AVERAGE
EMACS ENJOYER

PROGETTO
SISTEMA DI GESTIONE DI UNA PALESTRA

MEMBRI DEL GRUPPO:

Ardò Cosimo Andrea
D'adda Tommaso
Ferrari Pagini Alessandro
Lagae Nicole

SISTEMA DI GESTIONE DI UNA PALESTRA

Il presente documento costituisce la relazione finale del lavoro svolto per il pre-appello del corso di Analisi e Progettazione del Software. A partire dalle specifiche originali del file **Sistema di Gestione Palestra**, già modellato nei task precedenti, il progetto affronta l'estensione del sistema con l'introduzione di nuove funzionalità.

IMPORTANTE LA RELAZIONE CONTIENE SOLO L'ESTENSIONE E NON LE TASK ANTECEDENTI CHE TUTTAVIA SI TROVANO NEL VPP

Casi D’uso

In questa sezione vengono riportati i diagrammi dei casi d’uso relativi al sistema di gestione della palestra, comprensivi delle nuove funzionalità introdotte (merchandising, monitoraggio benessere, porta un amico e questionari).
La prima sezione offre un diagramma completo dei casi d’uso del sistema, mentre nella seconda vengono riportati i diagrammi di dettaglio per singoli casi d’uso selezionati.

