# 03- Elaborazione - Iterazione 2

## 3.1 Introduzione

Durante questa seconda iterazione ci si concentrerà su:

- 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 principale di successo dei 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.

### 3.1.1 Modello dei Casi d'Uso

Riportiamo i casi d'uso analizzati in formato dettagliato.

**UC4: Registrazione nuovo Cliente**

| | |
|---|---|
| **Nome** | UC4: Registrazione nuovo cliente |
| **Portata** | Gestione sistema Gym4U |
| **Livello** | Obiettivo utente |
| **Attore primario** | Amministratore del Sistema |
| **Parti interessate e interessi** | • Amministratore del Sistema: Desidera registrare un nuovo cliente nel sistema.<br>• Palestra: Desidera mantenere un elenco aggiornato degli iscritti con informazioni complete.<br>• Cliente (Nuovo Iscritto): Desidera essere registrato correttamente nel sistema per accedere alle funzionalità della palestra. |
| **Pre-condizioni** | L'Amministratore è identificato e autenticato nel sistema Gym4U. |
| **Garanzia di successo** | 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. |
| **Scenario principale di successo** | 1. L'Amministratore seleziona l'opzione "Registrazione Nuovo Cliente".<br>2. Inserisce i dettagli del nuovo cliente, come nome, cognome, data di nascita, indirizzo, e contatti.<br>3. Il Sistema ritorna le informa inserite. |

| | |
|---|---|
| | 4. L’amministratore verifica che il certificato medico fornito sia valido e lo registra.<br>5. L'Amministratore associa l'iscritto all’abbonamento richiesto.<br>6. L’Amministratore associa il Metodo di Pagamento con i dati comunicati dal Cliente.<br>7. Il Sistema ritorna tutte le informazioni inserite.<br>8. L'Amministratore conferma la registrazione del nuovo Cliente.<br>9. Il Sistema registra le informazioni del nuovo Cliente.<br>10. L’amministratore rilascia il badge univoco al Cliente. |
| **Estensioni** | ***a**: In qualsiasi momento, il Sistema fallisce:<br>1. L'Amministratore del Sistema riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.<br>2. Il Sistema ricostruisce lo stato precedente.<br>a. Il Sistema rileva delle anomalie che impediscono il ripristino:<br>i. Il Sistema segnala un errore all'Amministratore del Sistema, registra l'errore, e passa in uno stato pulito.<br>ii. L'Amministratore del Sistema inizia una nuova registrazione del cliente.<br><br>**2.a**: L'Amministratore inserisce i dettagli del nuovo cliente, ma il sistema segnala un errore di validazione:<br>a. Formato del numero di telefono o email non valido:<br>1. Il sistema notifica all'Amministratore l'errore e chiede di correggere i dati.<br>2. L'Amministratore corregge i dettagli del nuovo cliente secondo le indicazioni del sistema.<br>3. Il sistema verifica nuovamente i dati inseriti e, se corretti, procede con la registrazione del cliente.<br>4. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico.<br>b. Data di nascita inserita in contrasto con la Regola di Business R7:<br>1. Il sistema notifica all'Amministratore che il cliente non può essere registrato in quanto in contrasto con la Regola di Business R7<br><br>**4.a:** Il Certificato Medico è scaduto:<br>1. Il Sistema comunica che il Certificato Medico non è valido.<br>2. Il Sistema interrompe la registrazione del Cliente.<br><br>**6.a:** Il Metodo di Pagamento è scaduto:<br>1. Il Sistema comunica che il Metodo di Pagamento non è valido.<br>2. Invita l’Amministratore a far comunicare al Cliente un Metodo di Pagamento valido. |
| **Requisiti speciali** | Interfaccia grafica intuitiva e di facile utilizzo. |

| | |
|---|---|
| | Il sistema deve gestire l'associazione degli iscritti agli abbonamenti in modo accurato.<br>Il tempo di risposta del sistema deve essere entro pochi secondi.<br>Lettore NFC compatibile con i badge. |
| **Elenco delle varianti tecnologiche e dei dati** | Il sistema deve essere in grado di gestire diverse tipologie di abbonamenti e dettagli degli iscritti.<br>Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande. |
| **Frequenza di ripetizioni** | Decine di volte al giorno, soprattutto nei periodi di nuove iscrizioni o promozioni. |
| **Varie** | |

**UC5: Gestione Abbonamento**

| | |
|---|---|
| **Nome** | UC5: Gestione Abbonamento |
| **Portata** | Gestione sistema Gym4U |
| **Livello** | Obiettivo utente |
| **Attore primario** | Cliente |
| **Parte interessate e interessi** | • Amministratore del Sistema: Desidera che il sistema di pagamento degli abbonamenti sia automatizzato<br>• Palestra: Desidera mantenere aggiornate le informazioni sugli abbonamenti e i metodi di pagamento dei suoi iscritti.<br>• Cliente: Desidera gestire il proprio abbonamento e il metodo di pagamento attraverso Gym4U. |
| **Pre-condizioni** | L'Utente è autenticato nel sistema Gym4U. |
| **Garanzia di successo** | L'Utente modifica correttamente l'abbonamento e/o il metodo di pagamento.<br>Le modifiche sono riflesse immediatamente nel sistema e nell'inventario della palestra. |
| **Scenario principale di successo** | 1. L'Utente accede alla sezione "Gestione Abbonamento" in Gym4U.<br>2. L'Utente visualizza le attuali impostazioni di abbonamento e di pagamento.<br>3. L'Utente seleziona quale impostazione vuole modificare.<br>4. L'Utente modifica l'abbonamento o il metodo di pagamento.<br>5. Il sistema ritorna il riepilogo delle impostazioni di abbonamento o del metodo di pagamento pagamento aggiornate.<br>6. L'Utente conferma le modifiche.<br>7. Il sistema salva le modifiche. |
| **Estensioni** | ***a**: In qualsiasi momento, il Sistema fallisce: |

| | |
|---|---|
| | 1. Il Cliente riavvia il Sistema, si autentica, e richiede il ripristino dello stato precedente.<br>2. Il Sistema ricostruisce lo stato precedente.<br>a. Il Sistema rileva delle anomalie che impediscono il ripristino:<br>i. Il Sistema segnala un errore al Cliente, registra l'errore, e passa in uno stato pulito.<br>ii. Il Cliente prova nuovamente la procedura di gestione dell'abbonamento.<br>**4.a**: Il sistema rileva un problema nell'aggiornamento istantaneo delle modifiche nell'inventario:<br>1. Il sistema tenta nuovamente l'aggiornamento:<br>a. Il sistema fa un altro tentativo di aggiornare le modifiche nell'inventario.<br>b. Se l'aggiornamento ha successo, il caso d'uso prosegue normalmente.<br>2. Il sistema rileva un problema persistente nell'aggiornamento:<br>a. Il sistema registra l'errore nell'aggiornamento delle modifiche.<br>b. L'utente viene invitato a contattare l'assistenza. |
| **Requisiti speciali** | Interfaccia utente intuitiva per la modifica dell'abbonamento e del metodo di pagamento.<br>Aggiornamento istantaneo delle modifiche nell'inventario della palestra. |
| **Elenco delle varianti tecnologiche e dei dati** | Gestione dei diversi tipi di abbonamenti e dei relativi metodi di pagamento.<br>Integrazione immediata e accurata delle modifiche nei dati dell'abbonamento. |
| **Frequenza di ripetizioni** | Relativo, con periodi di modifiche degli abbonamenti durante offerte speciali o cambiamenti nei requisiti dell'utente. |
| **Varie** | |

**UC6: Prenotazione Personal Trainer**

| | |
|---|---|
| **Nome** | UC6: Prenotazione Personal Trainer |
| **Portata** | Gestione sistema Gym4U |
| **Livello** | Obiettivo utente |
| **Attore primario** | Cliente |
| **Parti interessate e interessi** | • Amministratore del Sistema: Desidera gestire e monitorare le prenotazioni dei Personal Trainer.<br>• Personal Trainer: Desidera ricevere notifiche sulle prenotazioni dei clienti. |

