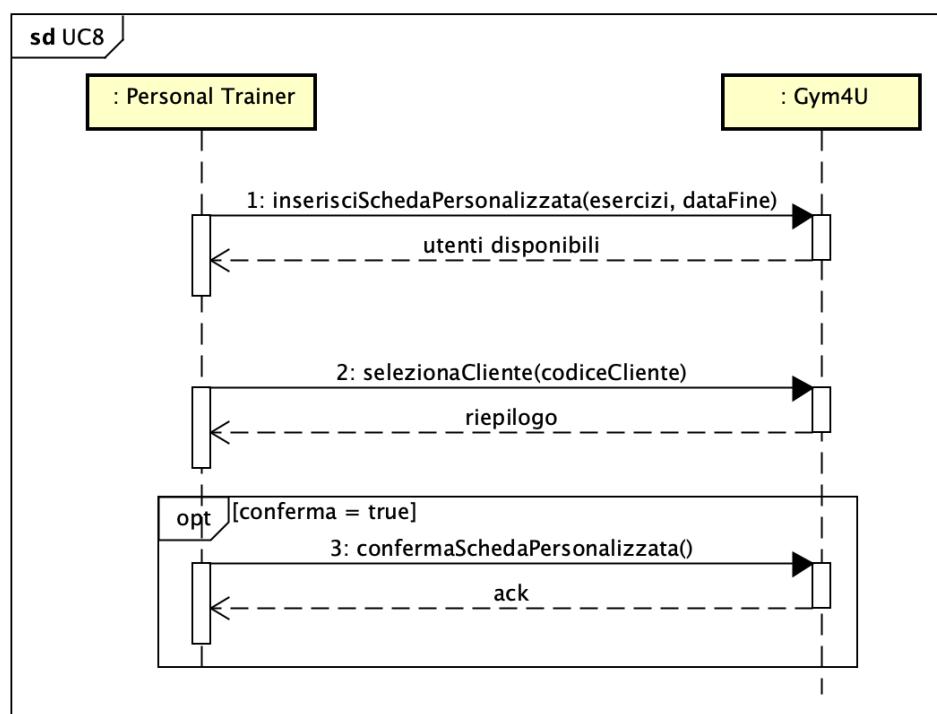
<b>Operazione</b>	confermaPresenza()
<b>Riferimenti</b>	Caso d'Uso UC10: Accesso in Palestra tramite Badge
<b>Pre-Condizioni</b>	Il Cliente ha prenotato una Lezione e vuole accedere in struttura
<b>Post-Condizioni</b>	Il campo validata della Prenotazione prenotazione è stato posto a true

→ Iterazione 3:

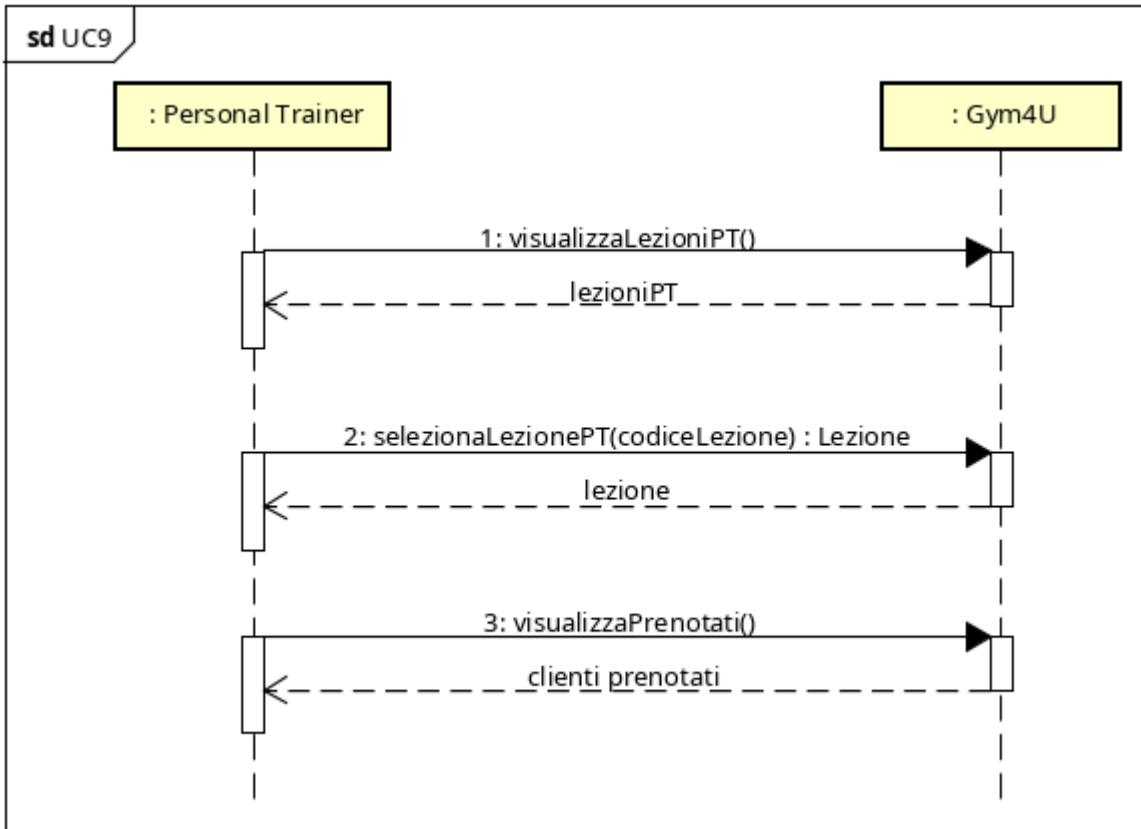
SSD Caso d'Uso UC7



SSD Caso d'Uso UC8



## SSD Caso d'Uso UC9



Contratti Operazioni UC7

Contratto CO1: *inserisciOfferta*

<b>Operazione</b>	<i>inserisciOfferta(scontoPercentuale: Float, dataInizio LocalDate, dataFine LocalDate)</i>
<b>Riferimenti</b>	Caso d'Uso UC7: Creazione di un'offerta promozionale
<b>Pre-Condizioni</b>	L'amministratore è autenticato in Gym4U
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente sconto di Sconto</li> <li>• sconto è stato inizializzato con le informazioni inserite</li> <li>• sconto è stato associato a Gym4U tramite l'associazione "corrente"</li> </ul>

Contratto CO1: *confermaOfferta*

<b>Operazione</b>	<i>confermaOfferta()</i>
<b>Riferimenti</b>	Caso d'Uso UC7: Creazione di un'offerta promozionale
<b>Pre-Condizioni</b>	Esiste un'istanza corrente di Sconto
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza sconto viene associata a Gym4U tramite l'associazione "gestisce"</li> <li>• È stata eliminata l'associazione "corrente" tra</li> </ul>

	sconto e Gym4U.
--	-----------------

Contratti Operazioni UC8

CO1: *inserisciSchedaPersonalizzata*

<b>Operazione</b>	<code>inserisciSchedaPersonalizzata(esercizi: []String, dataFineLocalDate)</code>
<b>Riferimenti</b>	Caso d'Uso UC8: Creazione di una scheda personalizzata
<b>Pre-Condizioni</b>	Il Personal Trainer pt è autenticato ed iscritto a Gym4U.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente sp di Scheda Personalizzata</li> <li>• sp è stato inizializzato con le informazioni inserite</li> <li>• sp è stato associato a pt tramite l'associazione "corrente"</li> </ul>

CO2: *selezionaCliente*

<b>Operazione</b>	<code>selezionaCliente(codiceCliente: Integer)</code>
<b>Riferimenti</b>	Caso d'Uso UC8: Creazione di una scheda personalizzata
<b>Pre-Condizioni</b>	Il Personal Trainer ha inserito correttamente una scheda personalizzata.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente c di Cliente</li> <li>• c è stato associato a Gym4U tramite l'associazione "corrente"</li> </ul>

CO3: *confermaSchedaPersonalizzata*

<b>Operazione</b>	<code>confermaSchedaPersonalizzata()</code>
<b>Riferimenti</b>	Caso d'Uso UC8: Creazione di una scheda personalizzata
<b>Pre-Condizioni</b>	Il Personal Trainer ha selezionato un Cliente.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza sp viene associata al Personal Trainer pt loggato tramite l'associazione "gestisce"</li> <li>• L'istanza sp viene associata all'istanza c di Cliente tramite l'associazione "relativa a"</li> <li>• È stata eliminata l'associazione "corrente" tra c e Gym4U</li> <li>• È stata eliminata l'associazione "corrente" tra sp e pt</li> </ul>

Contratti Operazioni UC9

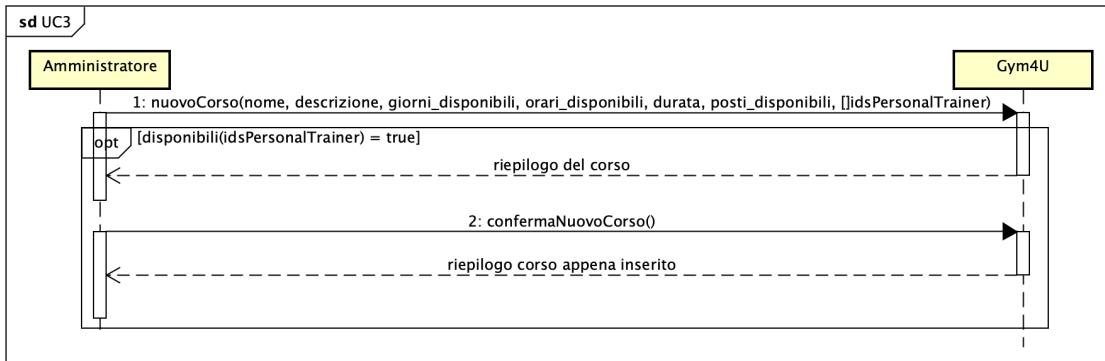
Contratto CO1: *selezionaLezione*

<b>Operazione</b>	selezionaLezione(codiceLezione: Integer)
<b>Riferimenti</b>	Caso d'Uso UC9: Visualizza prenotati di una lezione
<b>Pre-Condizioni</b>	Esiste una lista di lezioni associate ad un Personal Trainer
<b>Post-Condizioni</b>	È stata selezionata l'istanza I di Lezione.

→ Iterazione 4:

SSD Caso d'Uso UC3

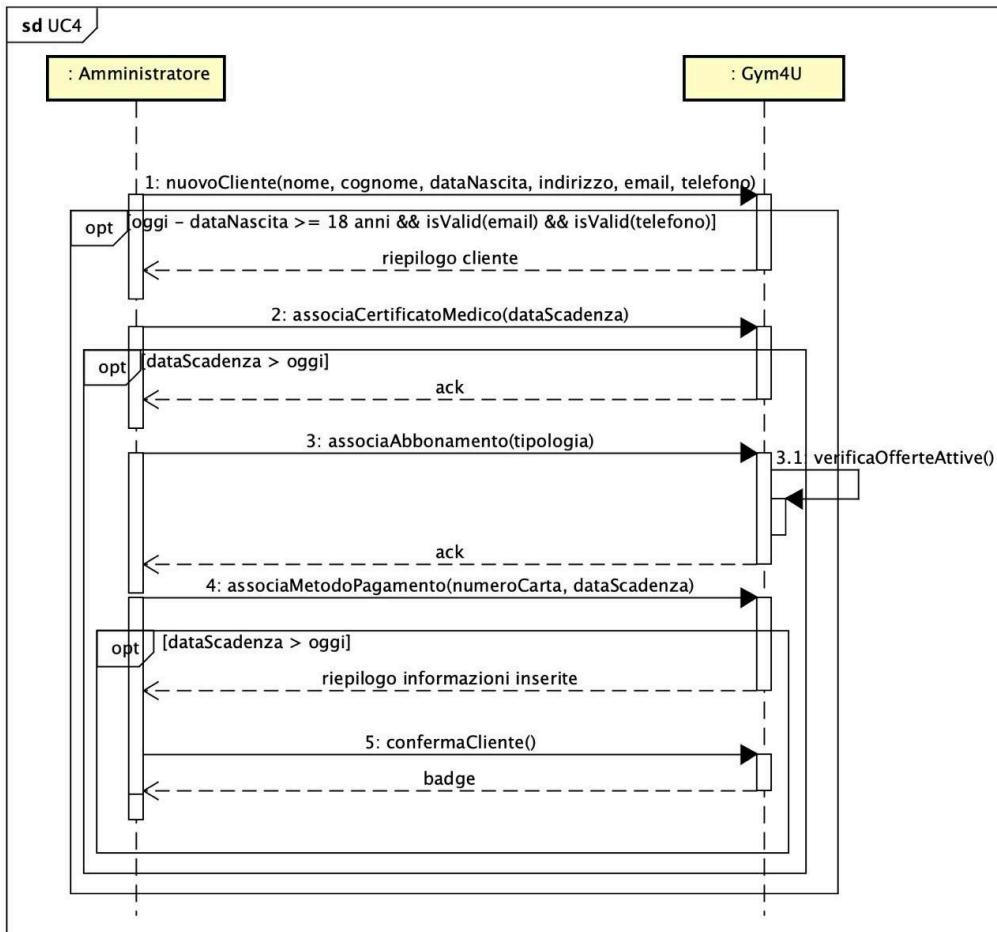
Abbiamo implementato lo scenario alternativo considerando la situazione in cui l'amministratore inserisca tra i Personal Trainer a cui associare il nuovo Corso uno che ha già in gestione 5 Corsi, andando contro alla Regola di Business R9.