Diagramma dei casi d'uso completo

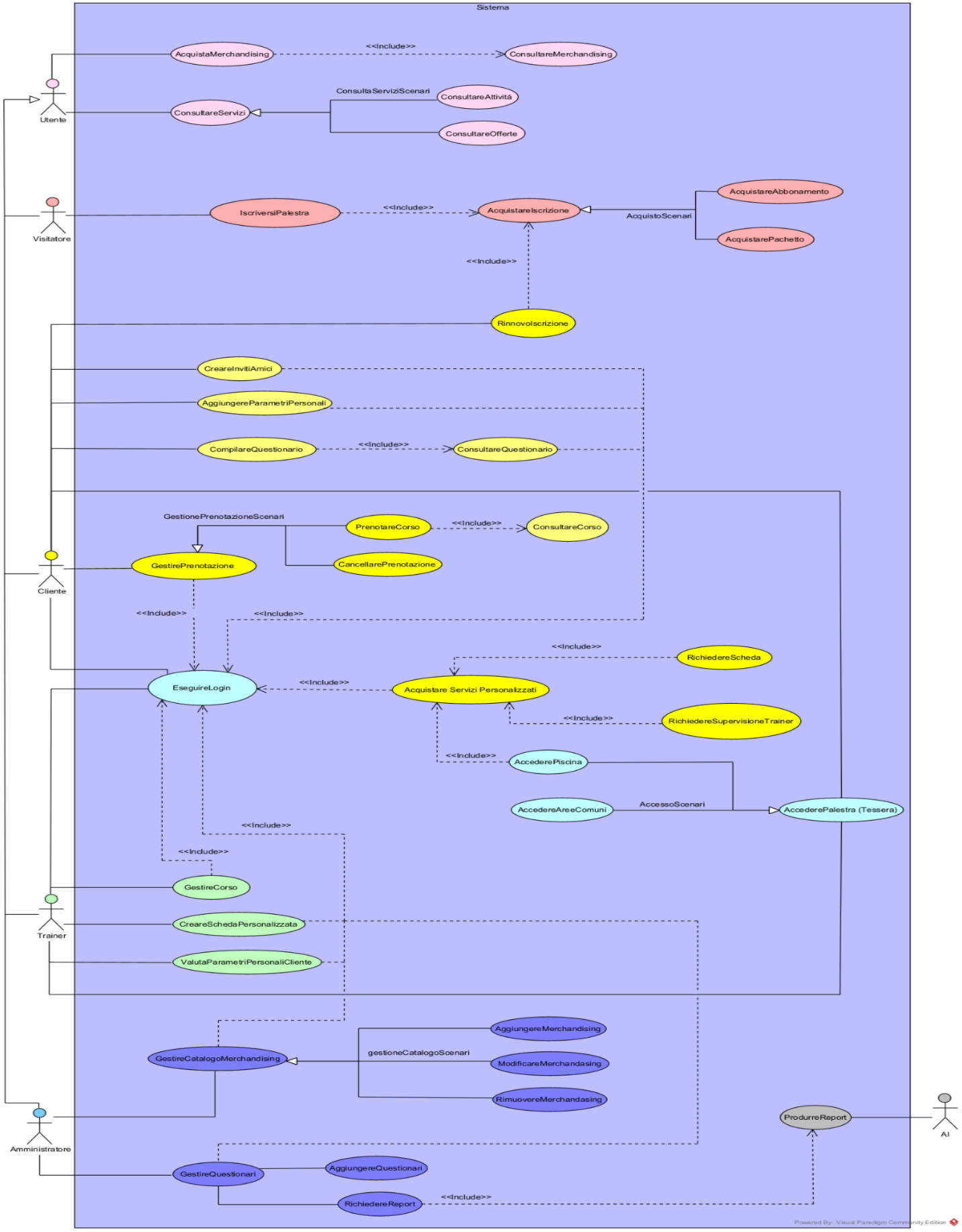
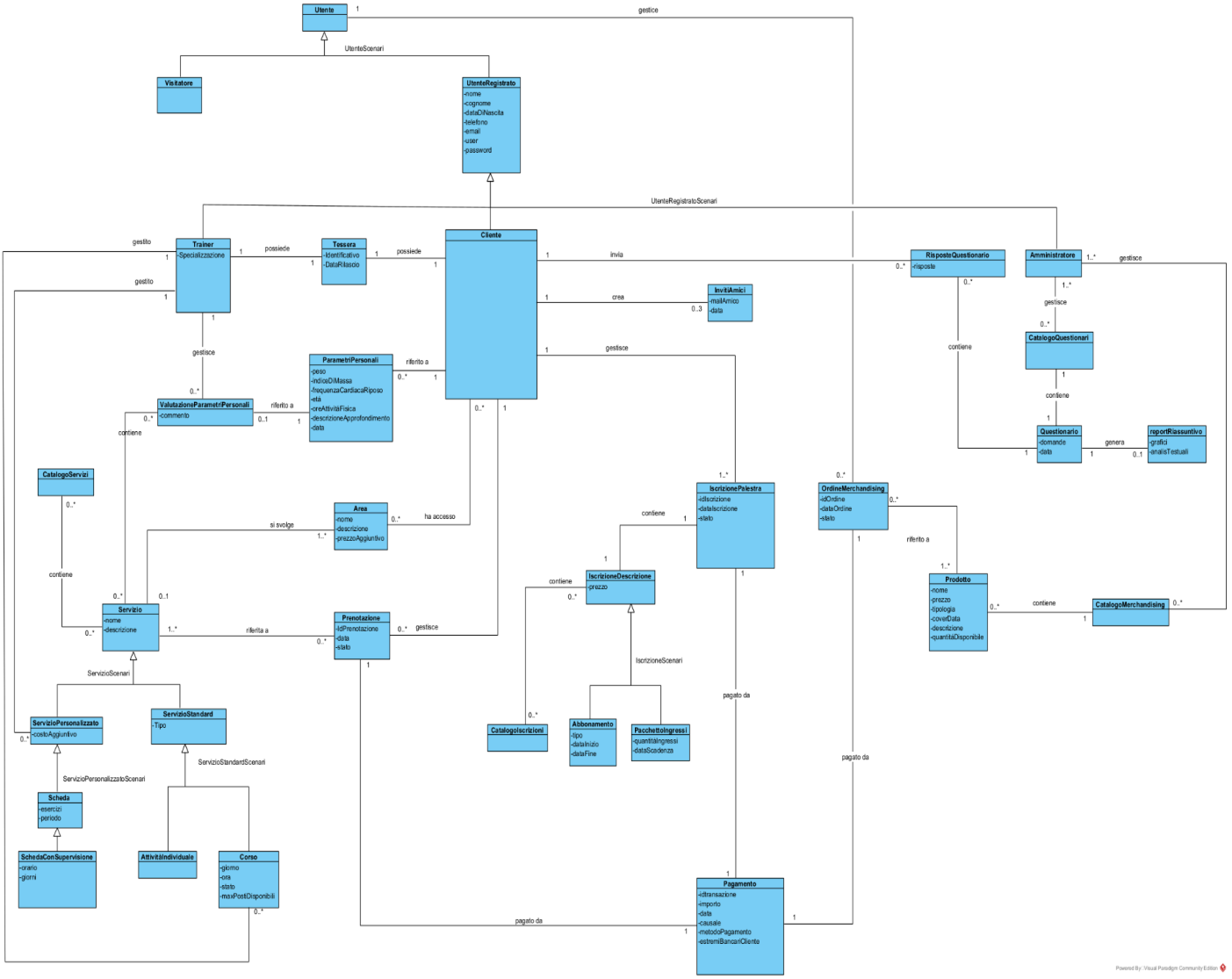