| | |
|---|---|
| | ● Cliente: Desidera prenotare una sessione con un Personal Trainer. |
| **Pre-condizioni** | L'Utente è identificato e autenticato nel sistema Gym4U. |
| **Garanzia di successo** | La prenotazione della sessione con il Personal Trainer è registrata nel sistema. L'Utente e il Personal Trainer ricevono notifiche. |
| **Scenario principale di successo** | 1. L'utente accede alla sezione "Prenotazioni Personal Trainer" su Gym4U.<br>2. Il Sistema verifica che il Cliente abbia un abbonamento attivo.<br>3. Il Sistema verifica che il Cliente abbia un certificato medico valido.<br>4. Il Cliente visualizza l'elenco dei Personal Trainer.<br>5. L'Utente seleziona il Personal Trainer desiderato e sceglie una data ed un orario per la sessione.<br>6. Il Sistema verifica la disponibilità del Personal Trainer e ritorna le informazioni inserite.<br>7. Il Cliente conferma la prenotazione.<br>8. Il Sistema invia una notifica all'Utente.<br>9. Il sistema invia una notifica al Personal Trainer riguardo alla nuova prenotazione. |
| **Estensioni** | ***a**: In qualsiasi momento, il Sistema fallisce:<br>1. L'Utente riavvia il Sistema, si autentica e richiede il ripristino dello stato precedente.<br>2. Il Sistema ricostruisce lo stato precedente.<br>a. Il Sistema rileva delle anomalie che impediscono il ripristino:<br>i. Il Sistema segnala un errore all'Utente, registra l'errore e passa in uno stato pulito.<br>ii. L'Utente inizia una nuova prenotazione con il Personal Trainer.<br><br>**6.a**: Il Personal Trainer non è disponibile nella data/orario selezionati:<br>1. Il Sistema notifica al Cliente e suggerisce alternative di data/orario.<br>2. L'Utente seleziona una nuova data/orario o sceglie un altro Personal Trainer disponibile.<br><br>**6.b**: Il sistema segnala un errore durante la conferma della prenotazione:<br>1. Il Sistema notifica all'Utente l'errore e chiede di riprovare.<br>2. L'Utente corregge gli errori secondo le indicazioni del sistema.<br>3. Il Sistema verifica nuovamente i dati inseriti e, se corretti, procede con la prenotazione.<br>4. Se il problema persiste, il sistema fornisce un messaggio di errore e suggerisce di contattare il supporto tecnico. |

| | |
|---|---|
| **Requisiti speciali** | Interfaccia grafica intuitiva per la prenotazione.<br>Sistema di notifiche efficiente per informare l'Utente e il Personal Trainer.<br>Gestione accurata della disponibilità dei Personal Trainer.<br>Tempo di risposta del sistema entro pochi secondi. |
| **Elenco delle varianti tecnologiche e dei dati** | Il sistema deve gestire diversi Personal Trainer e le relative disponibilità.<br>Supporto per la visualizzazione chiara e gestione efficace dei dati su un monitor piatto grande. |
| **Frequenza di ripetizioni** | Abbastanza frequente, varia in base alle richieste degli Utenti e alla disponibilità dei Personal Trainer. |
| **Varie** | |

**UC10: Accesso in Palestra tramite Badge**

| | |
|---|---|
| **Nome** | UC10: Accesso in Palestra tramite Badge |
| **Portata** | Gestione sistema Gym4U |
| **Livello** | Obiettivo utente |
| **Attore primario** | Cliente |
| **Parti interessate e interessi** | • Cliente: Desidera accedere alla palestra tramite il proprio badge e confermare la propria presenza.<br>• 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. |
| **Pre-condizioni** | Il Cliente ha prenotato un'attività o una sessione in palestra tramite Gym4U. |
| **Garanzia di successo** | Il Cliente ottiene l'accesso alla palestra passando il proprio badge univoco. |
| **Scenario principale di successo** | 1. Il Cliente si presenta al tornello d'ingresso della palestra.<br>2. Il Cliente mostra il proprio badge all'apposito lettore.<br>3. Il Sistema verifica la validità del badge.<br>4. Il Sistema verifica che il Cliente abbia una prenotazione valida.<br>5. Il Sistema conferma la presenza del Cliente.<br>6. Il tornello è libero di poter girare per far entrare il Cliente. |
| **Estensioni** | ***a**: In qualsiasi momento, il Sistema fallisce:<br>1. Il Cliente riavvia il Sistema, si autentica e richiede il ripristino |

| | |
|---|---|
| | dello stato precedente.<br>2. Il Sistema ricostruisce lo stato precedente.<br>a. Il Sistema rileva delle anomalie che impediscono il ripristino:<br>i. Il Sistema segnala un errore al Cliente, registra l'errore e passa in uno stato pulito.<br>ii. Il Cliente inizia un nuovo accesso in palestra.<br><br>**3.a**: Il badge non è valido:<br>1a: Il Sistema segnala che il badge non è stato riconosciuto:<br>a. Il Sistema invita il Cliente a riprovare.<br>1b: Il Sistema segnala che il badge è stato già usato:<br>a. Il Sistema invita il Cliente a contattare l'assistenza.<br><br>**4.a:** Il Cliente non ha una prenotazione valida:<br>1a: Il Cliente non ha una lezione prenotata per quel giorno:<br>a. Il Sistema invita al Cliente a prenotarsi alla lezione desiderata.<br>1b: Il Cliente sta violando la Regola di Business R4:<br>a. Il Sistema invita il Cliente a presentarsi al più mezz'ora prima dell'inizio della lezione per la quale è prenotato.<br><br>**6.a**: Il tornello ha un malfunzionamento:<br>1. Il Cliente constata che il tornello non si è sbloccato.<br>2. Il Cliente contatta l'assistenza. |
| **Requisiti speciali** | Lettore NFC installato ai tornelli d'ingresso.<br>Interfaccia veloce e affidabile per la scansione e la verifica dei badge. |
| **Elenco delle varianti tecnologiche e dei dati** | Utilizzo di tecnologie di scansione NFC integrate nei tornelli d'ingresso della palestra.<br>Archiviazione delle informazioni di prenotazione dei clienti per la verifica dell'accesso.<br>Connessione dati sicura per garantire l'affidabilità dell'accesso tramite badge. |
| **Frequenza di ripetizioni** | Frequente, ogni volta che i Clienti devono accedere alle attività della palestra tramite Gym4U. |
| **Varie** | |

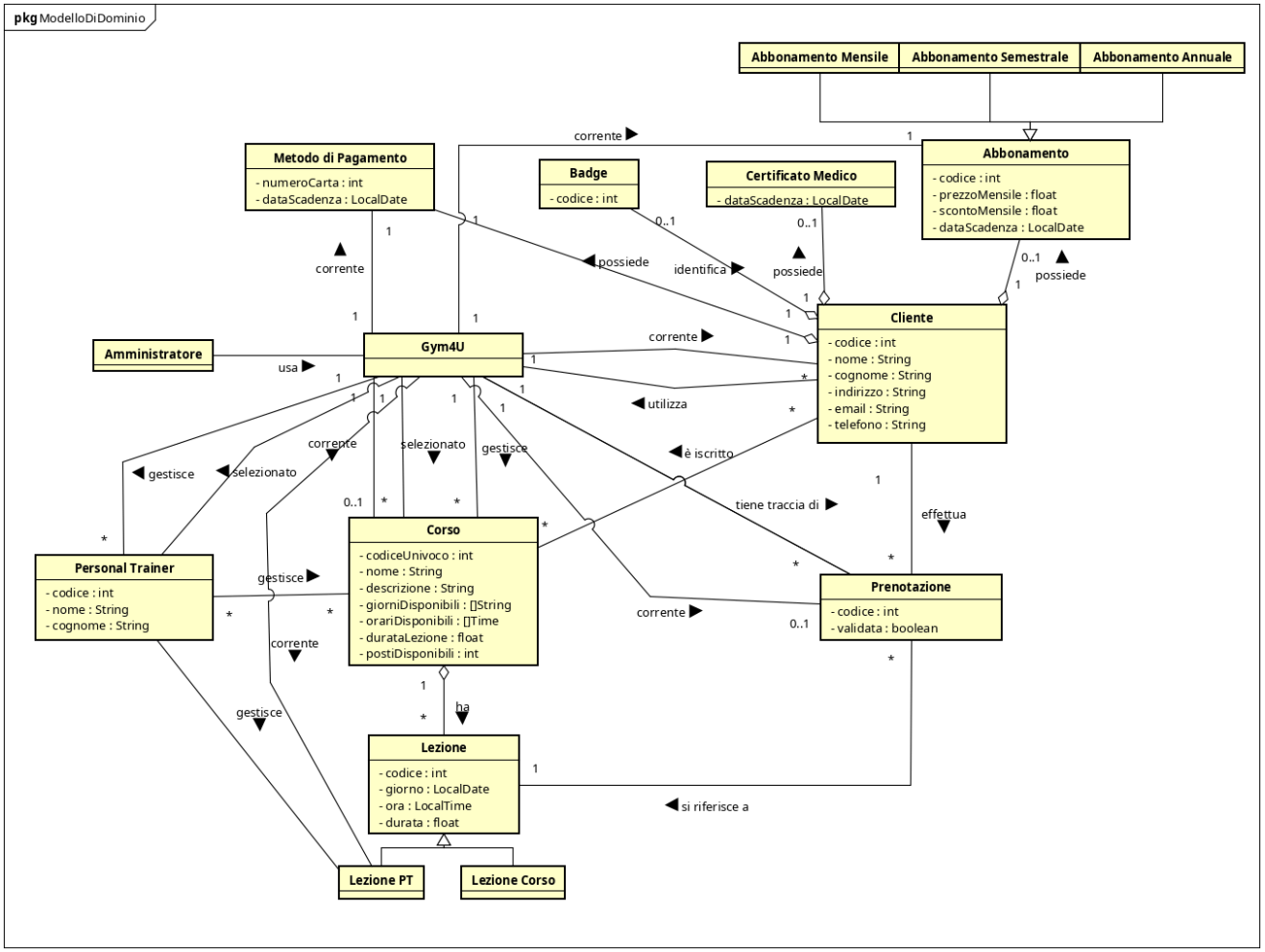
## 3.2 Analisi Orientata agli Oggetti

Al fine di descrivere il dominio da un punto di vista ad oggetti e gestire ulteriori requisiti, saranno utilizzati nuovamente gli stessi strumenti dell'iterazione precedente (Modello di Dominio, SSD  Sequence System Diagram e Contratti delle operazioni). In particolare i paragrafi seguenti permettono di evidenziare i cambiamenti che tali elaborati hanno subito rispetto alla fase precedente.