SSD Caso d'Uso UC4

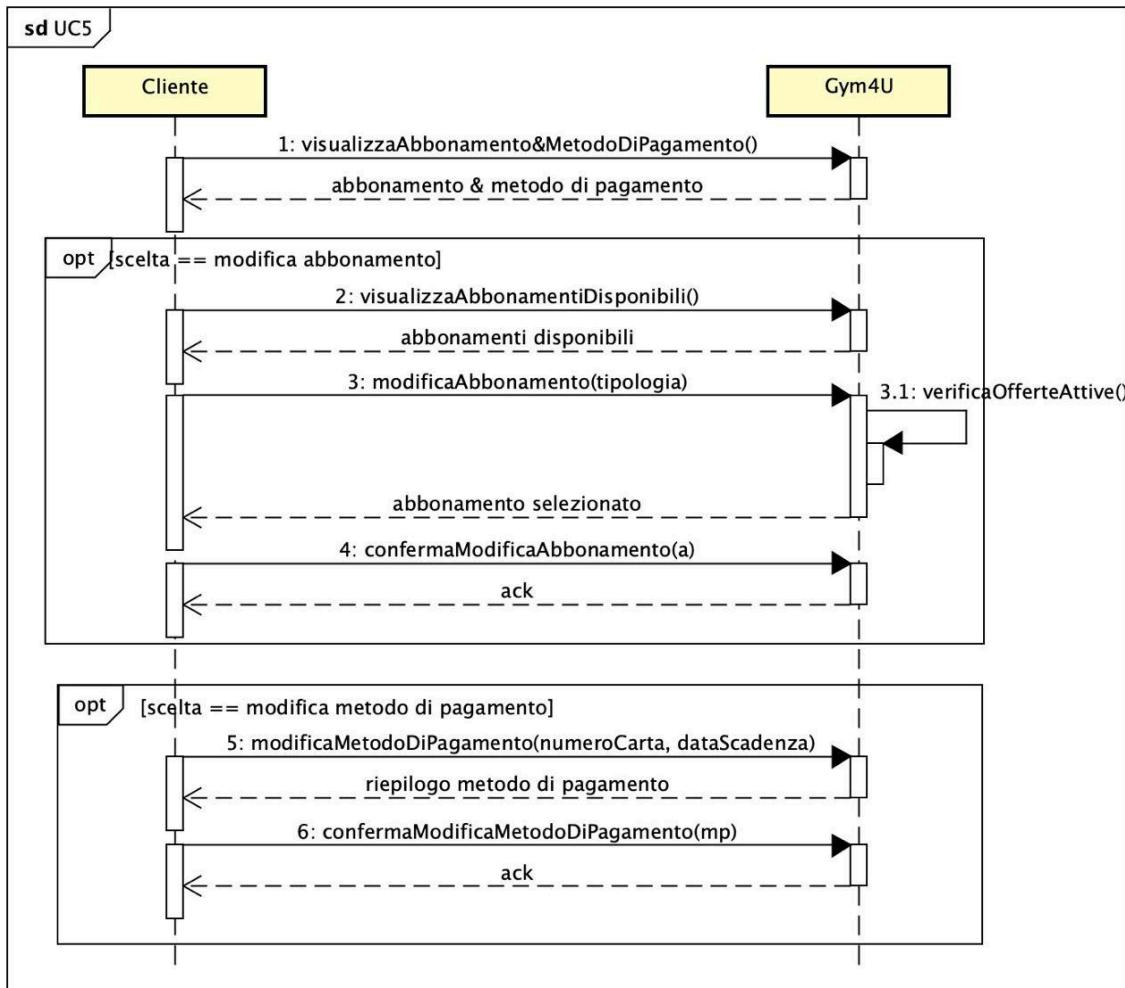
Implementazione modifiche dovute all'UC7

Dopo aver selezionato la tipologia di abbonamento per il Cliente, il sistema verifica la presenza di offerte attive al momento dell'iscrizione e, se presenti, le applica all'abbonamento selezionato.



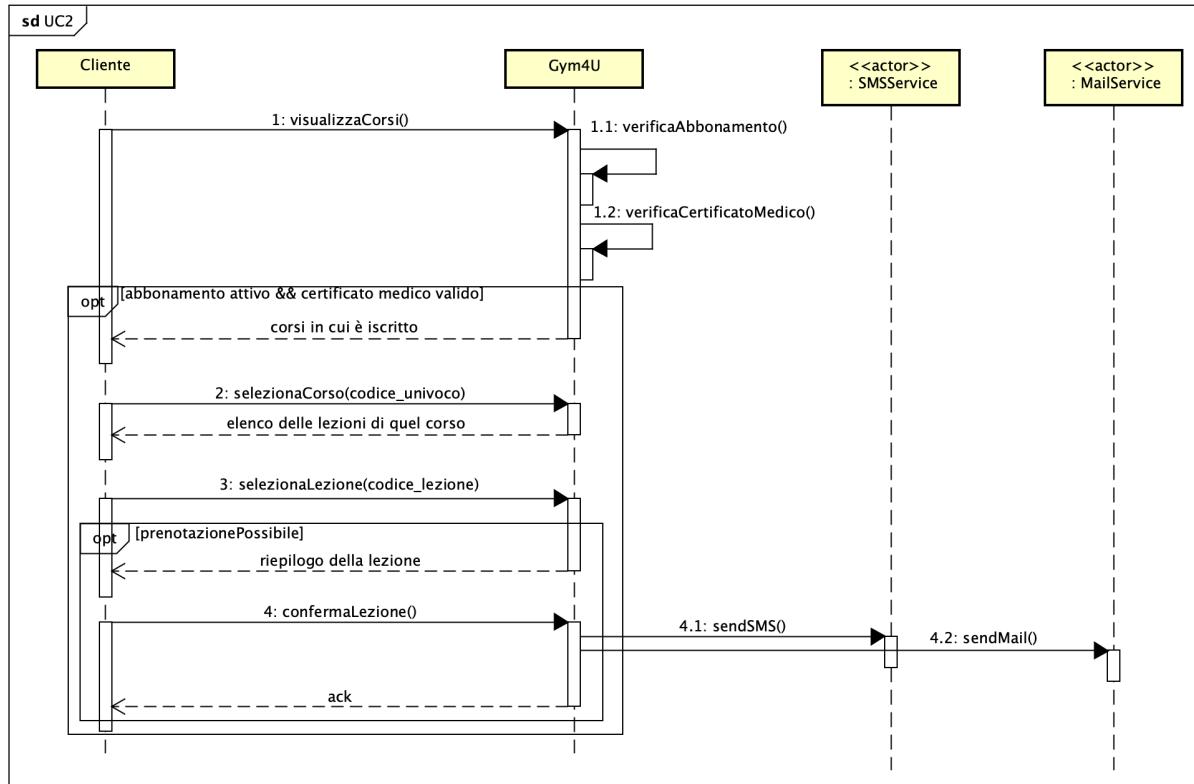
## SSD Caso d'Uso UC5

Dopo aver selezionato la tipologia di abbonamento con cui modificare quello attuale, il sistema verifica la presenza di offerte attive al momento della modifica e, se presenti, le applica all'abbonamento selezionato.



## SSD Caso d'Uso UC2

Per l'implementazione del passo 11 dell'UC2 "Il Sistema invia una conferma della prenotazione all'utente via email" abbiamo optato per l'impiego del Pattern GoF **Observer**.



Le modifiche apportate al Modello di Dominio si riflette sui seguenti contratti delle operazioni:

### Contratti Operazioni UC5

#### CO2: confermaModificaAbbonamento

<b>Operazione</b>	confermaAbbonamento()
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	Esiste un'istanza a di Abbonamento
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>L'istanza a di Abbonamento viene associata al Cliente c tramite l'associazione "possiede"</li> <li>È stata eliminata l'associazione "corrente" tra a e c</li> </ul>

CO4: *confermaModificaMetodoDiPagamento*

<b>Operazione</b>	confermaMetodoDiPagamento()
<b>Riferimenti</b>	Caso d'Uso UC5: Gestione Abbonamento
<b>Pre-Condizioni</b>	Esiste una lista di Personal Trainer personalTrainers
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• L'istanza mp di MetodoDiPagamento viene associata al Cliente c tramite l'associazione "possiede"</li> <li>• È stata eliminata l'associazione "corrente" tra mp e c</li> </ul>

Contratti Operazioni UC6

Contratto CO2: *confermaPrenotazione*

<b>Operazione</b>	confermaPrenotazione()
<b>Riferimenti</b>	Caso d'Uso UC6: Prenotazione Personal Trainer
<b>Pre-Condizioni</b>	<ul style="list-style-type: none"> <li>• È in corso la prenotazione di una Lezione Personal Trainer</li> <li>• Esiste un'istanza l di LezionePT</li> </ul>
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza p di Prenotazione</li> <li>• L'istanza p è stata associata alla Lezione l</li> <li>• L'istanza p è stata associata al Cliente cliente</li> <li>• L'istanza p viene associata a Gym4U tramite l'associazione "tiene traccia di"</li> <li>• È stata eliminata l'associazione "corrente" tra l e pt</li> <li>• L'istanza l è stato associata a pt tramite l'associazione "gestisce"</li> </ul>

Contratti Operazioni UC8

CO1: *inserisciSchedaPersonalizzata*

<b>Operazione</b>	inserisciSchedaPersonalizzata(esercizi: []String, dataFine LocalDate)
<b>Riferimenti</b>	Caso d'Uso UC8: Creazione di una scheda personalizzata
<b>Pre-Condizioni</b>	Il Personal Trainer pt è autenticato ed iscritto a Gym4U.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>• È stata creata un'istanza corrente sp di Scheda Personalizzata</li> <li>• sp è stato inizializzato con le informazioni inserite</li> <li>• sp è stato associato a pt tramite l'associazione "corrente"</li> </ul>

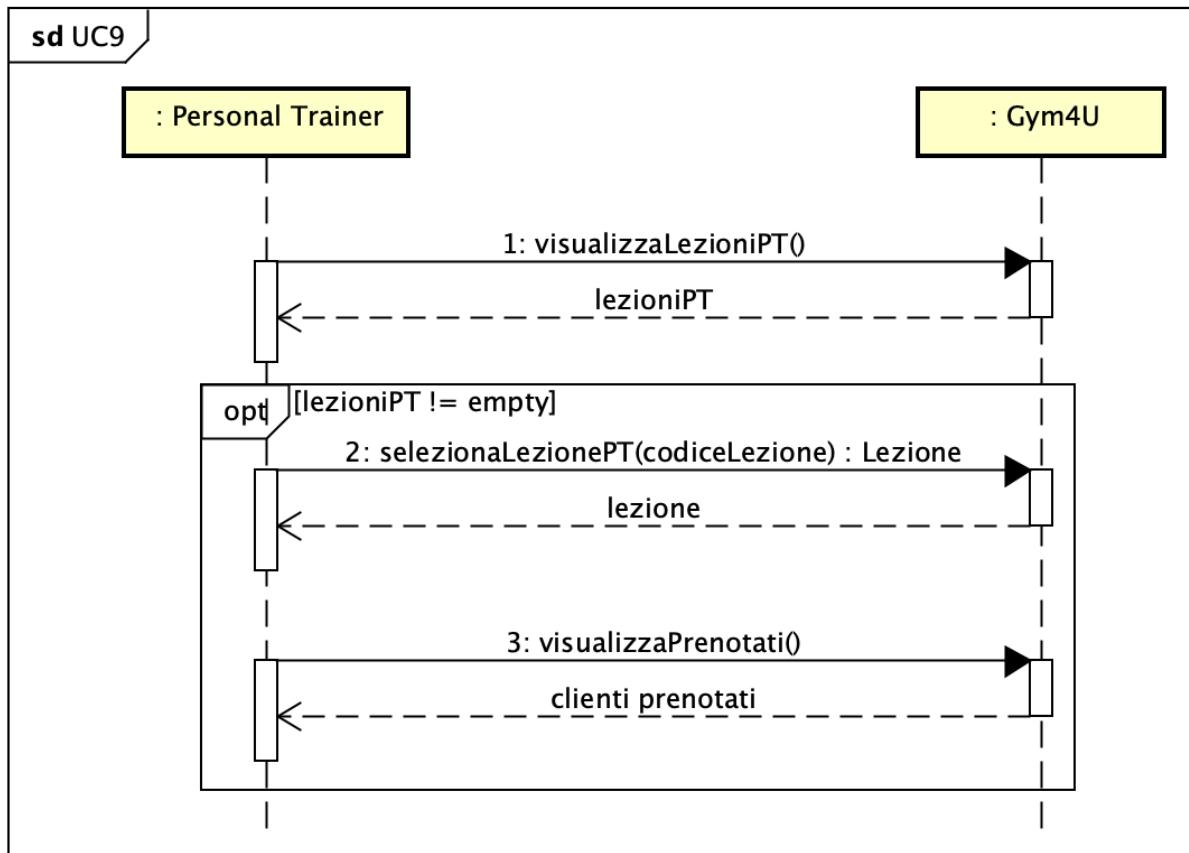
CO3: confermaSchedaPersonalizzata

<b>Operazione</b>	confermaSchedaPersonalizzata()
<b>Riferimenti</b>	Caso d'Uso UC8: Creazione di una scheda personalizzata
<b>Pre-Condizioni</b>	Il Personal Trainer ha selezionato un Cliente.
<b>Post-Condizioni</b>	<ul style="list-style-type: none"> <li>L'istanza sp viene associata al Personal Trainer pt loggato tramite l'associazione "gestisce"</li> <li>L'istanza sp viene associata all'istanza c di Cliente tramite l'associazione "relativa a"</li> <li>È stata eliminata l'associazione "corrente" tra c e Gym4U</li> <li>È stata eliminata l'associazione "corrente" tra sp e pt</li> </ul>

→ Documentazione Completa:

SSD Caso d'Uso UC9

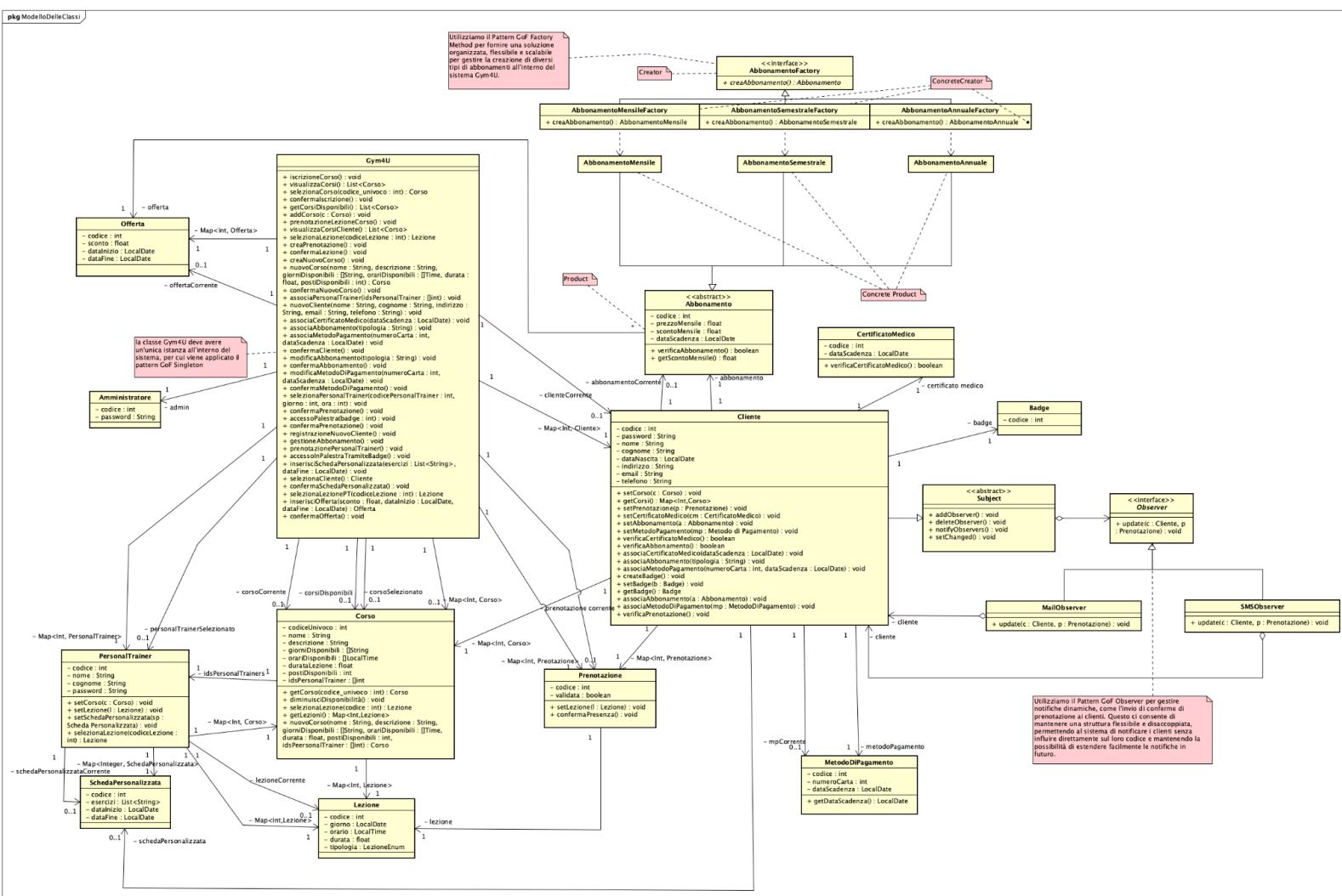
Viene previsto lo scenario alternativo 2.a



# 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 deciso di utilizzare il Pattern GoF **Factory Method** nel caso degli abbonamenti per le seguenti ragioni:

- Flessibilità nell'aggiunta di nuovi tipi di abbonamenti: Il pattern Factory Method consente di aggiungere nuovi tipi di abbonamenti senza dover modificare il codice esistente. Se, in futuro, si desidera introdurre un nuovo tipo di abbonamento (ad esempio, un abbonamento trimestrale), è sufficiente creare una nuova classe concreta e la rispettiva factory senza influenzare le altre parti del sistema.

- Organizzazione del codice: Il pattern Factory Method aiuta a organizzare il codice in modo modulare. Ogni factory è responsabile della creazione di un tipo specifico di oggetto, contribuendo a mantenere una struttura chiara e comprensibile.
- Scalabilità: Nel caso in cui la logica di creazione degli abbonamenti possa diventare più complessa nel tempo (ad esempio, con regole specifiche per la creazione di determinati tipi di abbonamenti), il pattern Factory Method offre uno spazio per gestire questa complessità in modo separato.

Per l'implementazione del passo 11 dell'UC2 “Il Sistema invia una conferma della prenotazione all'utente via email” abbiamo optato per l'impiego del Pattern GoF **Observer**. Questo ci consente di gestire dinamicamente l'invio di conferme di prenotazione ai clienti. I sistemi di notifica (email e SMS) agiscono come Observer, mentre il Cliente funge da Concrete Subject. Questa scelta architettonale ci offre una soluzione flessibile e scalabile, permettendo un'espansione agevole delle modalità di notifica senza dover modificare direttamente il codice del Cliente.

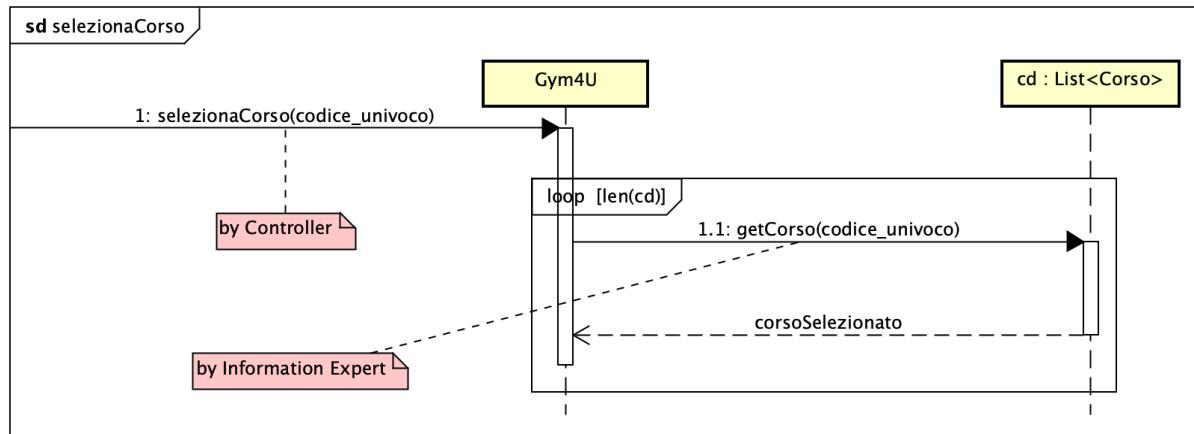
Questo pattern offre un'ottima separazione tra la logica di business della conferma della prenotazione e le diverse modalità di notifica. Se in futuro si desiderasse aggiungere o rimuovere un canale di notifica, sarà possibile farlo senza dover modificare il codice del Cliente o della conferma della prenotazione stessa. Inoltre, il Pattern Observer si presta bene a garantire una bassa dipendenza tra il soggetto e gli osservatori, contribuendo così a mantenere un sistema modularizzato ed estensibile.

# Diagramma di sequenza

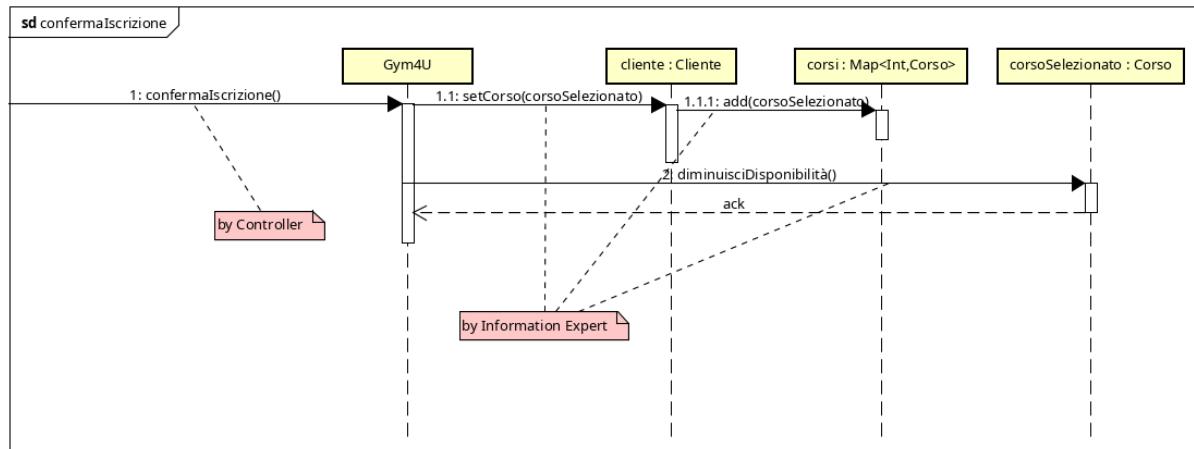
→ Iterazione 1:

UC1

- selezionaCorso

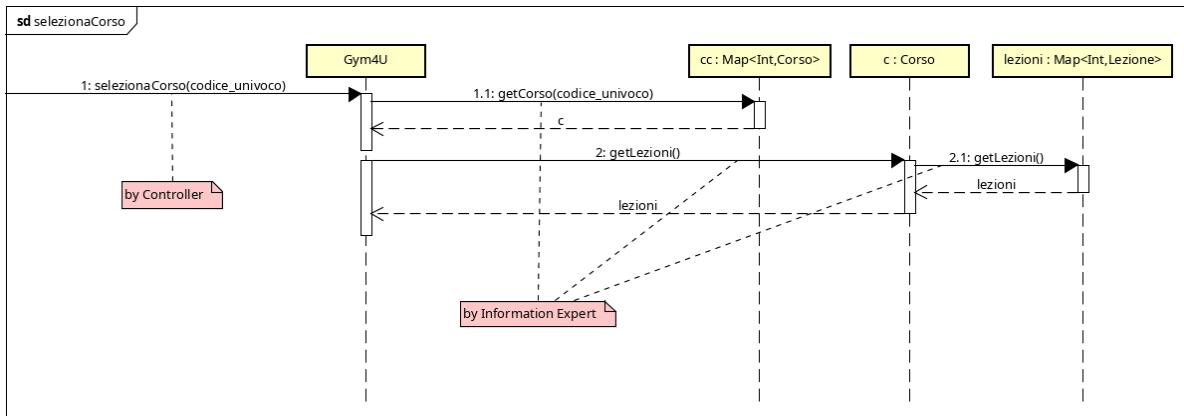


- confermalscrizione

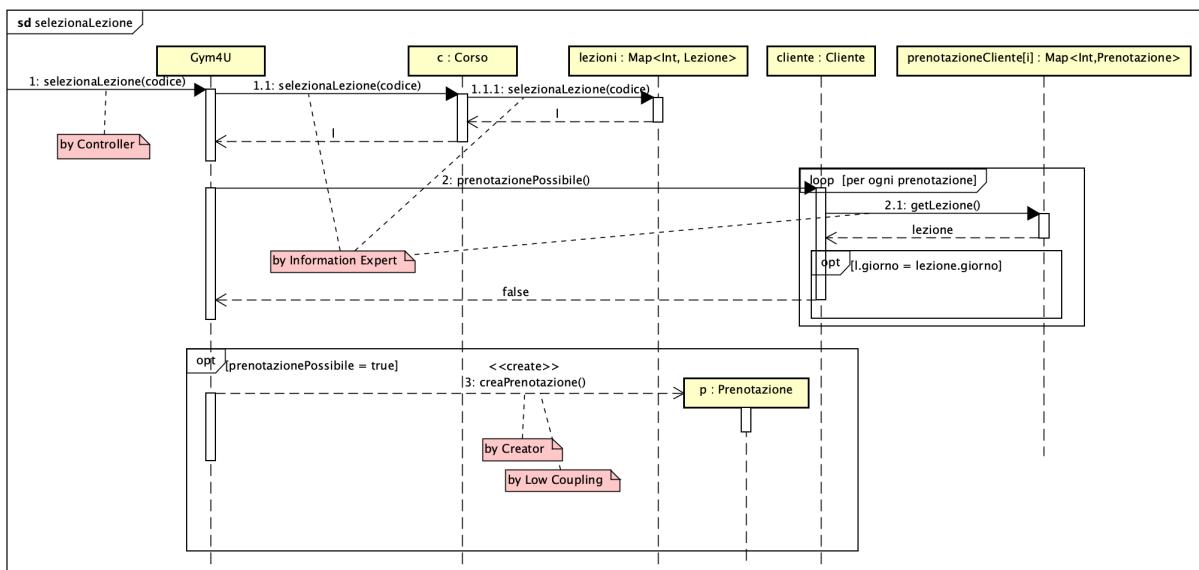


## UC2

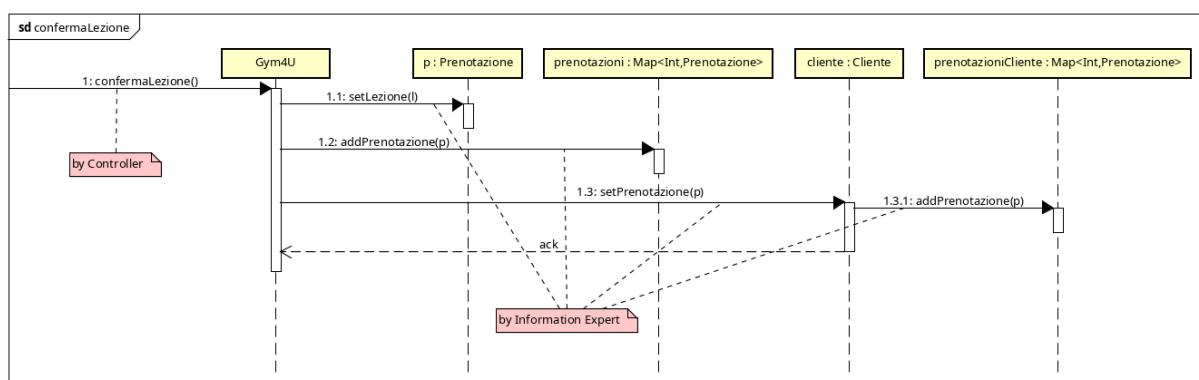
- selezionaCorso



- selezionaLezione

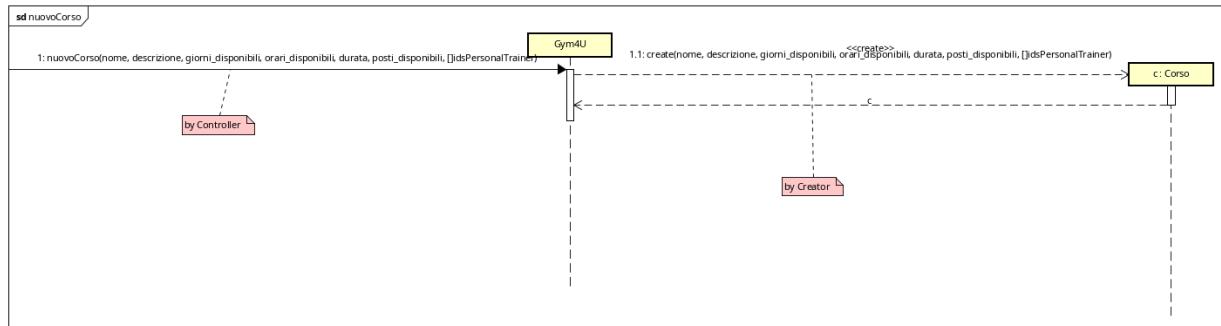


- confermaLezione

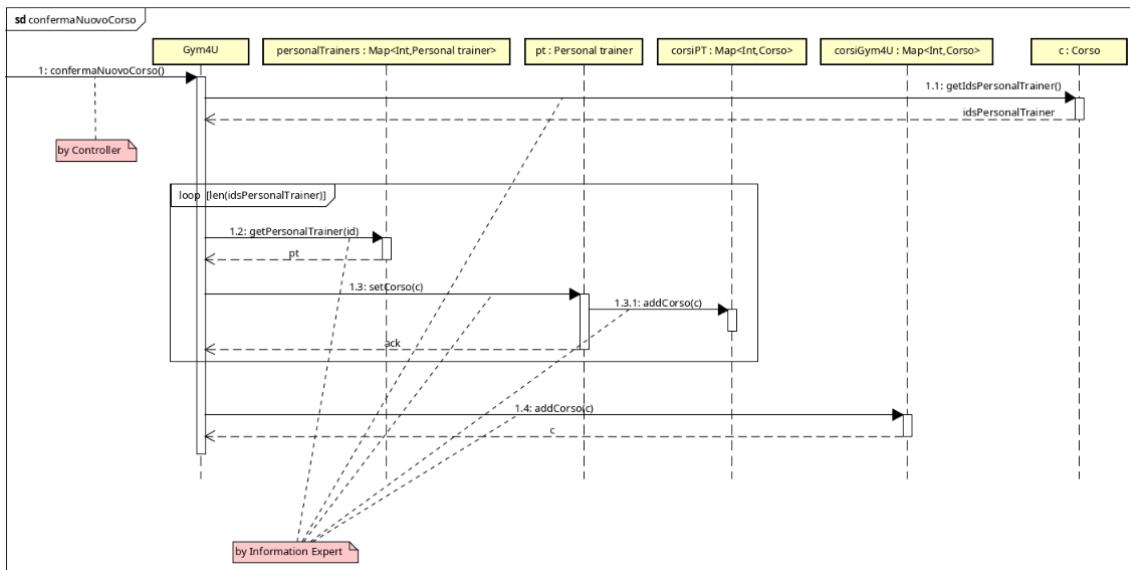


## UC3

- nuovoCorso

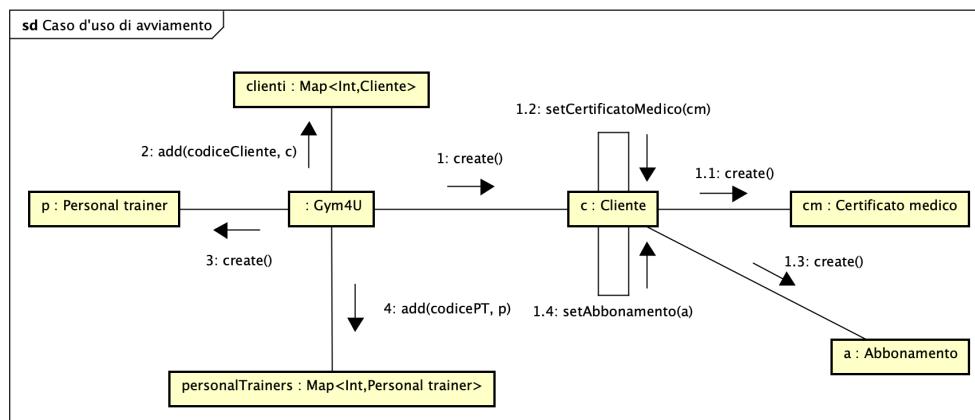


- confermaNuovoCorso



## Caso d'Uso di Avviamento

Al fine di rendere utilizzabile e funzionante il sistema durante la prima iterazione, è stato ideato un caso d'uso di avviamento per creare degli utenti aventi un abbonamento attivo ed un certificato medico valido.

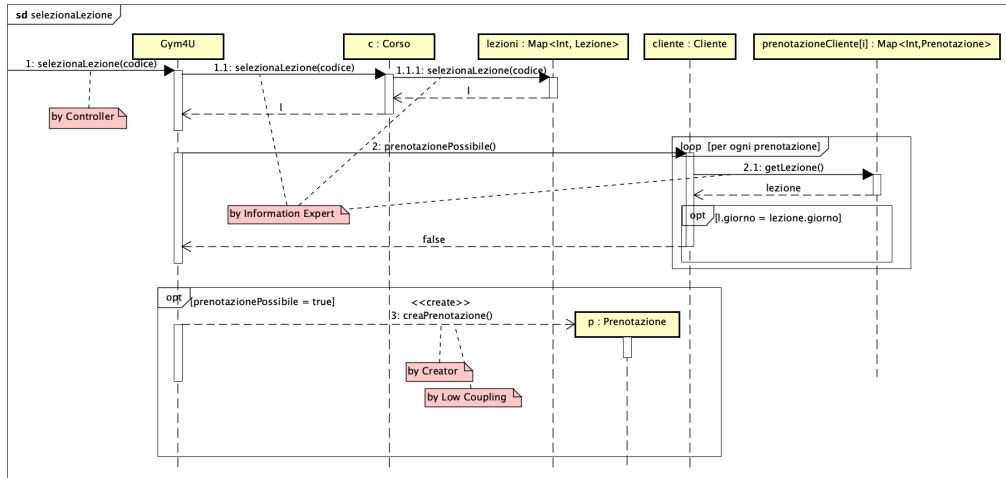


→ Iterazione 2:

UC2

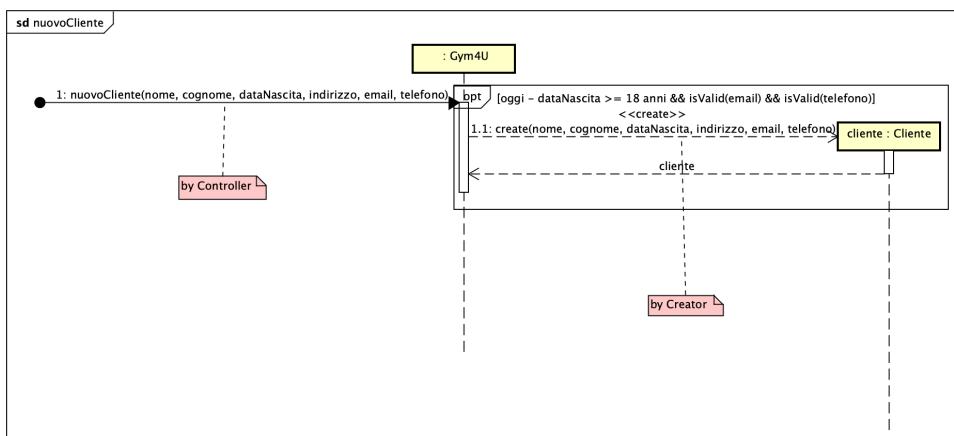
- selezionaLezione

Modifica relativa allo scenario alternativo 7.a: viene fatta valere la Regola di Business R5