Diagramma di Dominio



Formato in dettaglio dei casi d'uso scelti

Caso d'Uso: AggiungereMerchandising

Campo	Descrizione
Nome caso d'uso	AggiungereMerchandising
Portata	Gestione palestra
Livello	Obbiettivo utente - Livello Amministratore di sistema
Attore primario	Amministratore
Parti interessate e interessi	- **Sistema**: deve registrare correttamente il prodotto con tutti i suoi dati e renderlo visibile agli utenti - **Amministratore**: desidera inserire nel sistema i prodotti di merchandising per essere acquistati da un utente
Pre-condizioni	- L'Amministratore è autenticato e autorizzato all'inserimento - Il sistema è operativo e accessibile
Garanzia di successo	- Il nuovo prodotto è stato aggiunto al catalogo - Gli utenti possono visualizzarlo e acquistarlo
Scenario principale di successo	1. L'amministratore accede al sistema 2. Il sistema permette l'inserimento di un nuovo prodotto. 3. L'amministratore inserisce il prodotto 4. Il sistema verifica che il prodotto non sia attualmente presente nel catalogo 5. Il sistema registra il prodotto e invia una conferma di inserimento
Estensione	- **4a** Il Prodotto è già stato caricato(fallimento) - **5a** Il Sistema fallisce nel caricamento del prodotto
Requisiti speciali	- Il Sistema deve essere responsivo e performante
Elenco delle variabili tecnologiche e dati	- Interfaccia grafica da cui poter gestire il sistema e aggiungere i prodotti
Frequenza di ripetizione	- Il sistema deve garantire la possibilitá di inserimento dei prodotti quando ritenuto opportuno dall'amministratore

Caso d'Uso: AggiungereParametriPersonali

Campo	Descrizione
Nome Del Caso d'Uso	AggiungereParametriPersonali
Portata	Gestione Palestra
Livello	Obbiettivo utente
Attore Primario	Cliente
Parti Interessate e Interessi	- **Cliente**: Inserisce i suoi dati personali nel sistema
Pre-condizioni	- Il Cliente deve avere un'iscrizione attiva in palestra
Garanzia di successo	- Il Cliente potrá vedere i suo parametri personali aggiunti nel sistema e successivamente riceverà un commento e una lista di possibili attività da fare in palestra in seguito ai suoi dati personali inseriti
Scenario principale di successo	1. Il Cliente accede al Sistema 2. Il Cliente carica i dati 3. Il Sistema controlla che il Cliente non abbia superato il limite mensile di inserimento 4. Il Sistema conferma l'inserimento
Estensioni	- **1a** Il Cliente ha superato il limite mensile di caricamento (fallimento)
Requisiti speciali	- Aggiornamento database ogni volta che Cliente carica dati
Elenco variabili tecnologiche e dati	- Interfaccia grafica per il cliente per poter caricare i dati e ricevere le valutazioni
Frequenza di ripetizione	- Mensile con possibilità di interruzione in qualsiasi momento