### 3.2.1 Modello di Dominio

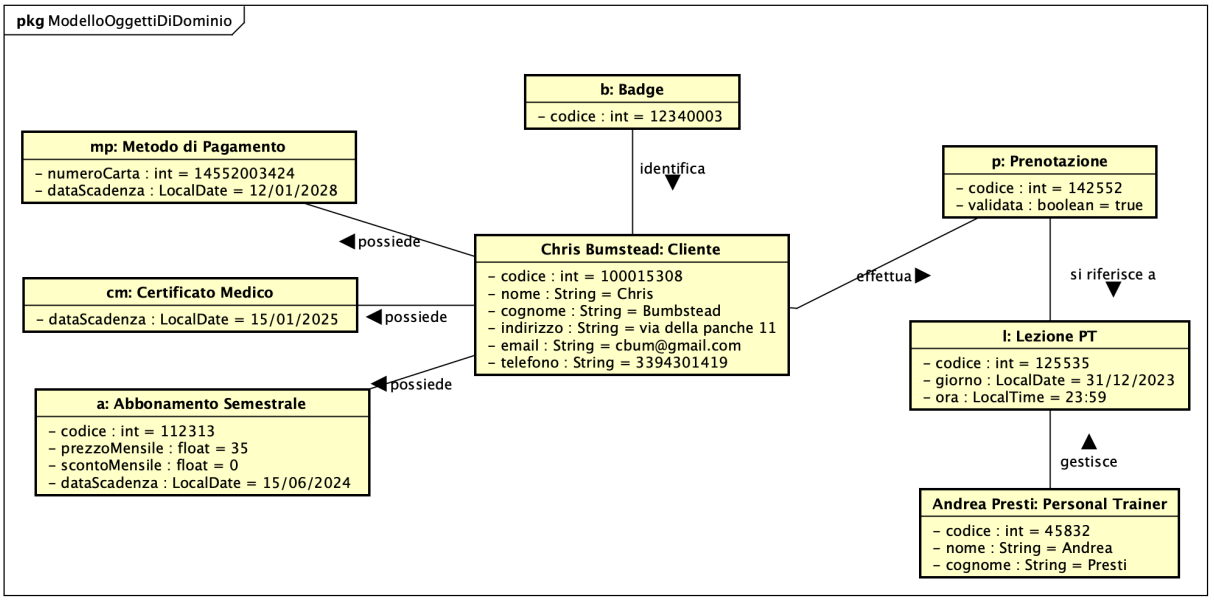
Relativamente ai casi d'uso scelto, dopo un'attenta valutazione dello scenario principale di successo è stato possibile identificare le seguenti classi concettuali da aggiungere al modello precedentemente delineato:

- 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.
- 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.
- 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.



### 3.2.2 Modello degli Oggetti di Dominio

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.
Presentiamo come esempio il seguente scenario:
“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”

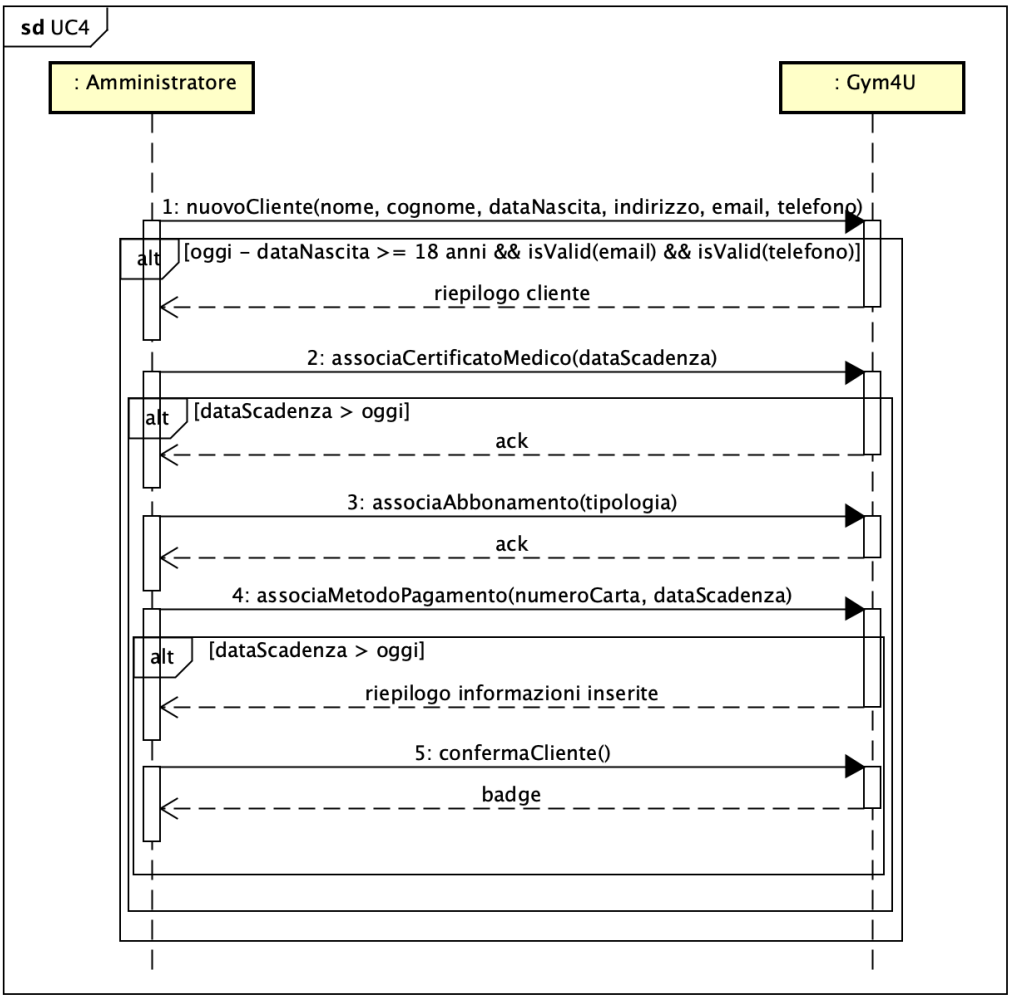


### 3.2.3 Diagramma di sequenza di sistema

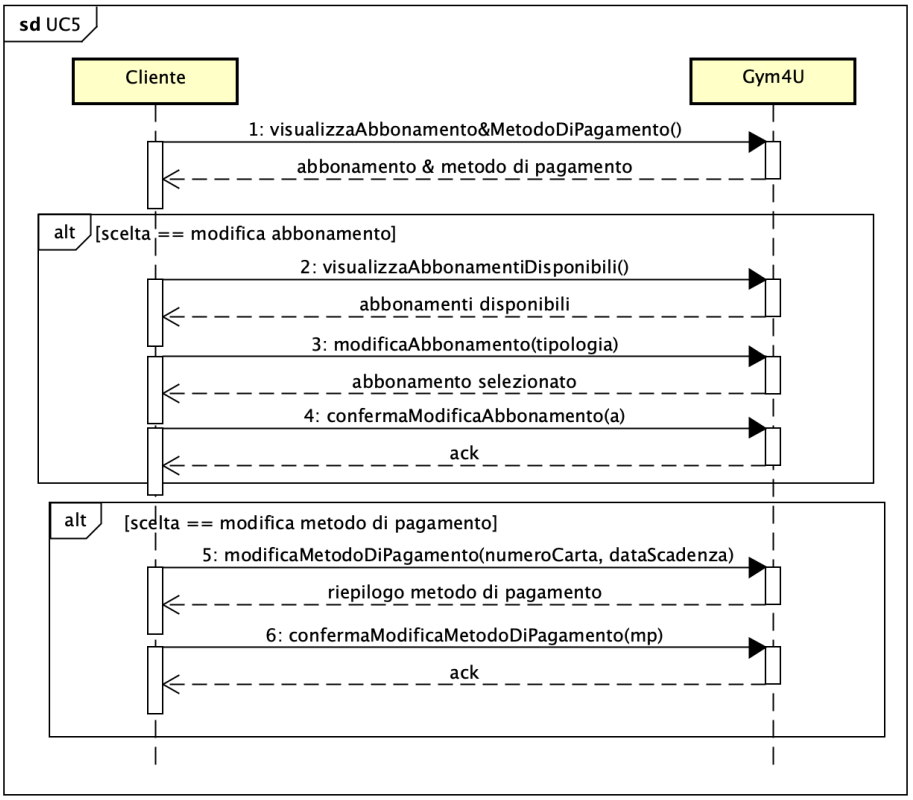
Procedendo con l'analisi Orientata agli Oggetti, il passo successivo è la creazione del Diagramma di Sequenza di Sistema (SSD) al fine di illustrare il corso degli eventi di input e di output per lo scenario principale di successo dei casi d'uso scelti (UC4, UC5, UC6 ed UC10), quindi avremo:

#### 3.2.3.1 - SSD Caso d'Uso UC4

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.