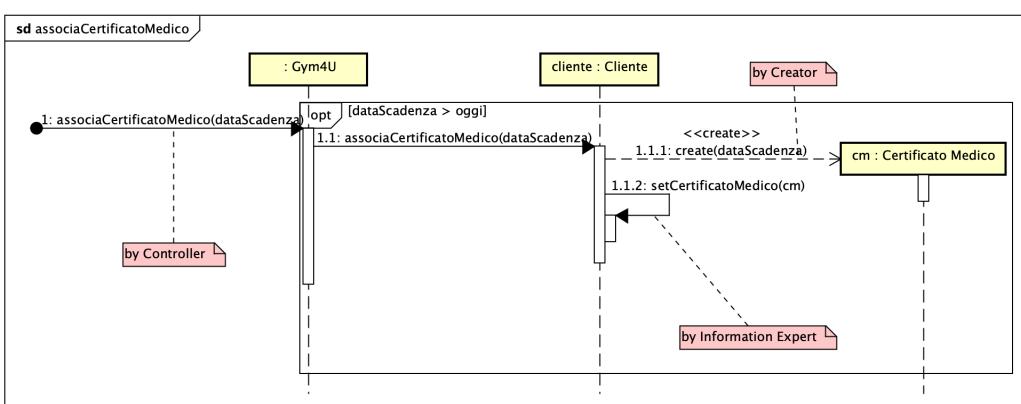


UC4

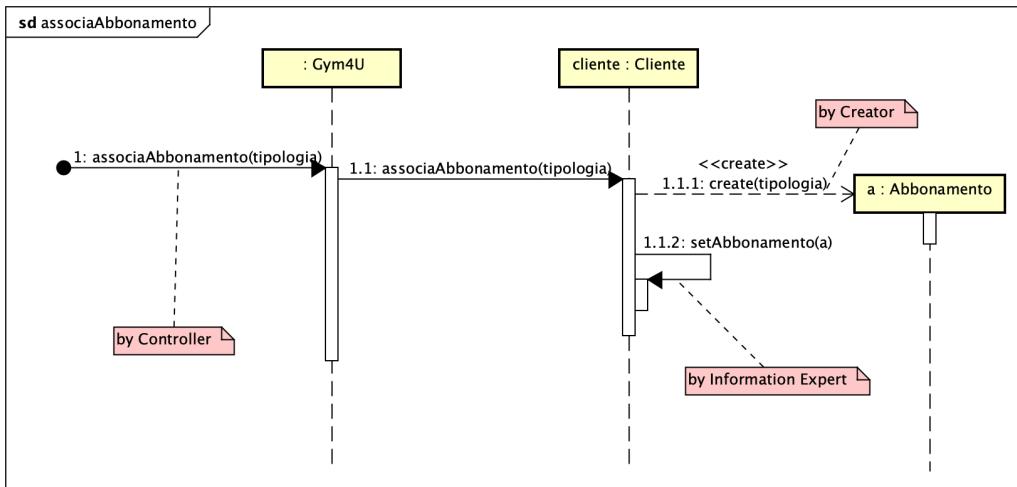
- nuovoCliente



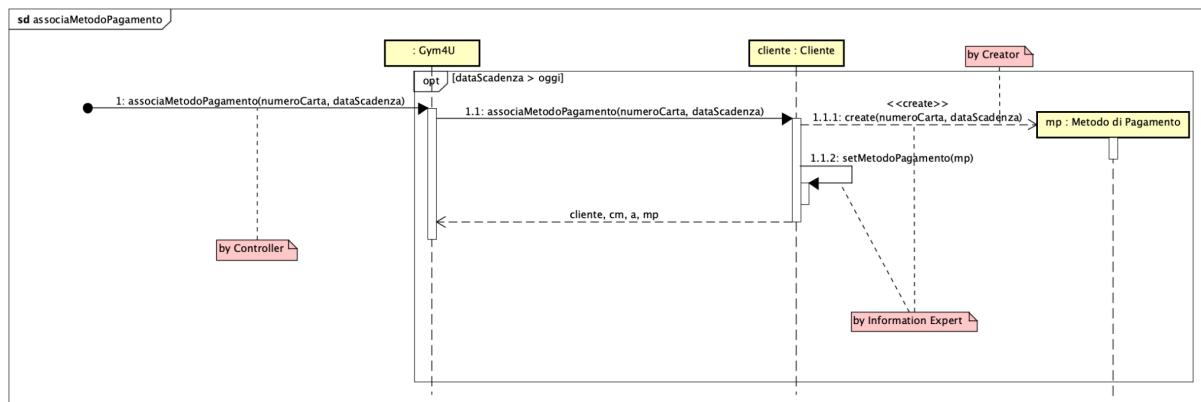
- associaCertificatoMedico



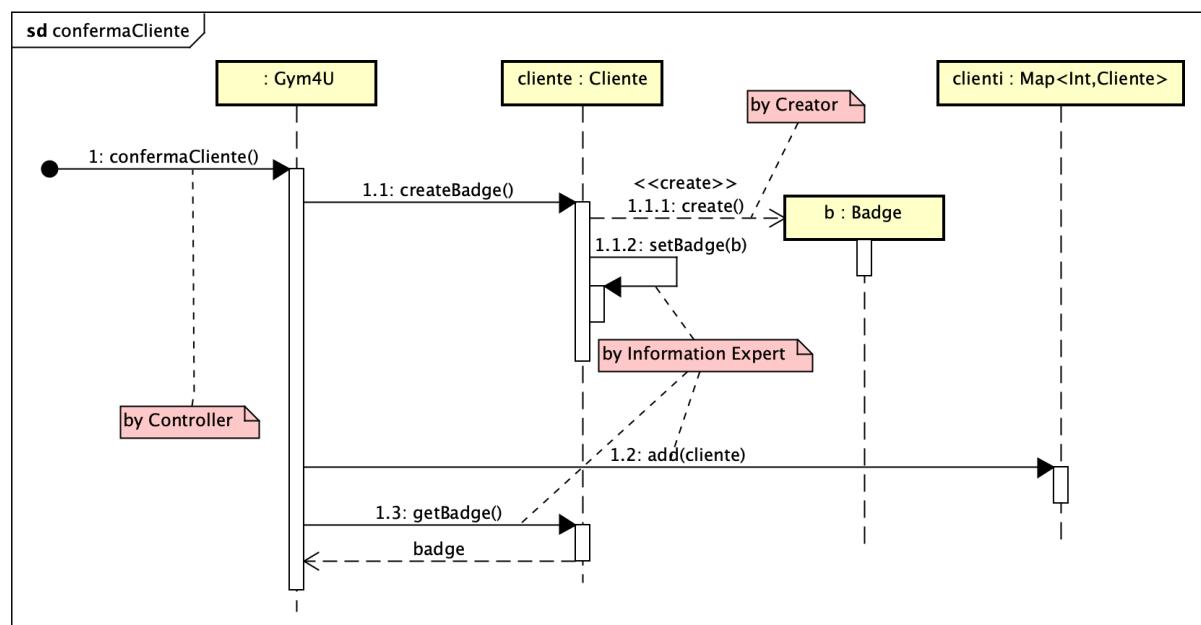
- associaAbbonamento



- associaMetodoPagamento

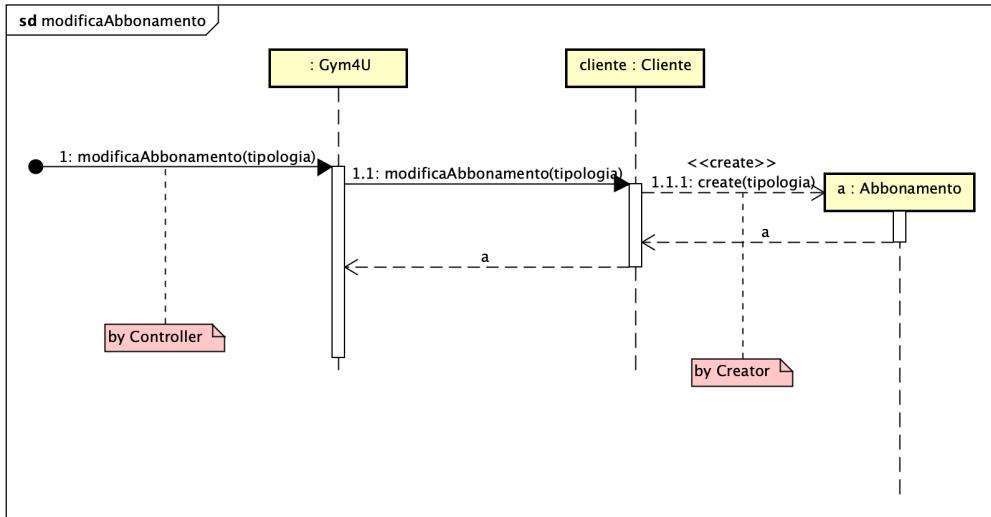


- confermaCliente

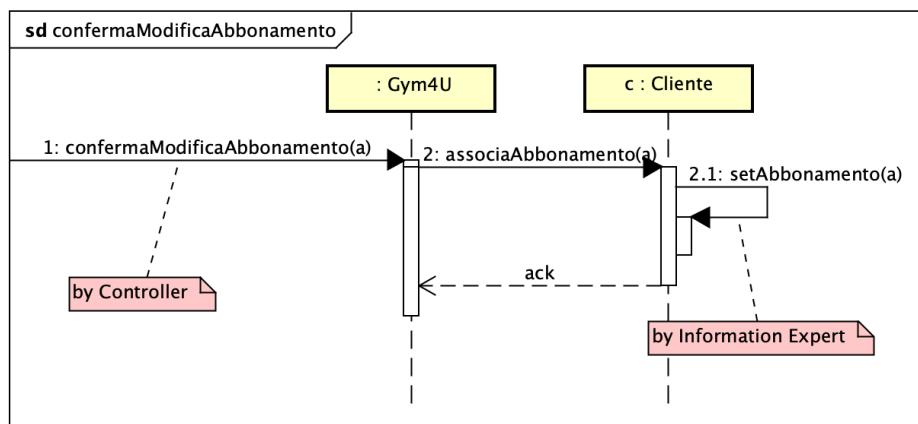


## UC5

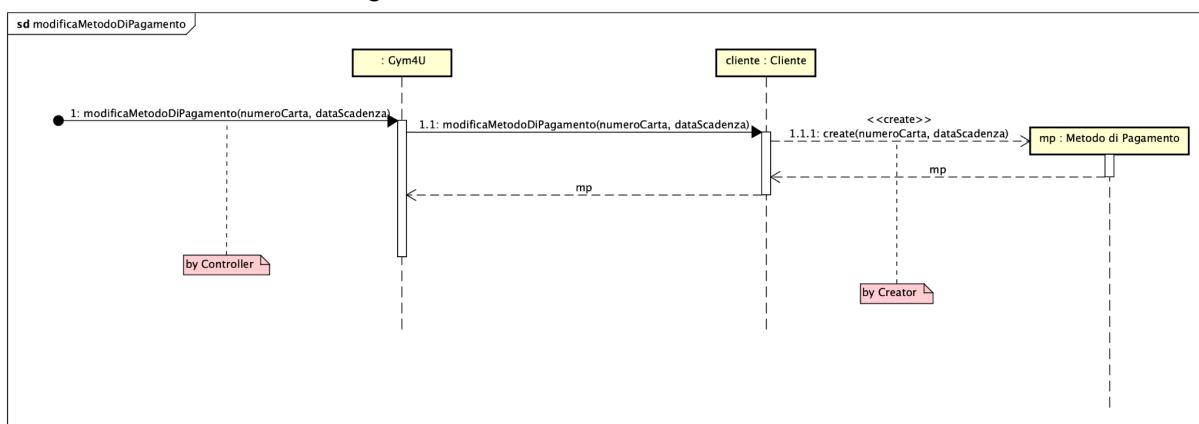
- modificaAbbonamento



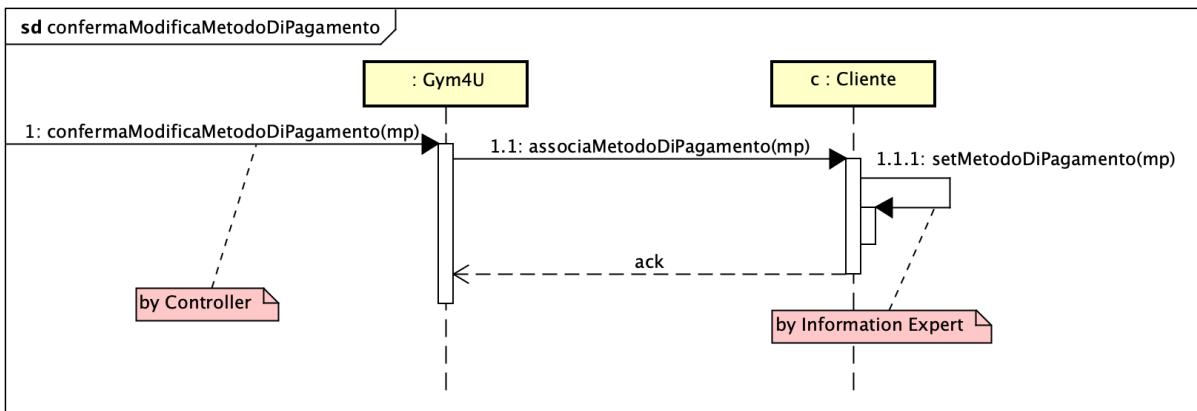