Caso d'Uso: CompilareQuestionario

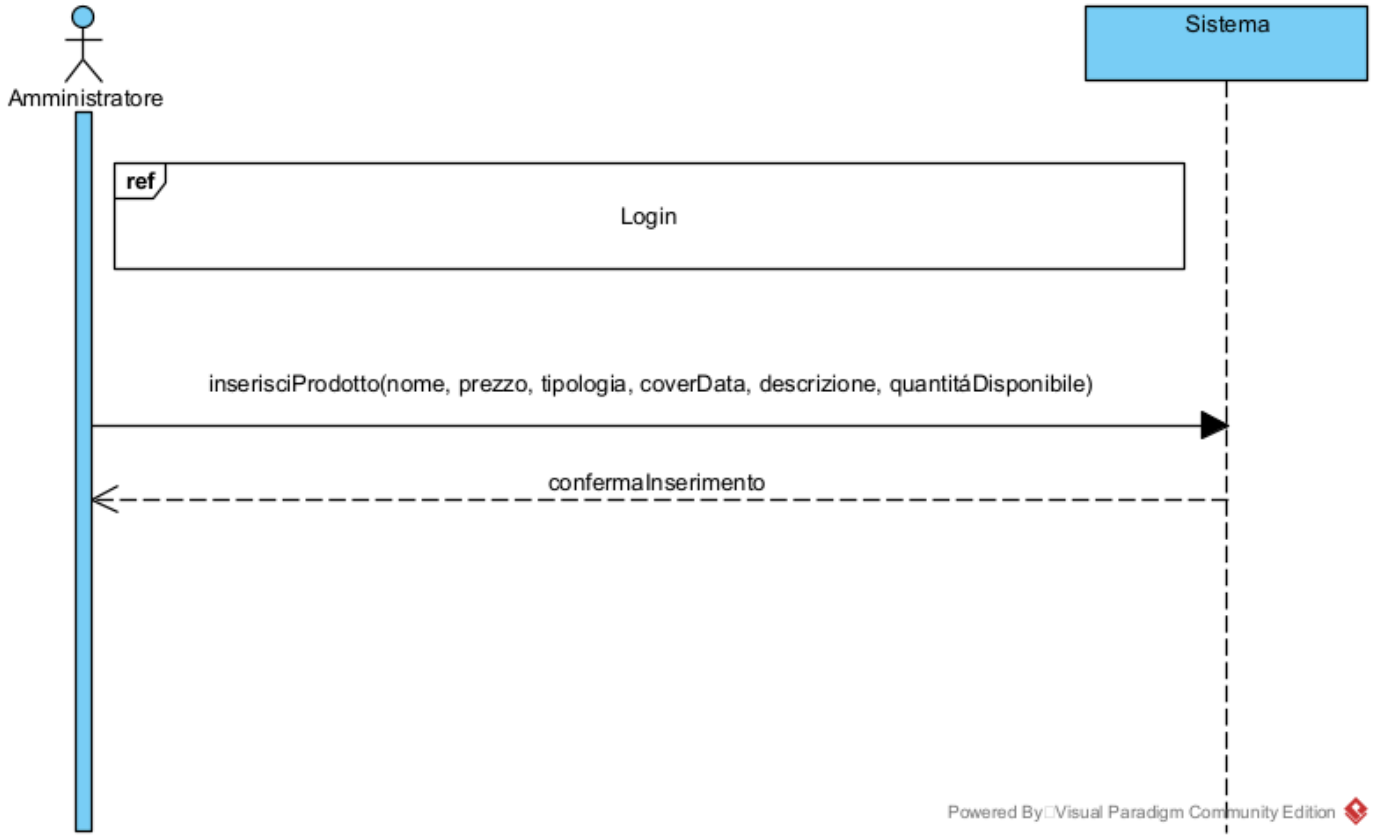
Campo	Descrizione
Nome caso d'uso	CompilareQuestionario
Portata	Gestione palestra
Livello	Obbiettivo utente
Attore primario	Cliente
Parti interessate e interessi	- **Cliente**: Una volta ricevuto il questionario può decidere se compilarlo
Pre-condizioni	- L'amministratore ha inviato un questionario - Il cliente accede al sistema - Il cliente riceve il questionario
Garanzia di successo	- Il cliente compila ed invia il questionario
Scenario principale di successo	1. Il Cliente accede al sistema 2. Il Cliente compila il questionario 3. Il sistema registra le informazioni 4. Il client riceve lo sconto
Estensione	- **3a** Non tutti i campi obbligatori sono stati compilati (compila dati mancanti) - **4a** I dati non sono stati caricati (fallimento)
Requisiti speciali	- Il cliente riceve lo sconto del 5% - Il sistema deve essere accessibile in tempo reale da dispositivi mobili e desktop
Elenco delle variabili tecnologiche e dati	- Interfaccia utente: inserimento dati nel questionario. - Output conferma: Conferma dati inseriti
Frequenza di ripetizione	- Ogni volta che l'amministratore invia un questionario