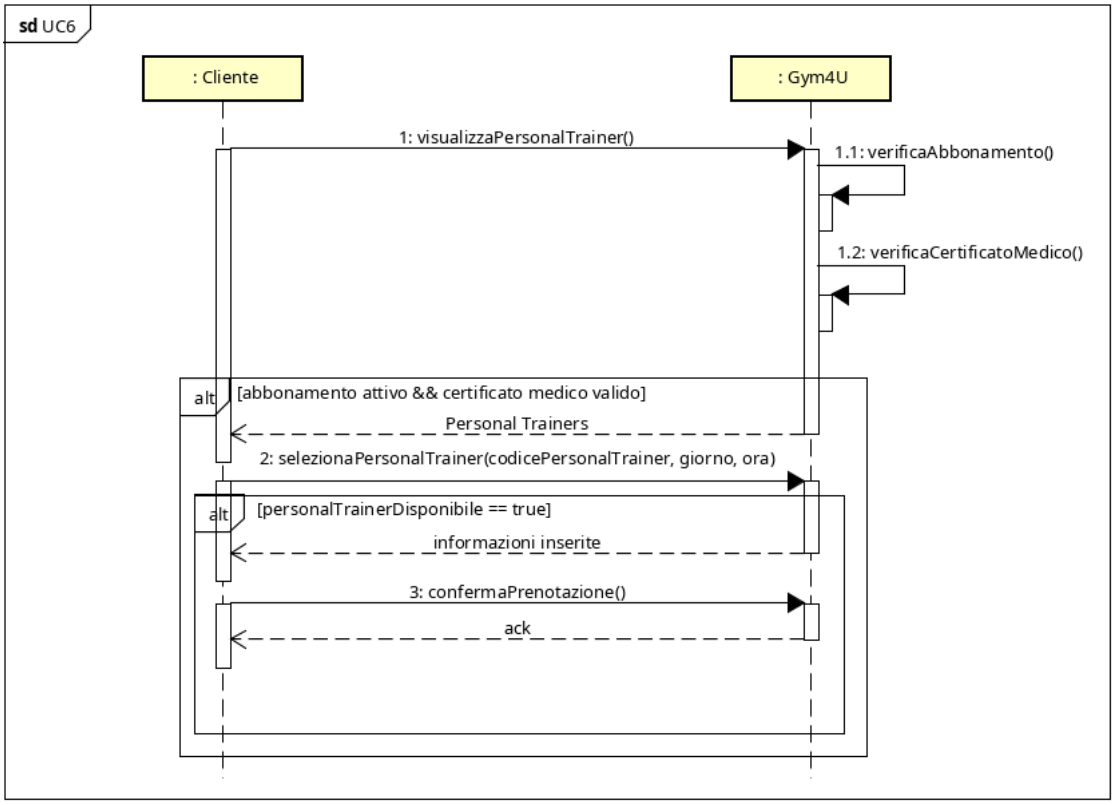


### 3.2.3.2 - SSD Caso d'Uso UC5

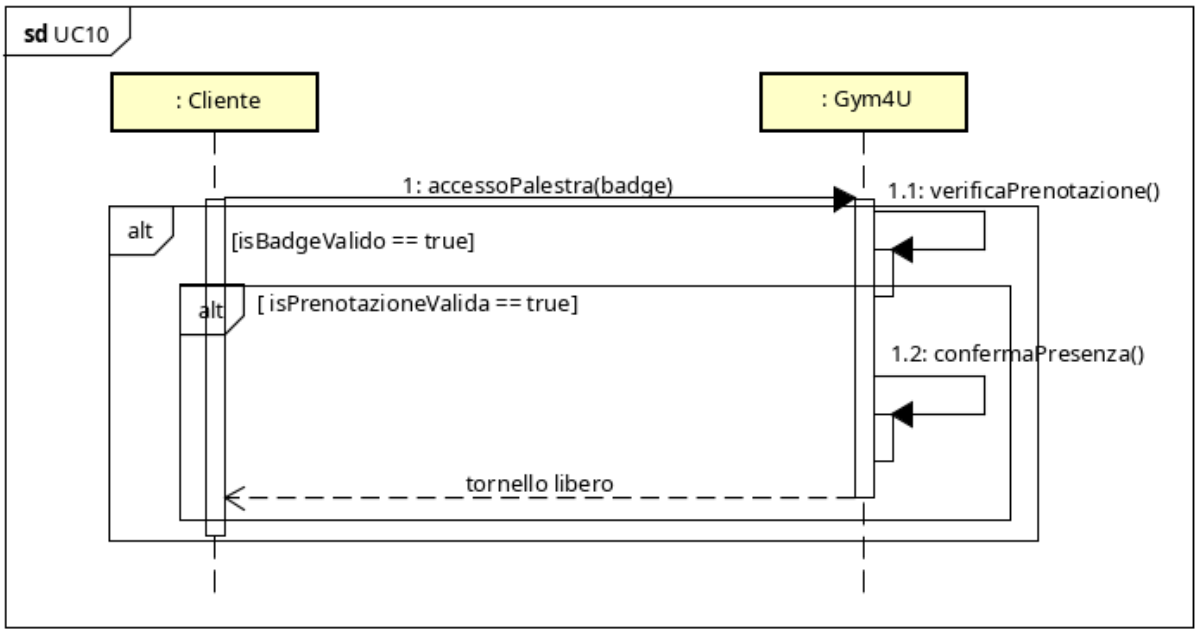



### 3.2.3.3 - SSD Caso d'Uso UC6

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



#### 3.2.3.4 - SSD Caso d'Uso UC10




### 3.2.4 Contratti delle operazioni

Vengono ora descritte attraverso i Contratti le principali operazioni di sistema che si occupano di gestire gli eventi di sistema individuati negli SSD.

#### 3.2.4.1 Contratti Operazioni UC4

Contratto CO1: nuovoCliente

| | |
|---|---|
| **Operazione** | nuovoCliente(nome: String, cognome: String, indirizzo: String, email: String, telefono: String) |
| **Riferimenti** | Caso d'Uso UC4: Registrazione nuovo Cliente |
| **Pre-Condizioni** | L'amministratore è autenticato in Gym4U |
| **Post-Condizioni** | • È stata creata un'istanza cliente di Cliente<br>• cliente è stato inizializzato con le informazioni inserite<br>• cliente è stato associato a Gym4U tramite l'associazione "corrente" |

Contratto CO2: associaCertificatoMedico

| | |
|---|---|
| **Operazione** | associaCertificatoMedico(dataScadenza: LocalDate) |
| **Riferimenti** | Caso d'Uso UC4: Registrazione nuovo Cliente |
| **Pre-Condizioni** | È in corso la registrazione di un nuovo Cliente cliente |
| **Post-Condizioni** | • È stata creata un'istanza cm di Certificato Medico<br>• cm è stato inizializzato con le informazioni inserite<br>• cm è stato associato a Cliente tramite l'associazione "possiede" |

Contratto CO3:  associaAbbonamento

| **Operazione** | associaAbbonamento(tipologia: String) |
|---|---|
| **Riferimenti** | Caso d'Uso UC4: Registrazione nuovo Cliente |
| **Pre-Condizioni** | È in corso la registrazione di un nuovo Cliente cliente |
| **Post-Condizioni** | ● È stata creata un'istanza a di Abbonamento<br>● a è stato inizializzato con le informazioni inserite<br>● a è stato associato a Cliente tramite l'associazione "possiede" |

Contratto CO4:  associaMetodoPagamento

| **Operazione** | associaMetodoPagamento(numeroCarta: String, dataScadenza: LocalDate) |
|---|---|
| **Riferimenti** | Caso d'Uso UC4: Registrazione nuovo Cliente |
| **Pre-Condizioni** | È in corso la registrazione di un nuovo Cliente cliente |
| **Post-Condizioni** | ● È stata creata un'istanza mp di Metodo di Pagamento<br>● mp è stato inizializzato con le informazioni inserite<br>● mp è stato associato a Cliente tramite l'associazione "possiede" |

Contratto CO4:  confermaCliente

| **Operazione** | confermaCliente() |
|---|---|
| **Riferimenti** | Caso d'Uso UC4: Registrazione nuovo Cliente |
| **Pre-Condizioni** | È in corso la registrazione di un nuovo Cliente cliente |
| **Post-Condizioni** | ● È stata creata un'istanza b di Badge<br>● b è stato associato a cliente tramite l'associazione "identifica"<br>● L'istanza corrente cliente è stata associata a Gym4U tramite l'associazione "utilizza"<br>● È stata eliminata l'associazione "corrente" tra Gym4U e l'istanza cliente |