- confermaModificaAbbonamento



- modificaMetodoDiPagamento

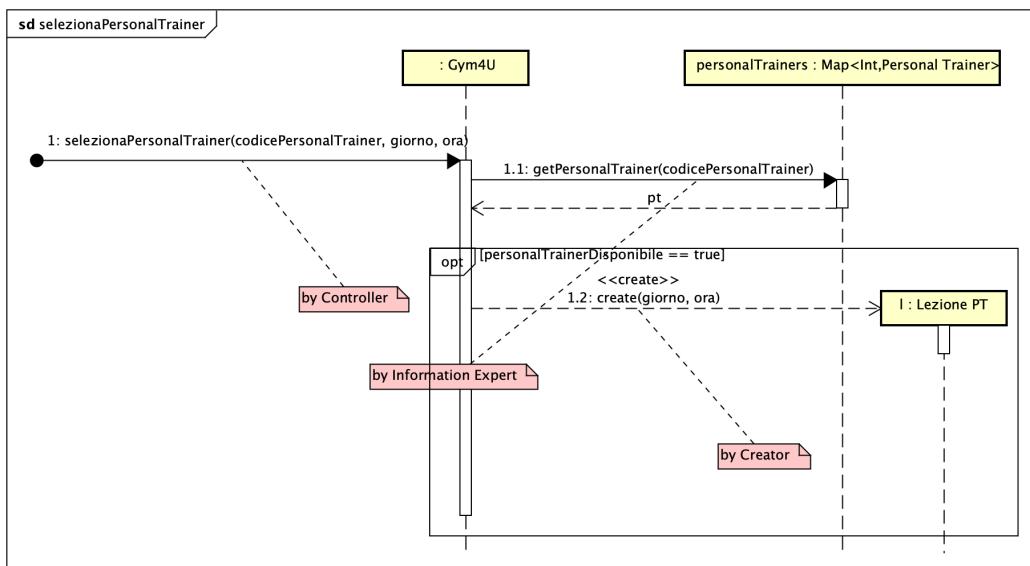


- confermaModificaMetodoDiPagamento

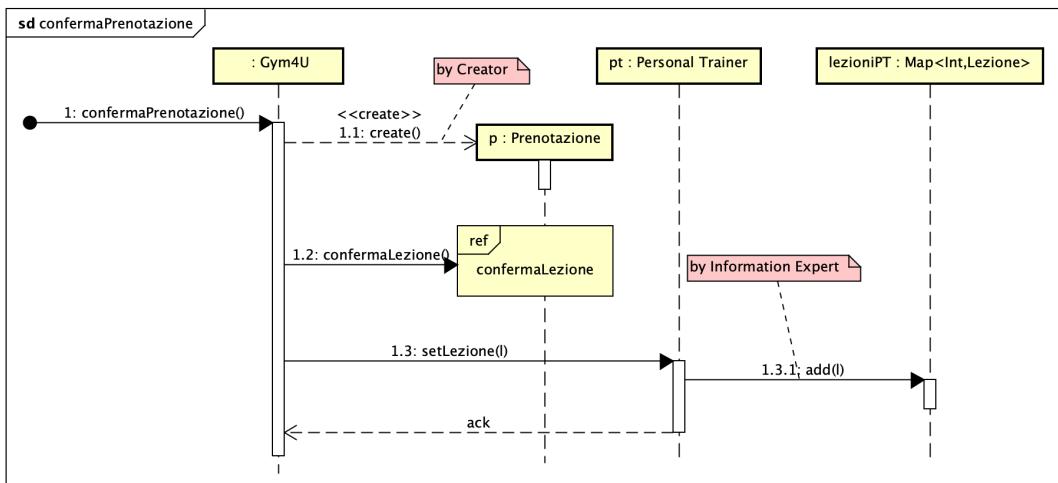


UC6

- selezionaPersonalTrainer

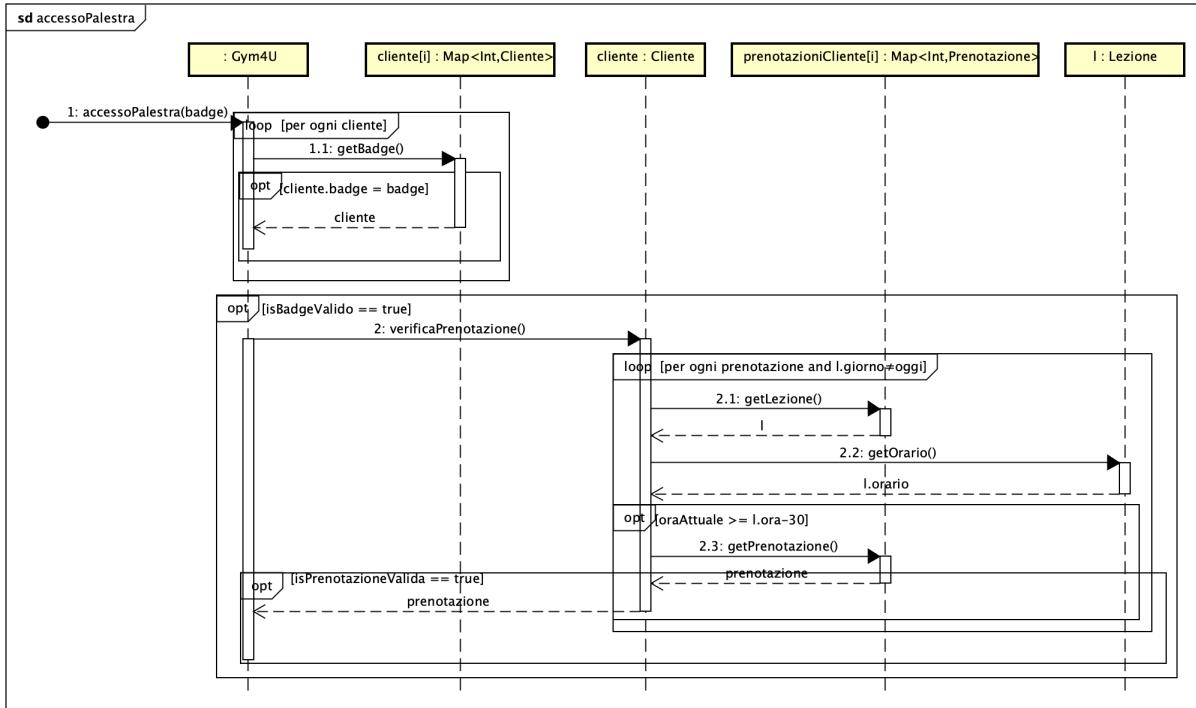


- confermaPrenotazione

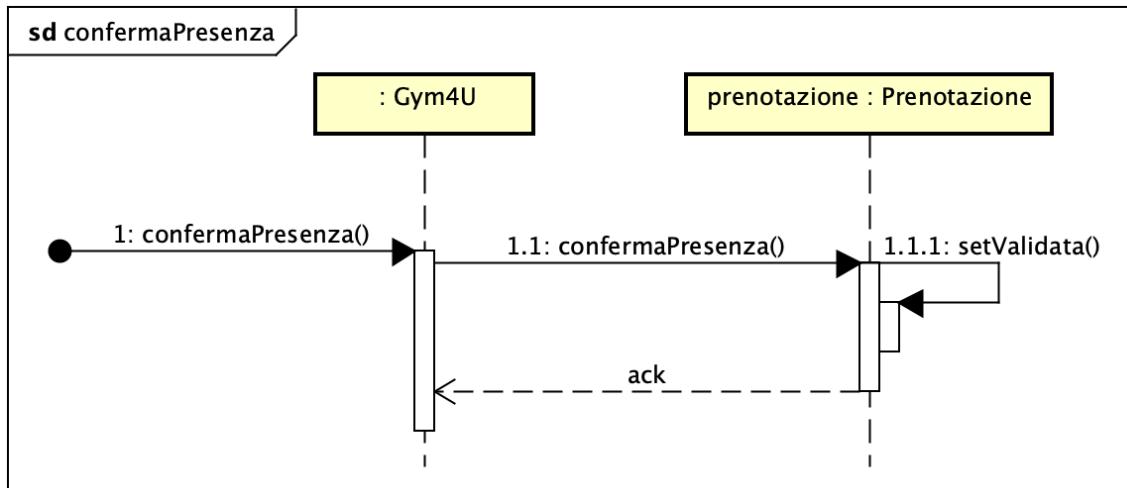


## UC10

- **accessoPalestra**



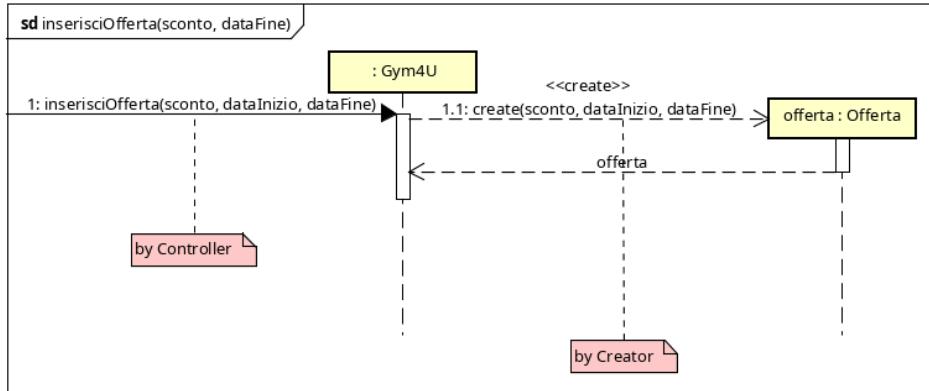
- **confermaPresenza**



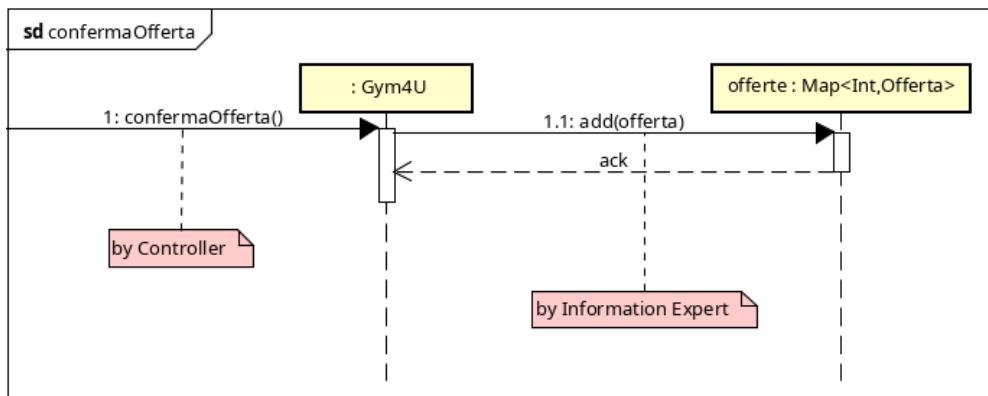
→ Iterazione 3:

UC7

- inserisciOfferta

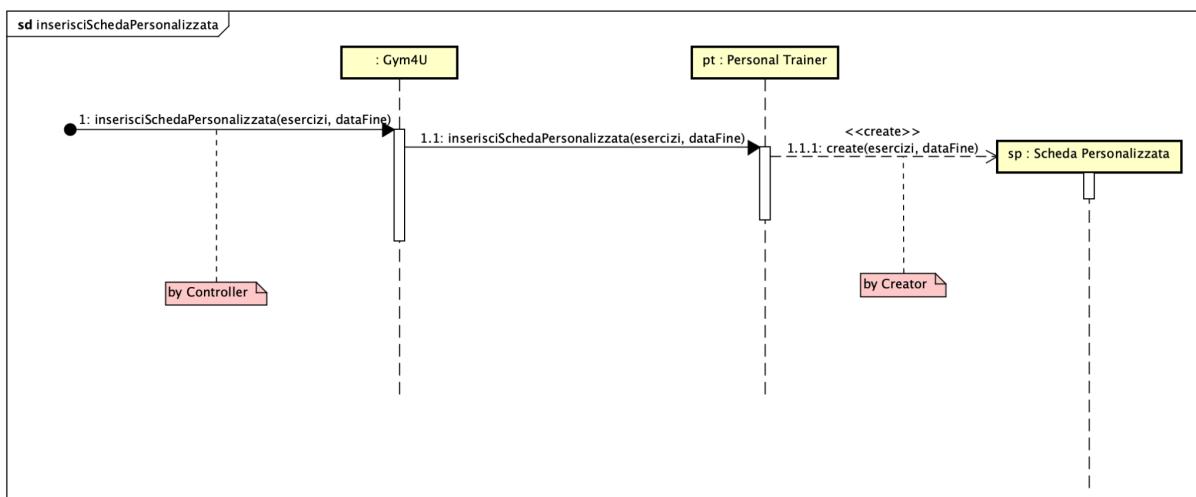


- confermaOfferta

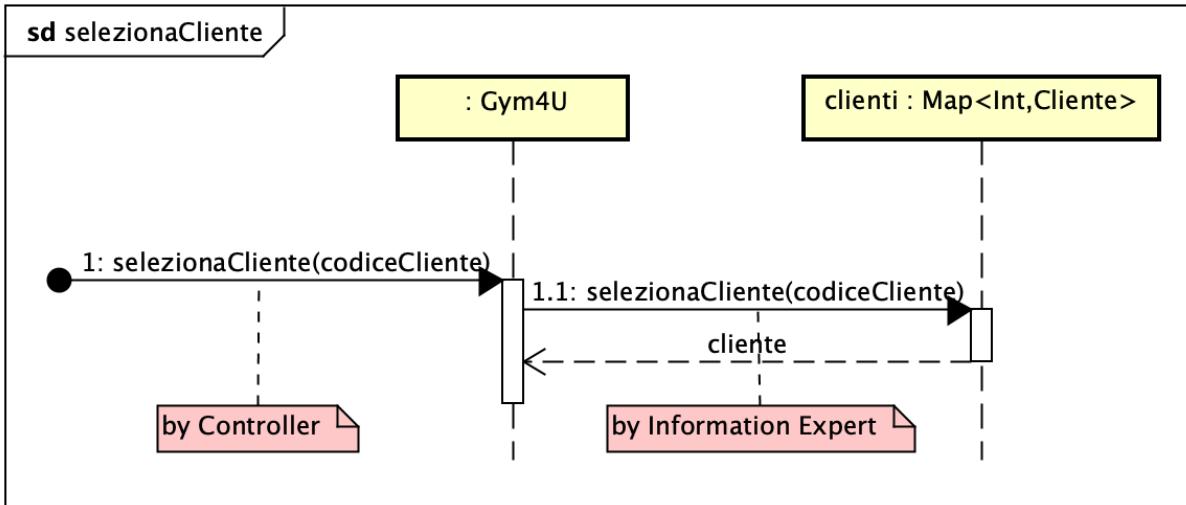


UC8

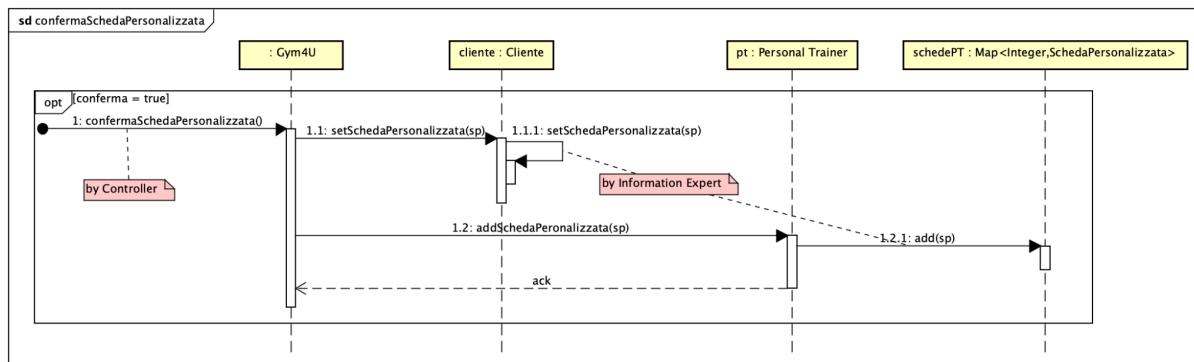
- inserisciSchedaPersonalizzata



- selezionaCliente

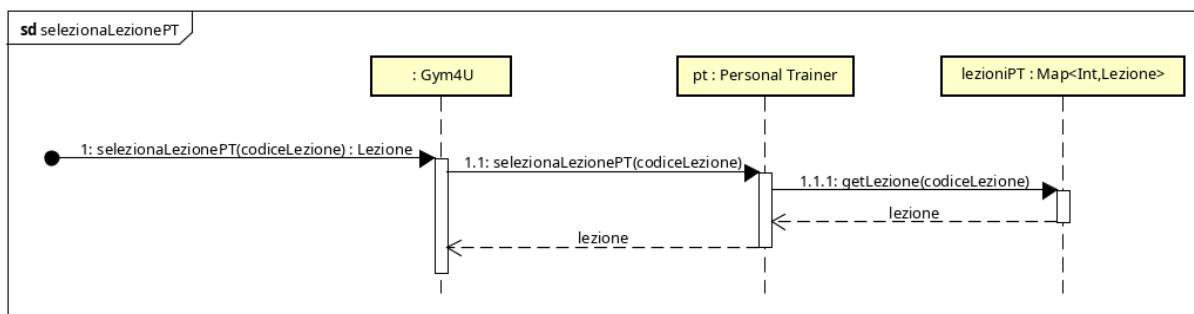


- confermaSchedaPersonalizzata



UC9

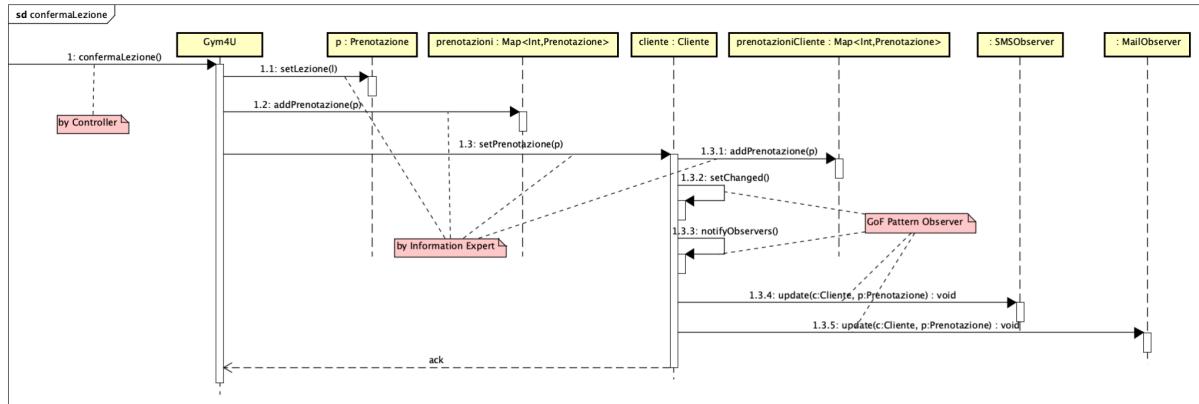
- selezionaLezione



## → Iterazione 4:

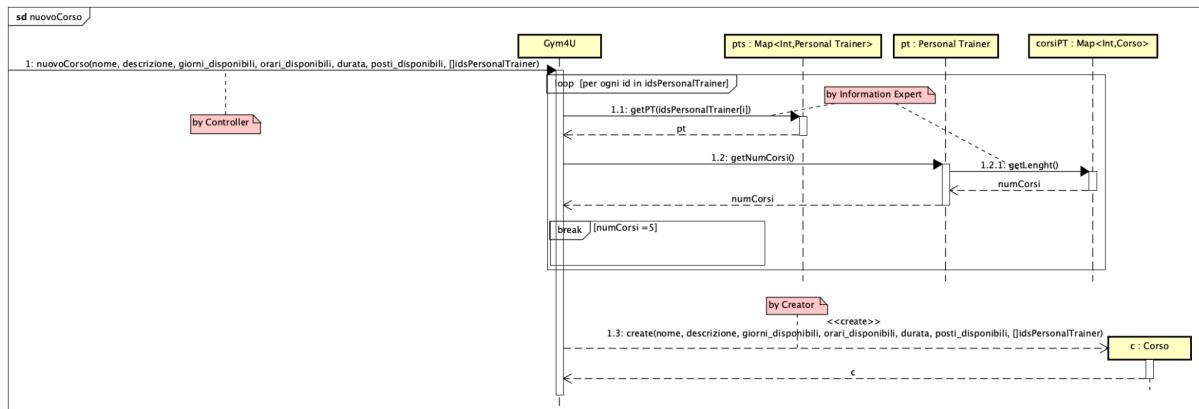
### UC2

- confermaLezione



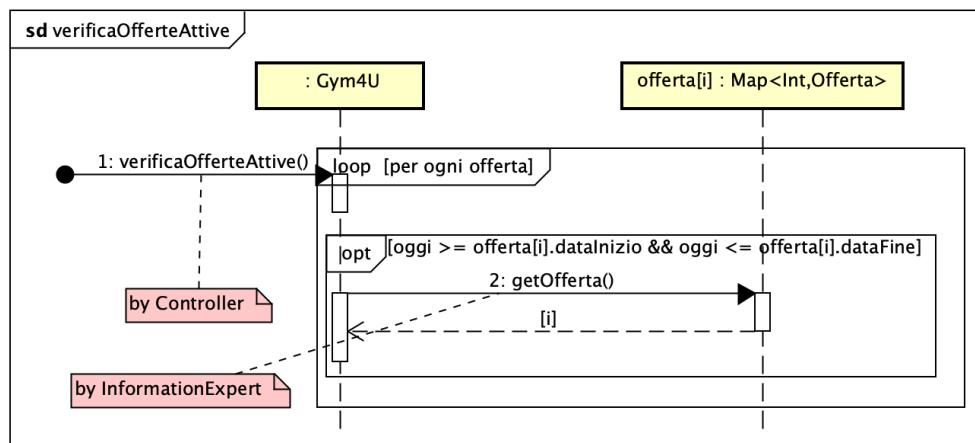
### UC3

- nuovoCorso



### UC4 e UC5

- verificaOfferteAttive



# Testing

## Introduzione

Il processo di testing riveste un ruolo cruciale nello sviluppo di un software robusto ed efficiente. Oltre a contribuire significativamente alla riduzione dei costi di manutenzione, il testing svolge un ruolo fondamentale nell'assicurare che l'applicazione soddisfi gli standard di funzionalità richiesti.

La probabilità di individuare eventuali malfunzionamenti è direttamente proporzionale al numero di test eseguiti. Tuttavia, è da notare che testare un programma in modo esaustivo è spesso un compito complesso, poiché le possibili combinazioni di input sono immense e non sempre riproducibili in un lasso di tempo ragionevole. Nonostante questa sfida, un processo di testing ben progettato può rilevare comportamenti anomali e indesiderati, contribuendo a garantire che il software sviluppato funzioni in modo corretto e conforme alle specifiche del committente.

I test possono essere suddivisi principalmente in due categorie:

**Test Unitari (Unit Test):** Questi test mirano a verificare la correttezza del codice in ogni sua parte, valutando singolarmente ogni metodo rispetto ai valori attesi. In Java, l'utilizzo di framework come JUnit è comune per l'automazione dei test di unità.

**Test Funzionali:** Questi test valutano il corretto funzionamento del sistema nella sua completezza. Trattando il sistema come una "scatola nera", ricevono input e verificano la correttezza degli output.

Nell'ambito dell'applicazione sviluppata, si è adottato un approccio concentrato principalmente sui test unitari, eseguiti tramite il framework JUnit. Questo approccio Bottom-Up consente di collaudare inizialmente le singole unità che, una volta integrate, costituiranno il programma completo.

## Individuazione dei casi di test e testing unitario

I test mirano a verificare la correttezza del codice, valutando ogni singolo metodo in base ai risultati attesi.

Ci siamo concentrati principalmente sui test unitari eseguiti attraverso JUnit, seguendo un approccio Bottom-Up. Questo ci ha permesso di collaudare inizialmente le singole unità che, integrate, costituiranno il programma completo, garantendo un controllo accurato e dettagliato della correttezza del codice a ogni livello di sviluppo.

→ Iterazione 1:

Abbiamo progettato e implementato diversi casi di test per valutare la corretta funzionalità dei casi d'uso implementati (UC1-UC2-UC3) in Gym4U.

Di seguito, illustriamo le strategie adottate per l'identificazione di ciascun test.

TestInit():

- Scopo: Verificare che l'inizializzazione del sistema Gym4U avvenga correttamente.
- Realizzazione: Inizializziamo un'istanza di Gym4U utilizzando il metodo getInstance() e verifichiamo che l'oggetto gym4u non sia nullo.

#### TestCreaCorso():

- Scopo: Verificare che il processo di creazione di un nuovo corso avvenga correttamente.
- Realizzazione: Creiamo un personal trainer, un corso, e le relative lezioni. Associamo le lezioni al personal trainer e verifichiamo che il corso e le lezioni siano correttamente registrati nel sistema.

#### TestVerificaCliente():

- Scopo: Verificare la corretta verifica dell'esistenza di un cliente nel sistema.
- Realizzazione: Creiamo un cliente e lo inseriamo nel sistema. Successivamente, verifichiamo che il sistema riconosca correttamente l'esistenza del cliente.

#### TestVisualizzaCorsi():

- Scopo: Verificare la corretta visualizzazione dei corsi disponibili per un cliente.
- Realizzazione: Creiamo un cliente con abbonamento annuale e certificato medico, e aggiungiamo corsi disponibili nel sistema. Verifichiamo che il sistema mostri correttamente i corsi disponibili in base alle condizioni del cliente.

#### TestSelezionaCorso():

- Scopo: Confermare che il processo di selezione di un corso avvenga correttamente.
- Realizzazione: Inizializziamo un set di corsi nel sistema. Selezioniamo un corso specifico e verifichiamo che il corso selezionato corrisponda a quello previsto.

#### TestConfermalscrizione():

- Scopo: Verificare che la conferma di iscrizione di un cliente a un corso conduca a un risultato corretto.
- Realizzazione: Creiamo un cliente, selezioniamo un corso e confermiamo l'iscrizione. Verifichiamo che il corso sia correttamente associato al cliente, che il corso selezionato sia resettato e che la disponibilità del corso sia diminuita.

#### TestVisualizzaCorsiCliente():

- Scopo: Verificare la corretta visualizzazione dei corsi a cui un cliente è iscritto.
- Realizzazione: Creiamo un cliente, associamo corsi al cliente e verifichiamo che il sistema mostri correttamente i corsi a cui il cliente è iscritto.

#### TestSelezionaCorsoRestituisceLezione():

- Scopo: Verificare che il sistema restituisca correttamente le lezioni di un corso selezionato.
- Realizzazione: Creiamo un cliente con un corso associato. Selezioniamo il corso e verifichiamo che il sistema restituisca correttamente le lezioni associate a quel corso.

#### TestSelezionaLezione():

- Scopo: Confermare che il sistema selezioni correttamente una lezione di un corso.
- Realizzazione: Creiamo un cliente con un corso associato. Selezioniamo il corso e una lezione specifica, verificando che la lezione selezionata corrisponda a quella prevista.

#### TestPrenotazionePossibile():

- Scopo: Verificare che il sistema determini correttamente la possibilità di prenotare una lezione.
- Realizzazione: Creiamo un cliente, associamo un corso al cliente e verifichiamo che il sistema riconosca correttamente la possibilità di prenotare o meno una lezione in base alle condizioni date.

#### TestConfermaLezione():

- Scopo: Verificare che la conferma di una lezione da parte di un cliente conduca a un risultato corretto.
- Realizzazione: Creiamo una prenotazione, un cliente e una lezione. Confermiamo la lezione e verifichiamo che la lezione sia correttamente associata alla prenotazione e registrata nel sistema.

#### TestNuovoCorso():

- Scopo: Verificare che il sistema inizi correttamente la creazione di un nuovo corso.
- Realizzazione: Avviamo il processo di creazione di un nuovo corso e verifichiamo che il sistema abbia correttamente impostato il corso corrente.

#### TestConfermaNuovoCorso():

- Scopo: Verificare che il sistema confermi correttamente la creazione di un nuovo corso.
- Realizzazione: Avviamo il processo di creazione di un nuovo corso e successivamente confermiamo la creazione, verificando che il corso corrente sia resettato.

#### TestAssociaPersonalTrainer():

- Scopo: Verificare che il sistema associi correttamente un corso a un personal trainer.
- Realizzazione: Creiamo un corso e un personal trainer. Associamo il corso al personal trainer e verifichiamo che il corso sia correttamente registrato tra quelli associati al personal trainer.

Sono stati effettuati anche dei test unitari in classi differenti da Gym4U per garantire una totale correttezza delle operazioni:

Classe TestCliente:

#### TestSetPrenotazione():

- Scopo: Verificare che il metodo setPrenotazione associ correttamente una prenotazione a un cliente.
- Realizzazione: Creiamo un cliente e una prenotazione, assegnamo la prenotazione al cliente e verifichiamo che la prenotazione sia stata correttamente associata nel sistema.

#### TestSetCorso():

- Scopo: Verificare che il metodo setCorso associi correttamente un corso a un cliente.
- Realizzazione: Creiamo un cliente, un personal trainer e un corso. Assegniamo il corso al cliente e verifichiamo che il corso sia stato correttamente associato nel sistema.

**TestVerificaCertificatoMedico():**

- Scopo: Verificare che il metodo verificaCertificatoMedico restituisca correttamente l'esito della verifica del certificato medico di un cliente.
- Realizzazione: Creiamo un cliente e associamo un certificato medico valido. Successivamente, verifichiamo che il metodo restituisca true indicando che il certificato medico è valido.

**TestVerificaAbbonamento():**

- Scopo: Verificare che il metodo verificaAbbonamento restituisca correttamente l'esito della verifica dell'abbonamento di un cliente.
- Realizzazione: Creiamo un cliente e associamo un abbonamento annuale valido. Verifichiamo che il metodo restituisca true indicando che l'abbonamento è valido.

**Classe TestCorso:**

**TestDiminisciDisponibilità():**

- Scopo: Verificare che il metodo diminisciDisponibilità riduca correttamente la disponibilità di un corso.
- Realizzazione: Creiamo un corso con disponibilità iniziale di 1. Chiamiamo il metodo diminisciDisponibilità una volta e verifichiamo che la disponibilità sia diminuita a 0. Successivamente, chiamiamo nuovamente il metodo e verifichiamo che la disponibilità rimanga a 0, in quanto non può essere inferiore a 0.

**Classe TestPersonalTrainer:**

**TestSetCorso():**

- Scopo: Verificare che il metodo setCorso di un personal trainer associ correttamente un corso a quel personal trainer.
- Realizzazione: Creiamo un corso e un personal trainer. Successivamente, chiamiamo il metodo setCorso del personal trainer e associamo il corso ad esso. Infine, verifichiamo che il corso sia correttamente associato al personal trainer attraverso la mappa dei corsi del personal trainer.

**Classe TestPrenotazione:**

**TestSetLezione():**

- Scopo: Verificare che il metodo setLezione di una prenotazione associ correttamente una lezione a quella prenotazione.
- Realizzazione: Creiamo una prenotazione e una lezione. Successivamente, chiamiamo il metodo setLezione della prenotazione e associamo la lezione ad essa. Infine, verifichiamo che la lezione sia correttamente associata alla prenotazione confrontandola con la lezione restituita dalla prenotazione.

→ Iterazione 2:

Abbiamo progettato e implementato diversi casi di test per valutare la corretta funzionalità dei casi d'uso implementati (UC4-UC5-UC6 ed UC10) in Gym4U.