Caso d'Uso: ValutareParametriPersonaliCliente

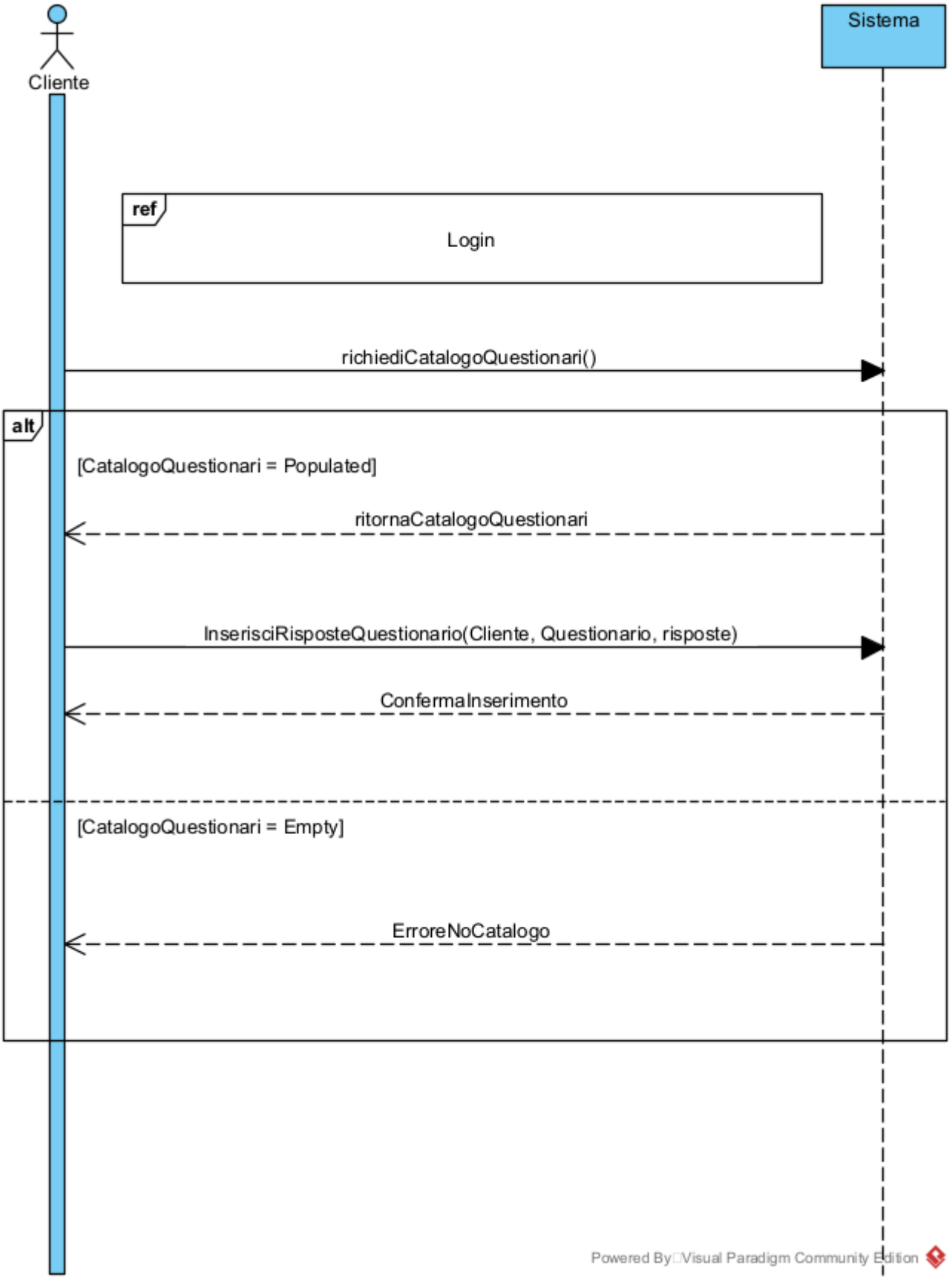
Campo	Descrizione
Nome Del Caso d'Uso	ValutareParametriPersonaliCliente
Portata	Gestione Palestra
Livello	Obbiettivo utente
Attore Primario	Trainer
Parti Interessate e Interessi	- **Trainer**: accede ai dati caricati dal Cliente e scrive commento complessivo e lista di attività suggerite
Pre-condizioni	- Il Cliente deve essere iscritto alla palestra - Il Cliente deve aver compilato i parametri personali mensili relativi al proprio stato di salute e attività fisica svolta (peso, indice di massa corporea, frequenza cardiaca a riposo ed eventuali descrizioni di approfondimento)
Garanzia di successo	- La valutazione viene caricata nel sistema - Il cliente riceve la valutazione
Scenario principale di successo	1. Il Trainer Richiede i ParametriPersonali Del Cliente 2. Il Trainer inserisce la valutazione sul parametro personale di interesse 3. Il Sistema inserisce i parametri personali nel database e invia la conferma di inserimento
Estensioni	- **3a** Il Sistema Fallisce nell'inserimento dei dati
Requisiti speciali	- Aggiornamento database ogni volta che Trainer carica i dati
Elenco variabili tecnologiche e dati	- Il Trainer ha un'interfaccia per ricevere, commentare i dati e caricarli
Frequenza di ripetizione	- Ogni volta che ci sono dei parametri personali disponibili da valutare