#### 3.2.4.2 Contratti Operazioni UC5

CO1: modificaAbbonamento

| **Operazione** | modificaAbbonamento(abbonamento: String) |
|---|---|
| **Riferimenti** | Caso d'Uso UC5: Gestione Abbonamento |
| **Pre-Condizioni** | L'Utente è autenticato ed iscritto a Gym4U. |
| **Post-Condizioni** | ● È stata creata un'istanza corrente a di Abbonamento tramite l'associazione "corrente"<br>● a è stata inizializzata con le informazioni inserite |

CO2: confermaModificaAbbonamento

| **Operazione** | confermaAbbonamento() |
|---|---|
| **Riferimenti** | Caso d'Uso UC5: Gestione Abbonamento |
| **Pre-Condizioni** | Esiste un'istanza a di Abbonamento |
| **Post-Condizioni** | ● L'istanza a di Abbonamento viene associata al Cliente c tramite l'associazione "possiede"<br>● L'istanza corrente a di Abbonamento viene eliminata |

CO3: modificaMetodoDiPagamento

| **Operazione** | modificaMetodoDiPagamento(numeroCarta: int, dataScadenza: LocalDate) |
|---|---|
| **Riferimenti** | Caso d'Uso UC5: Gestione Abbonamento |
| **Pre-Condizioni** | L'Utente è autenticato ed iscritto a Gym4U. |
| **Post-Condizioni** | ● È stata creata un'istanza corrente mp di MetodoDiPagamento<br>● mp è stata inizializzata con le informazioni inserite |

CO4: confermaModificaMetodoDiPagamento

| **Operazione** | confermaMetodoDiPagamento() |
|---|---|
| **Riferimenti** | Caso d'Uso UC5: Gestione Abbonamento |
| **Pre-Condizioni** | Esiste una lista di Personal Trainer personalTrainers |
| **Post-Condizioni** | ● L'istanza mp di MetodoDiPagamento viene associata al Cliente c tramite l'associazione "possiede" |

#### 3.2.4.3 Contratti Operazioni UC6

Contratto CO1: selezionaPersonalTrainer

| | |
|---|---|
| **Operazione** | selezionaPersonalTrainer(codicePersonalTrainer: int, giorno: LocalDate, ora: LocalTime) |
| **Riferimenti** | Caso d'Uso UC6: Prenotazione Personal Trainer |
| **Pre-Condizioni** | Esiste una lista di Personal Trainer personalTrainers |
| **Post-Condizioni** | ● È stata creata un'istanza corrente l di LezionePT<br>● l è stata inizializzata con le informazioni inserite<br>● È stata selezionata l'istanza pt di Personal Trainer |

Contratto CO2: confermaPrenotazione

| | |
|---|---|
| **Operazione** | confermaPrenotazione() |
| **Riferimenti** | Caso d'Uso UC6: Prenotazione Personal Trainer |
| **Pre-Condizioni** | ● È in corso la prenotazione di una Lezione Personal Trainer<br>● Esiste un'istanza l di LezionePT |
| **Post-Condizioni** | ● È stata creata un'istanza p di Prenotazione<br>● L'istanza p è stata associata alla Lezione l<br>● L'istanza p è stata associata al Cliente cliente<br>● L'istanza p viene associata a Gym4U tramite l'associazione "tiene traccia di"<br>● È stata eliminata l'associazione "corrente" tra l e Gym4U<br>● L'istanza l è stato associata a pt tramite l'associazione "gestisce" |

#### 3.2.4.4 Contratti Operazioni UC10

Contratto CO1: confermaPresenza

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

## 3.3 Progettazione

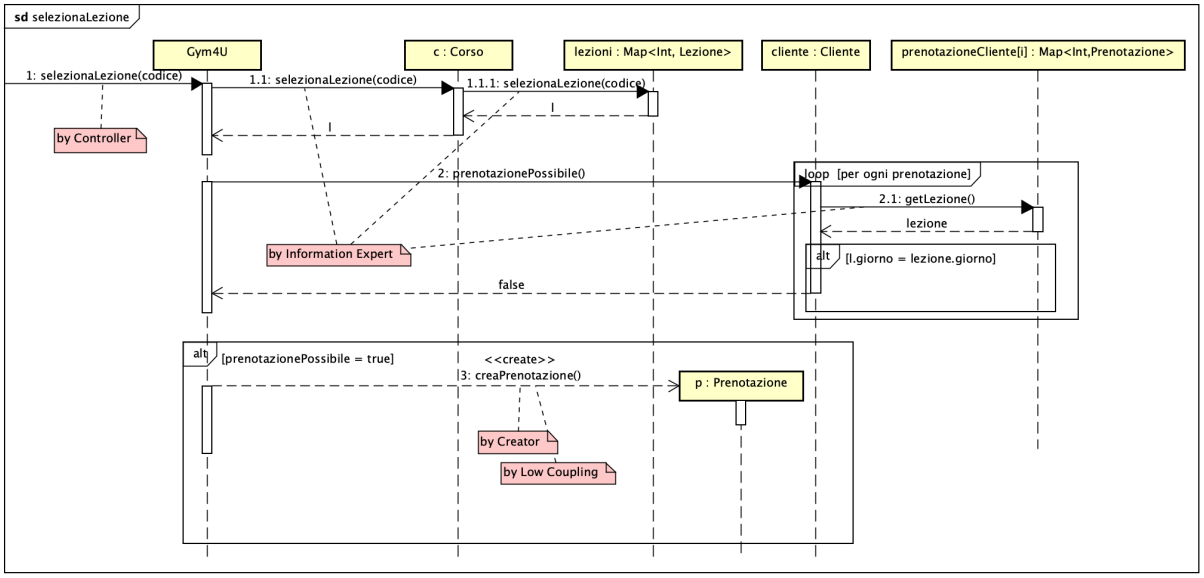
L’elaborato principale di questa fase che è stato preso in considerazione è il Modello di Progetto, ovvero l'insieme dei diagrammi che descrivono la progettazione logica sia da un punto di vista dinamico (Diagrammi di Interazione) che da un punto di vista statico (Diagramma delle Classi). Seguono dunque i diagrammi di Interazione più significativi e il diagramma delle Classi relativi al caso d’uso UC4, UC5, UC6 ed UC10 determinati a seguito di un attento studio degli elaborati scritti in precedenza.