Di seguito, illustriamo le strategie adottate per l'identificazione di ciascun test.

TestNuovoCliente():

- Scopo: Verificare la corretta creazione di un nuovo cliente nel sistema.
- Realizzazione: Creiamo un nuovo cliente e verifichiamo che il cliente sia correttamente creato, associato a un certificato medico, abbonato, e con un metodo di pagamento valido.

TestConfermaNuovoCliente():

- Scopo: Confermare che il processo di registrazione del cliente conduca a un risultato corretto.
- Realizzazione: Dopo aver registrato un nuovo cliente verifichiamo che le associazioni siano corrette e che venga correttamente restituito il badge associato al nuovo iscritto

TestModificaAbbonamento():

- Scopo: Verificare la corretta modifica dell'abbonamento di un cliente.
- Realizzazione: Registriamo un nuovo cliente, associamo un certificato medico, un abbonamento e un metodo di pagamento. Successivamente, creiamo il nuovo abbonamento che il cliente vuole impostare e verifichiamo chè ciò avvenga.

TestConfermaModificaAbbonamento():

- Scopo: Confermare che il processo di modifica dell'abbonamento conduca a un risultato corretto.
- Realizzazione: Dopo aver registrato un nuovo cliente, associamo un certificato medico, un abbonamento e un metodo di pagamento. Successivamente, modifichiamo l'abbonamento e confermiamo che la modifica sia stata applicata correttamente.

TestModificaMetododiPagamento():

- Scopo: Verificare la corretta modifica del metodo di pagamento di un cliente.
- Realizzazione: Registriamo un nuovo cliente, associamo un certificato medico, un abbonamento e un metodo di pagamento. Successivamente, creiamo il nuovo metodo di pagamento che il cliente vuole impostare e verifichiamo chè ciò avvenga.

TestConfermaModificaMetododiPagamento():

- Scopo: Confermare che il processo di modifica del metodo di pagamento conduca a un risultato corretto.
- Realizzazione: Dopo aver registrato un nuovo cliente, associamo un certificato medico, un abbonamento e un metodo di pagamento. Successivamente, modifichiamo il metodo di pagamento e confermiamo che la modifica sia stata applicata correttamente.

**TestVisualizzaPersonalTrainers():**

- Scopo: Verificare la corretta visualizzazione dei personal trainers disponibili per un cliente.
- Realizzazione: Creiamo un cliente con abbonamento annuale e certificato medico. Confermiamo che la lista dei personal trainer visualizzata sia coerente con quelli disponibili nel sistema.

**TestSelezionaPersonalTrainer():**

- Scopo: Confermare che il processo di selezione di un personal trainer avvenga correttamente.
- Realizzazione: Inizializziamo un set di personal trainers nel sistema. Selezioniamo un personal trainer specifico per un cliente, verificando che il personal trainer selezionato corrisponda a quello previsto e che una nuova lezione sia programmata correttamente.

**TestIsPersonalTrainerDisponibile():**

- Scopo: Verificare se un personal trainer è disponibile in un determinato giorno e orario.
- Realizzazione: Impostiamo un personal trainer come selezionato nel sistema. Testiamo la sua disponibilità in un giorno e orario specifico, confermando che il risultato rifletta correttamente la sua disponibilità.

**TestIsPersonalTrainerDisponibile2():**

- Scopo: Confermare che il sistema riconosca correttamente l'indisponibilità di un personal trainer in un giorno e orario specifico.
- Realizzazione: Impostiamo un personal trainer come selezionato nel sistema. Testiamo la sua indisponibilità in un giorno e orario specifico, confermando che il risultato rifletta correttamente l'indisponibilità.

**TestConfermaPrenotazione():**

- Scopo: Verificare che la conferma di prenotazione di una lezione conduca a un risultato corretto.
- Realizzazione: Creiamo un cliente, selezioniamo un personal trainer e programmiamo una lezione. Confermiamo la prenotazione e verifichiamo che la lezione sia correttamente associata al personal trainer e registrata nel sistema.

**TestIsBadgeValido():**

- Scopo: Verificare la validità di un badge di accesso.
- Realizzazione: Creiamo un cliente con un badge valido e confermiamo che il sistema riconosca correttamente la validità del badge.

**TestIsBadgeValido2():**

- Scopo: Verificare che il sistema riconosca correttamente l'invalidità di un badge di accesso.
- Realizzazione: Creiamo un cliente con un badge valido e confermiamo che il sistema riconosca correttamente l'invalidità del badge inserito per l'ingresso.

#### TestIsPrenotazioneValida():

- Scopo: Verificare la validità di una prenotazione per un cliente.
- Realizzazione: Creiamo un cliente, associamo una prenotazione valida e verifichiamo che il sistema riconosca correttamente la validità della prenotazione.

#### TestIsPrenotazioneValida2():

- Scopo: Verificare che il sistema riconosca correttamente l'invalidità di una prenotazione per un cliente.
- Realizzazione: Creiamo un cliente, associamo una prenotazione valida e verifichiamo che il sistema riconosca correttamente l'invalidità della prenotazione poichè non rispetta le politiche interne della palestra.

#### TestConfermaPresenza():

- Scopo: Verificare che la conferma di presenza ad una lezione conduca a un risultato corretto.
- Realizzazione: Creiamo una prenotazione, confermiamo la presenza e verifichiamo che la conferma di presenza sia registrata correttamente nel sistema.

→ Iterazione 3:

Abbiamo progettato e implementato diversi casi di test per valutare la corretta funzionalità dei casi d'uso implementati (UC7-UC8 ed UC9) in Gym4U.

Di seguito, illustriamo le strategie adottate per l'identificazione di ciascun test.

#### TestInserisciOfferta():

- Scopo: Verificare la corretta procedura di inserimento di un'offerta nel sistema.
- Realizzazione: Creiamo un'offerta con parametri specifici, eseguiamo la procedura di inserimento tramite il metodo inserisciOfferta e verifichiamo che l'offerta corrente nel sistema abbia gli stessi attributi della nuova offerta.

#### TestConfermaOfferta():

- Scopo: Confermare che il processo di inserimento di un'offerta conduca a un risultato corretto.
- Realizzazione: Dopo aver inserito un'offerta, impostiamo l'offerta corrente nel sistema e confermiamo l'inserimento. Verifichiamo che l'offerta inserita sia presente nell'insieme delle offerte del sistema.

#### TestInserisciSchedaPersonalizzata():

- Scopo: Verificare la corretta procedura di inserimento di una scheda personalizzata nel sistema.
- Realizzazione: Creiamo una scheda personalizzata con esercizi e data di scadenza specifici, eseguiamo la procedura di inserimento tramite il metodo inserisciSchedaPersonalizzata e verifichiamo che la scheda personalizzata corrente nel sistema abbia gli stessi attributi della nuova scheda.

TestSelezionaCliente():

- Scopo: Verificare la corretta procedura di selezione di un cliente nel sistema.
- Realizzazione: Creiamo un cliente e lo inseriamo nel sistema. Successivamente, eseguiamo la procedura di selezione tramite il metodo selezionaCliente e verifichiamo che il cliente corrente nel sistema sia lo stesso cliente selezionato.

TestConfermaSchedaPersonalizzata():

- Scopo: Confermare che il processo di inserimento di una scheda personalizzata conduca a un risultato corretto.
- Realizzazione: Dopo aver inserito una scheda personalizzata, impostiamo la scheda corrente nel sistema, selezioniamo un cliente e un personal trainer, e confermiamo l'inserimento della scheda. Verifichiamo che la scheda personalizzata sia associata correttamente al cliente e al personal trainer.

TestVisualizzaLezioniPT():

- Scopo: Verificare la corretta visualizzazione delle lezioni di un personal trainer nel sistema.
- Realizzazione: Impostiamo un personal trainer come personal trainer selezionato nel sistema, quindi eseguiamo la procedura di visualizzazione delle lezioni tramite il metodo visualizzaLezioniPT. Verifichiamo che l'elenco delle lezioni visualizzate corrisponda alle lezioni del personal trainer selezionato.

TestSelezionaLezionePT():

- Scopo: Verificare la corretta procedura di selezione di una lezione di un personal trainer nel sistema.
- Realizzazione: Creiamo una lezione associata a un personal trainer, impostiamo il personal trainer come personal trainer selezionato nel sistema, e eseguiamo la procedura di selezione della lezione tramite il metodo selezionaLezionePT. Verifichiamo che la lezione corrente nel sistema sia la lezione selezionata.

TestVisualizzaPrenotati():

- Scopo: Verificare la corretta visualizzazione dei clienti prenotati per una lezione nel sistema.
- Realizzazione: Impostiamo una lezione come lezione corrente nel sistema, creiamo e inseriamo clienti prenotati per questa lezione. Eseguiamo la procedura di visualizzazione dei prenotati tramite il metodo visualizzaPrenotati e verifichiamo che l'elenco dei clienti visualizzati corrisponda ai clienti prenotati per la lezione corrente.

## Test di sistema

Una fase di test manuale eseguita con successo alla conclusione dell'applicazione ha coinvolto la verifica dell'interfaccia utente e l'analisi delle funzionalità principali. Questa fase di testing è stata progettata per garantire che l'applicazione fosse coerente, performante e conforme alle specifiche del progetto.

Verifica dell'Interfaccia Utente:

**Obiettivo:** Garantire che l'interfaccia utente fosse intuitiva, ben strutturata e rispondesse alle aspettative degli utenti finali (Personal Trainer, Cliente, Admin).

**Attività:** Navigazione attraverso le diverse schermate dell'applicazione per testare la chiarezza dell'organizzazione, la facilità di utilizzo e la consistenza del design.

Test delle Funzionalità Principali:

**Obiettivo:** Verificare che tutte le funzionalità principali identificate nelle specifiche fossero implementate correttamente e rispondessero ai requisiti.

**Attività:** Esecuzione di scenari di utilizzo tipici per ciascun tipo di utente. Ad esempio, creazione di programmi di allenamento come Personal Trainer, prenotazione di sessioni come Cliente, e gestione di utenti come Admin.

Analisi della Coerenza dei Dati:

**Obiettivo:** Assicurarsi che i dati visualizzati fossero coerenti con quelli memorizzati nel sistema.

**Attività:** Esplorazione dei dati visualizzati nelle interfacce utente confrontandoli con quelli presenti nel database, verificando la consistenza e l'accuratezza.

Valutazione delle Prestazioni:

**Obiettivo:** Garantire che l'applicazione fosse reattiva e performante durante le interazioni utente.

**Attività:** Test di carico simulato e analisi delle risposte del sistema in situazioni di utilizzo intensivo.

In conclusione, la fase di test manuale ha restituito risultati positivi, confermando che l'applicazione era coerente con le specifiche del progetto e che le funzionalità principali erano implementate correttamente. La verifica dell'interfaccia utente ha confermato un'usabilità intuitiva, mentre l'analisi della coerenza dei dati ha evidenziato l'integrità del sistema. La valutazione delle prestazioni ha indicato che l'applicazione era reattiva e performante anche in condizioni di utilizzo intenso. L'esito complessivo della fase di test manuale è stato pertanto positivo, preparando l'applicazione per il rilascio e l'adozione da parte degli utenti.

# Refactoring e conclusioni

## Refactoring

Abbiamo implementato una procedura di accesso dell'utente a Gym4U attraverso la richiesta di un codice (codice univoco per Amministratore, Cliente e Personal Trainer; 1 per simulare l'accesso in palestra) e una password ("0" di default). Una volta che le credenziali sono verificate con successo, il sistema mostra un menu corrispondente al ruolo dell'utente (Cliente, Amministratore, o Personal Trainer). Questo approccio permette una gestione differenziata delle funzionalità in base al tipo di utente.

Il ciclo di input controlla la correttezza delle credenziali e seleziona il menu appropriato in base al ruolo dell'utente autenticato. Successivamente, l'utente può scegliere varie opzioni nel menu, o uscire dal sistema. Le opzioni del menu sono specifiche per il ruolo dell'utente: questo meccanismo di autenticazione e gestione dei menu consente una navigazione intuitiva e sicura all'interno dell'applicazione, offrendo all'utente solo le opzioni pertinenti al suo ruolo e garantendo un'esperienza utente coerente e sicura.

Parallelamente, abbiamo introdotto i casi d'uso CRUD UC11, UC12, UC13, UC14 e UC15 completando così l'insieme delle operazioni fondamentali per il sistema. Questi casi d'uso consentono di gestire in modo completo le informazioni relative ai clienti, agli amministratori e ai personal trainer all'interno dell'applicazione, garantendo un'esperienza utente fluida e completa.

## Test di accettazione

In conclusione delle attività propedeutiche per portare a termine le fasi precedenti, è stato eseguito un meticoloso esame dell'intero codice sorgente. Questo processo è culminato con la realizzazione del test di accettazione conclusivo, mirato a simulare le esperienze degli utenti nelle diverse veste di Personal Trainer, Cliente e Amministratore.

Nel corso di questa fase di testing, sono stati meticolosamente esplorati e verificati i vari casi d'uso identificati. Questo approccio ha permesso di valutare l'applicazione da diverse prospettive utente, garantendo che tutte le funzionalità rispondano alle aspettative e soddisfino i requisiti specificati nelle fasi precedenti del processo di sviluppo.

L'esecuzione del test di accettazione conclusivo rappresenta un passo chiave verso l'assicurazione della qualità del prodotto software, consentendo di identificare eventuali anomalie o miglioramenti necessari. Tale fase sottolinea l'importanza di una revisione accurata e di test approfonditi per garantire la robustezza e l'affidabilità dell'applicazione prima della sua distribuzione o implementazione in ambienti operativi reali.