### 3.1.1 Diagrammi di Sequenza

#### 3.3.1.1 - UC2

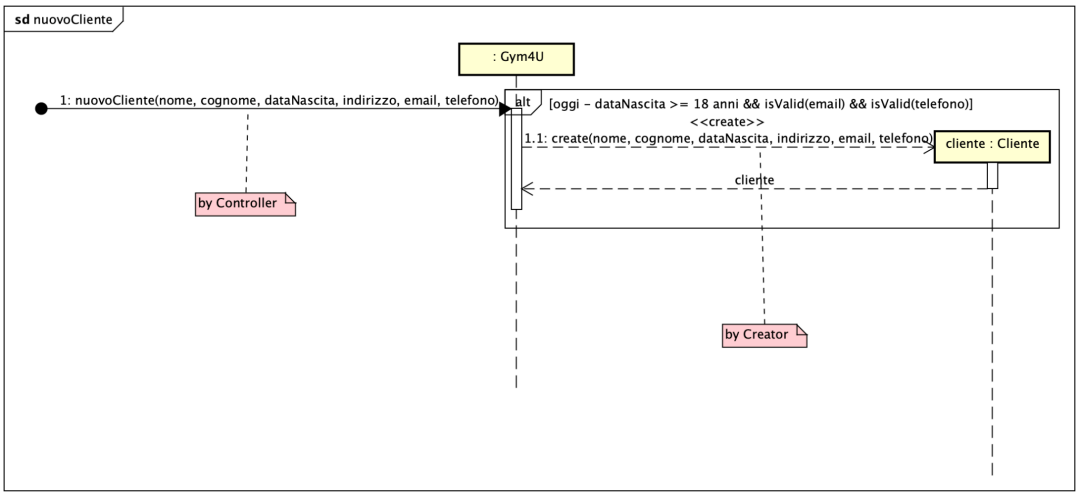
- selezionaLezione

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

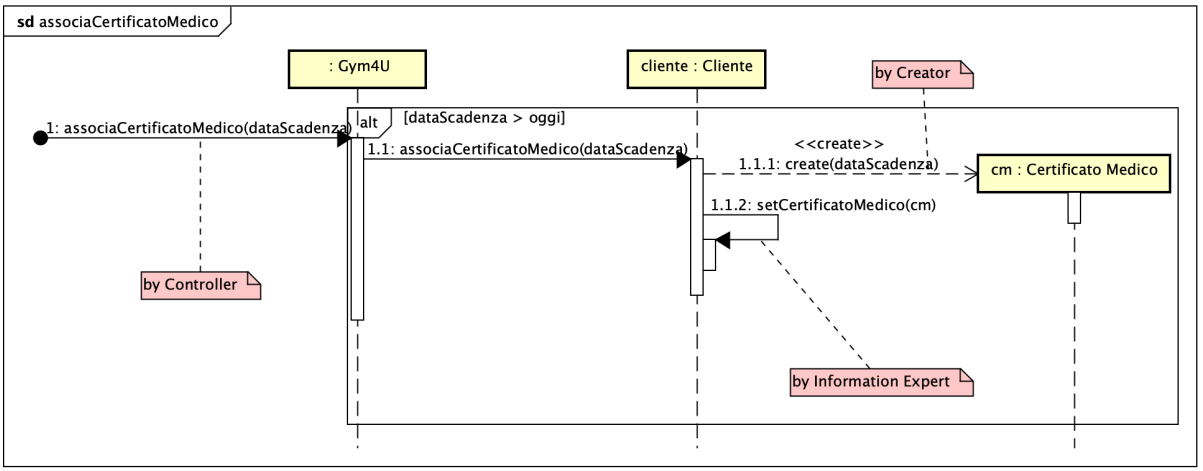



#### 3.3.1.1 - UC4

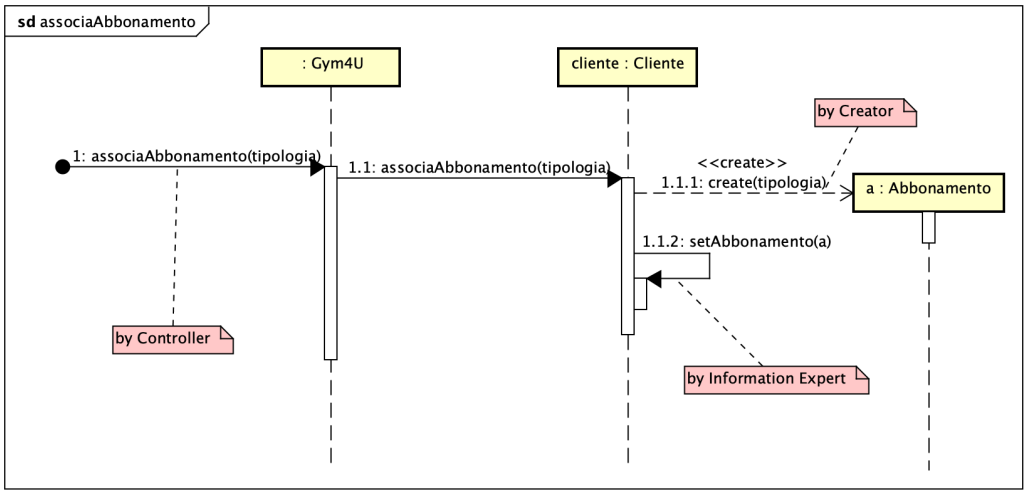
- nuovoCliente



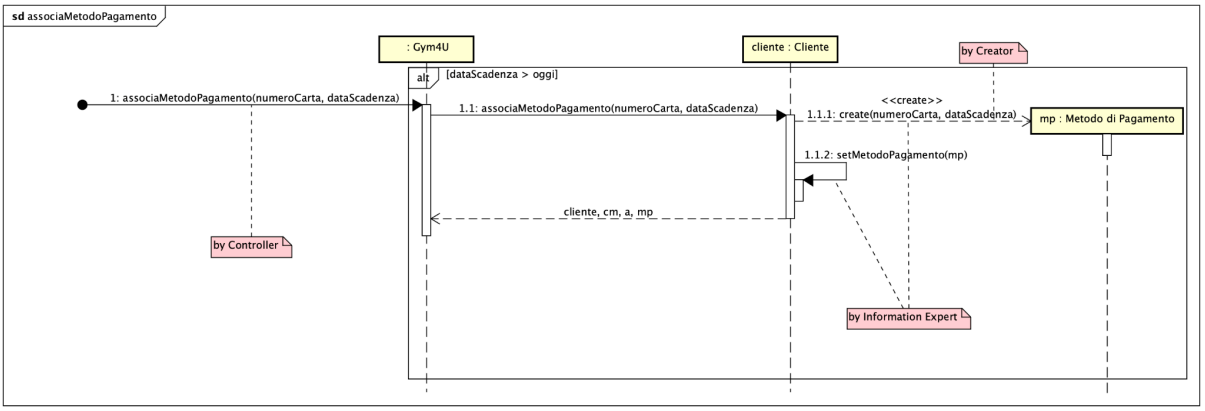
- associaCertificatoMedico



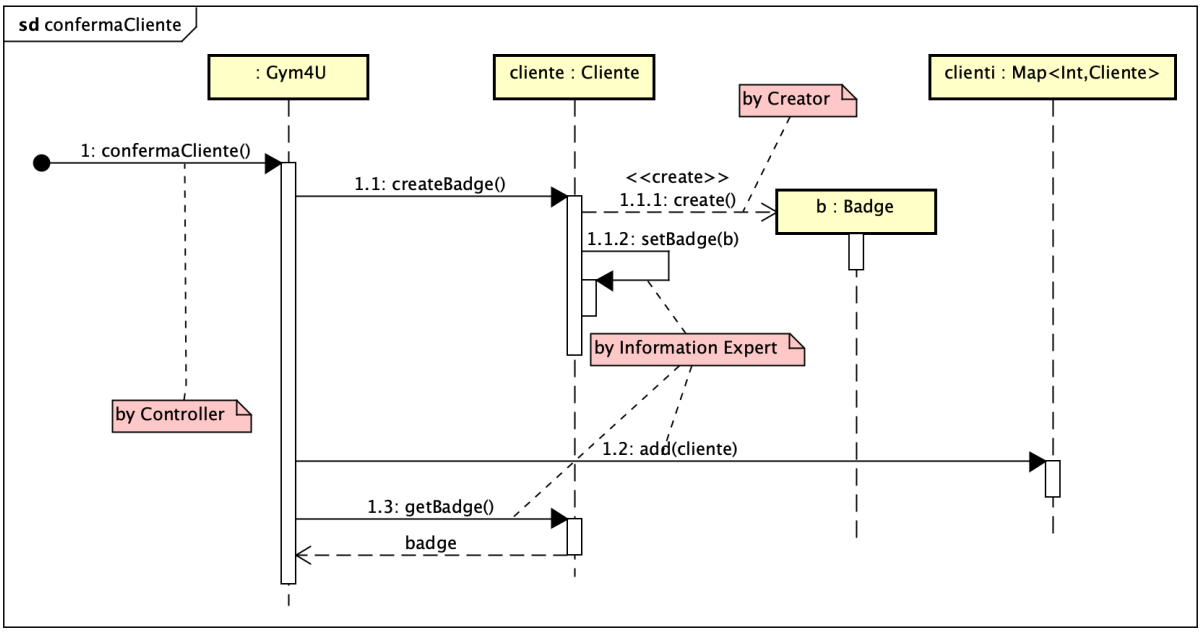

- associaAbbonamento




- associaMetodoPagamento

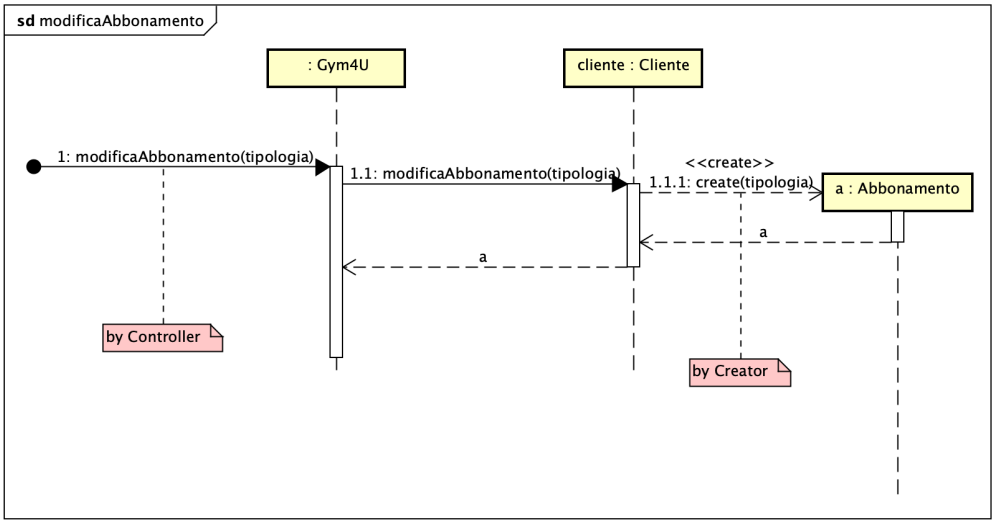


- confermaCliente

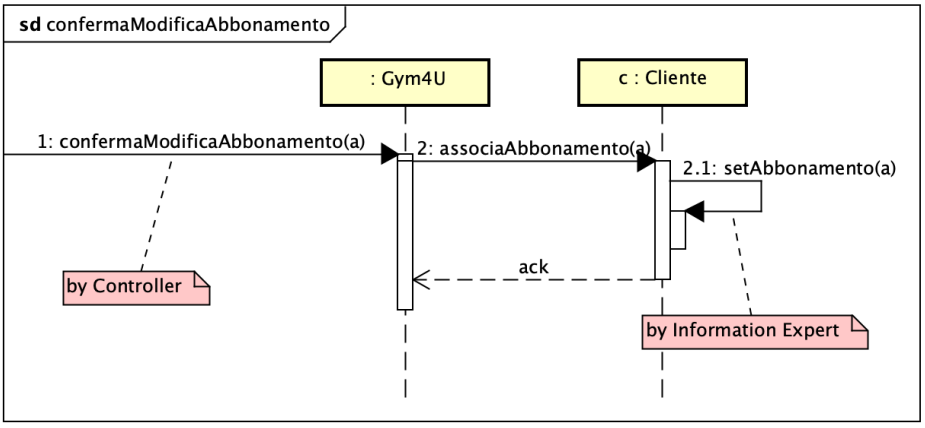



### 3.3.1.2 - UC5

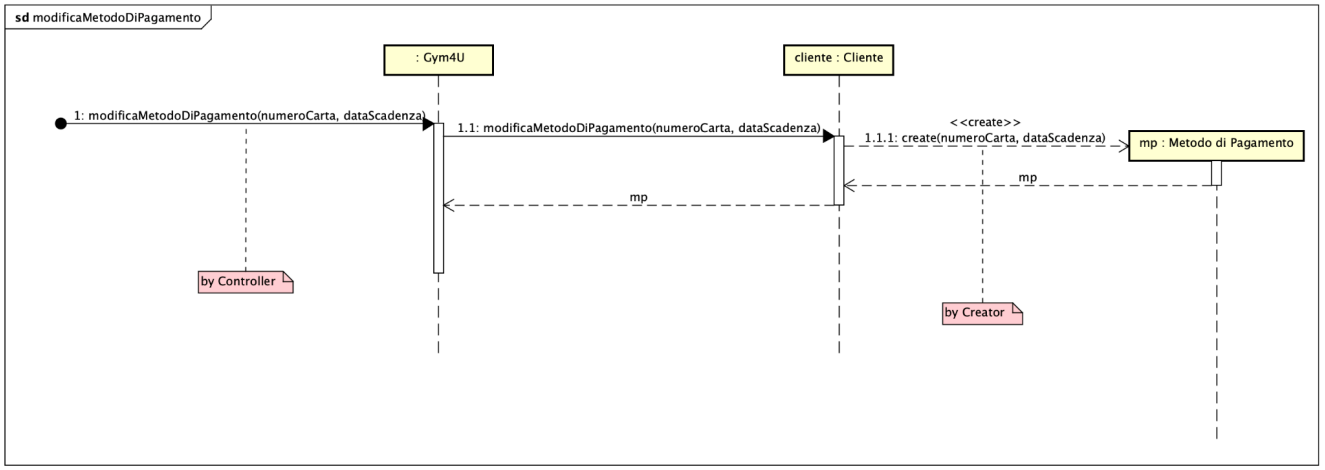
- modificaAbbonamento



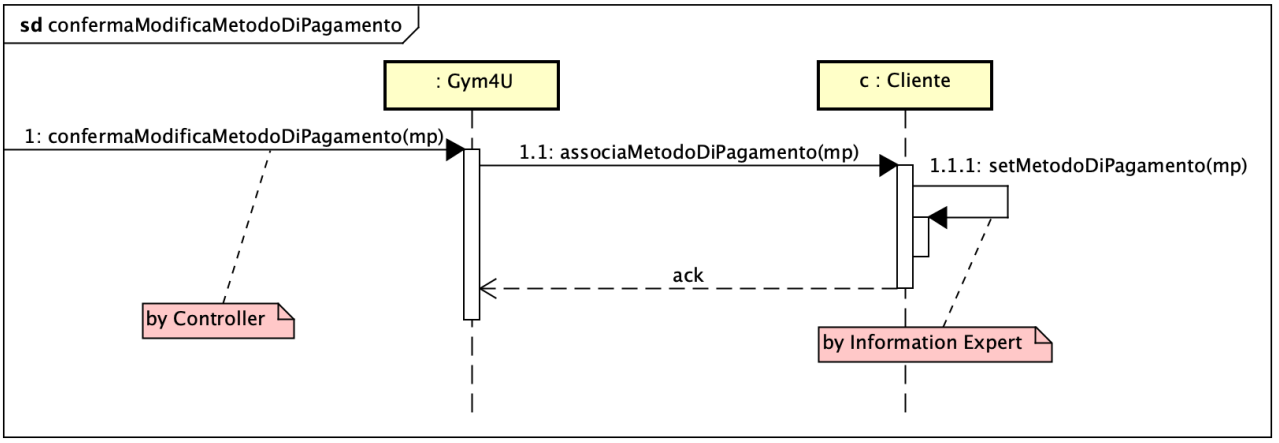

- confermaModificaAbbonamento



- modificaMetodoDiPagamento

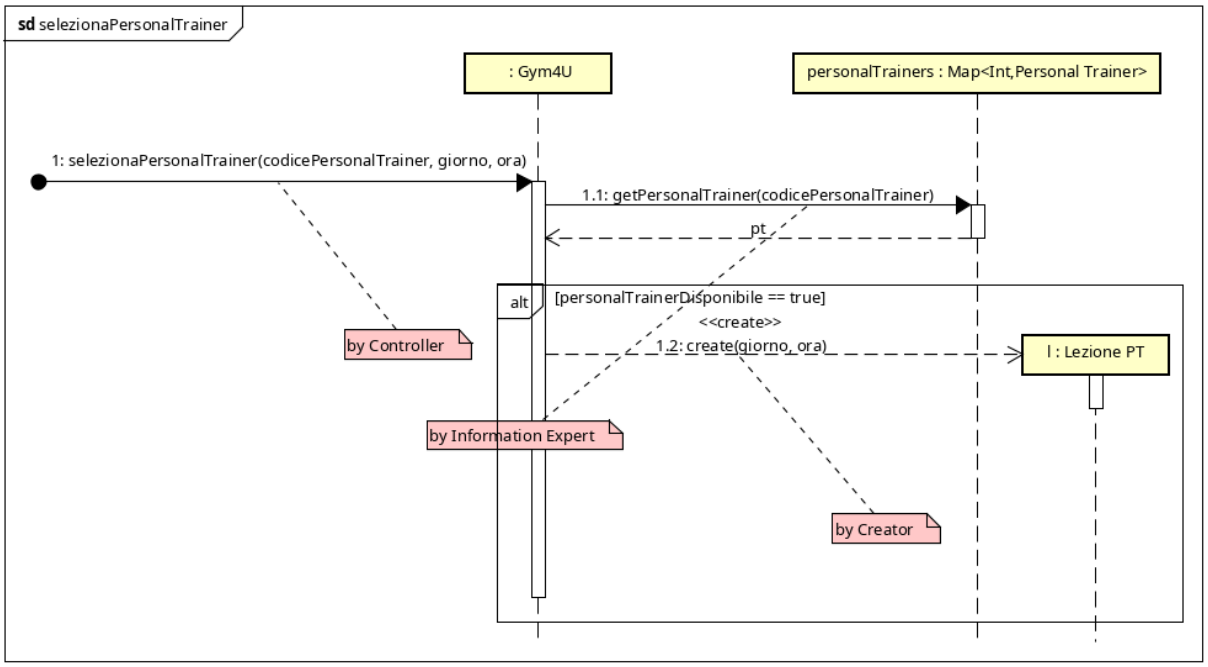



- confermaModificaMetodoDiPagamento

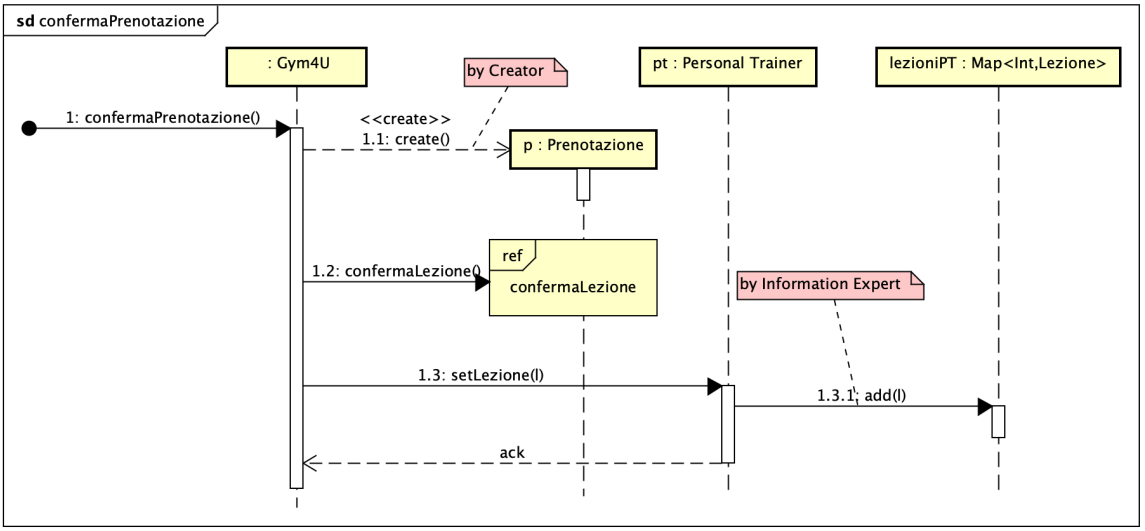


### 3.3.1.3 - UC6

- selezionaPersonalTrainer

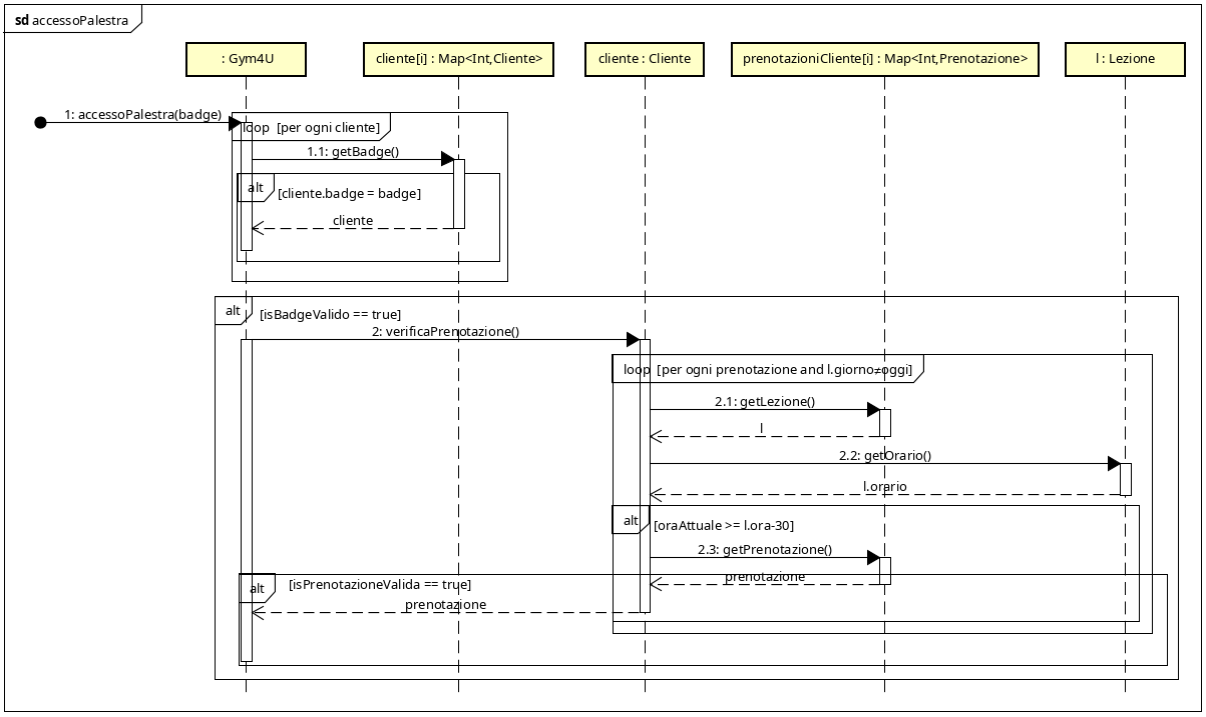



- confermaPrenotazione

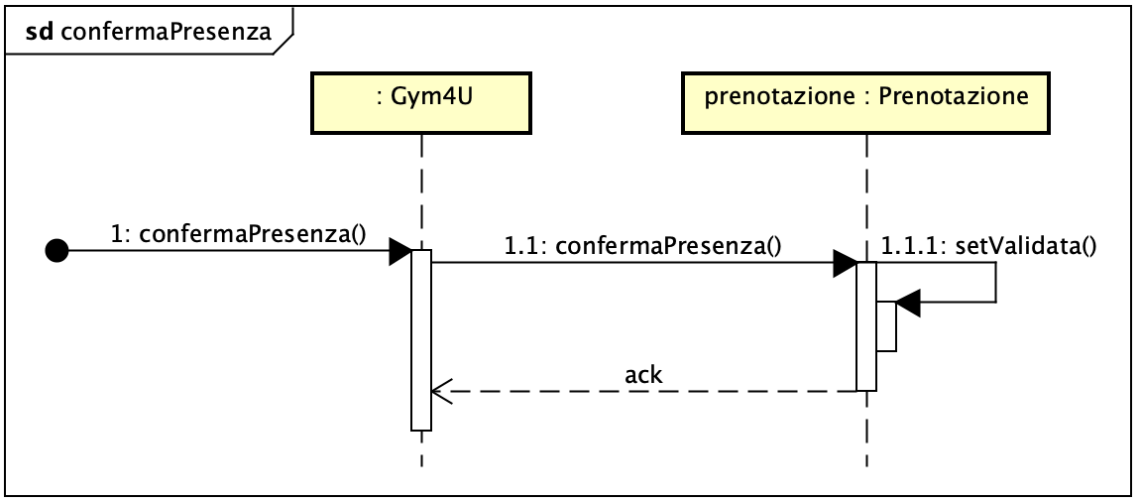


### 3.3.1.4 - UC10

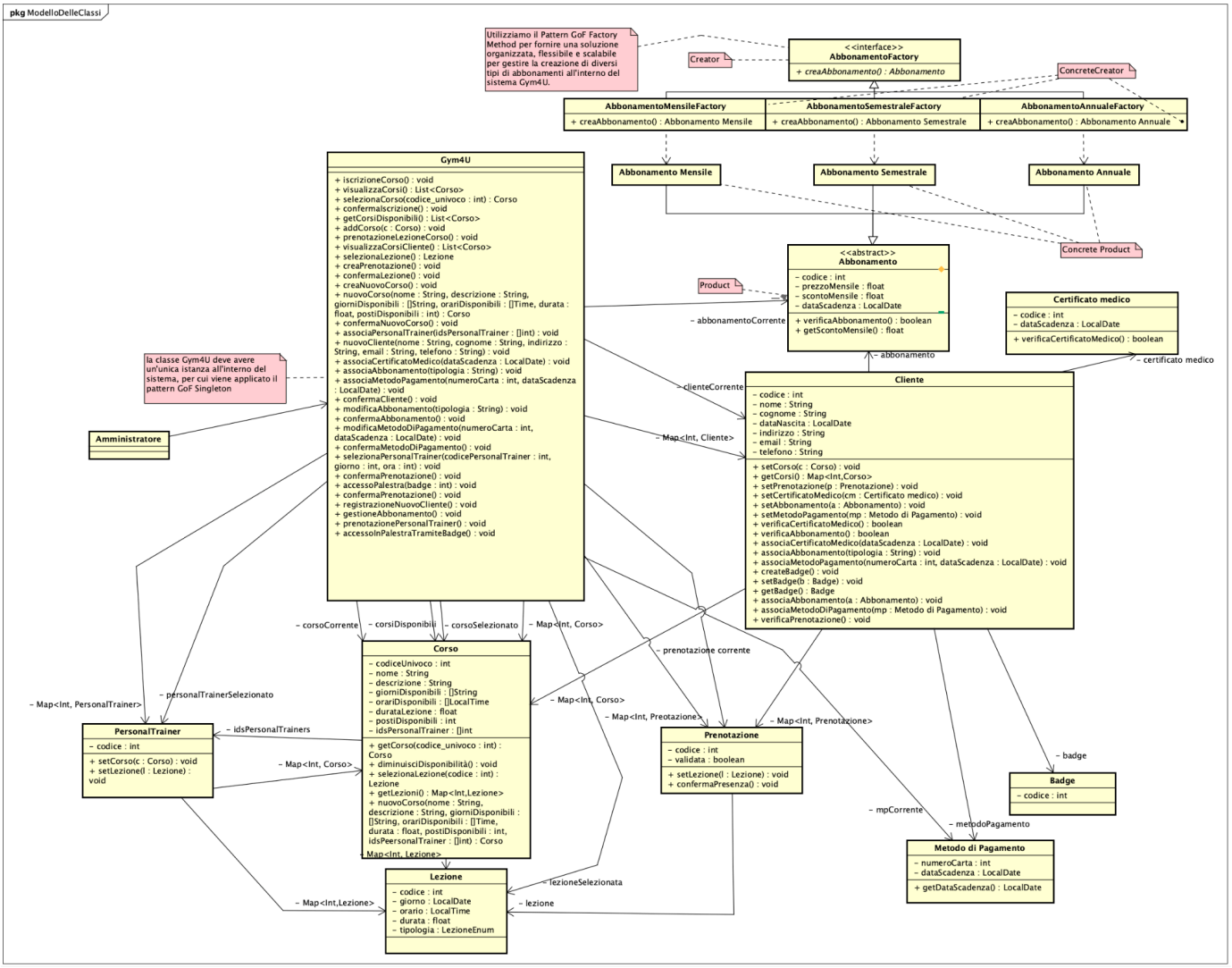
- accessoPalestra




- confermaPresenza



### 3.3.2 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.

### 3.4 Codice Prodotto e Test

Anche in questa seconda iterazione, la tipologia di test adottata nel nostro progetto è quella dei test unitari (Unit Test). Questi 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.

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.