Diagramma di sequenza di sistema (SSD)

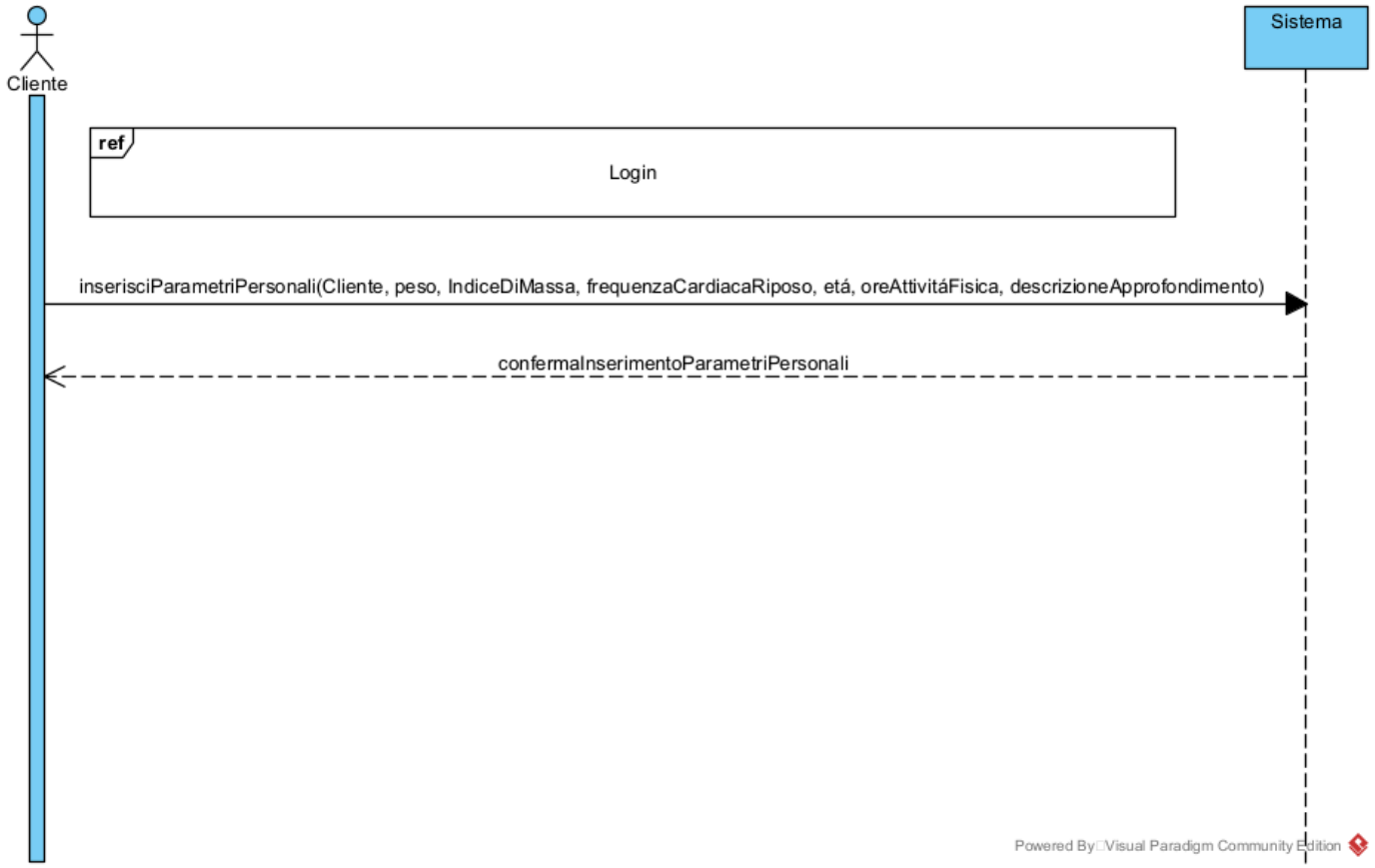
AggiungereMerchandising



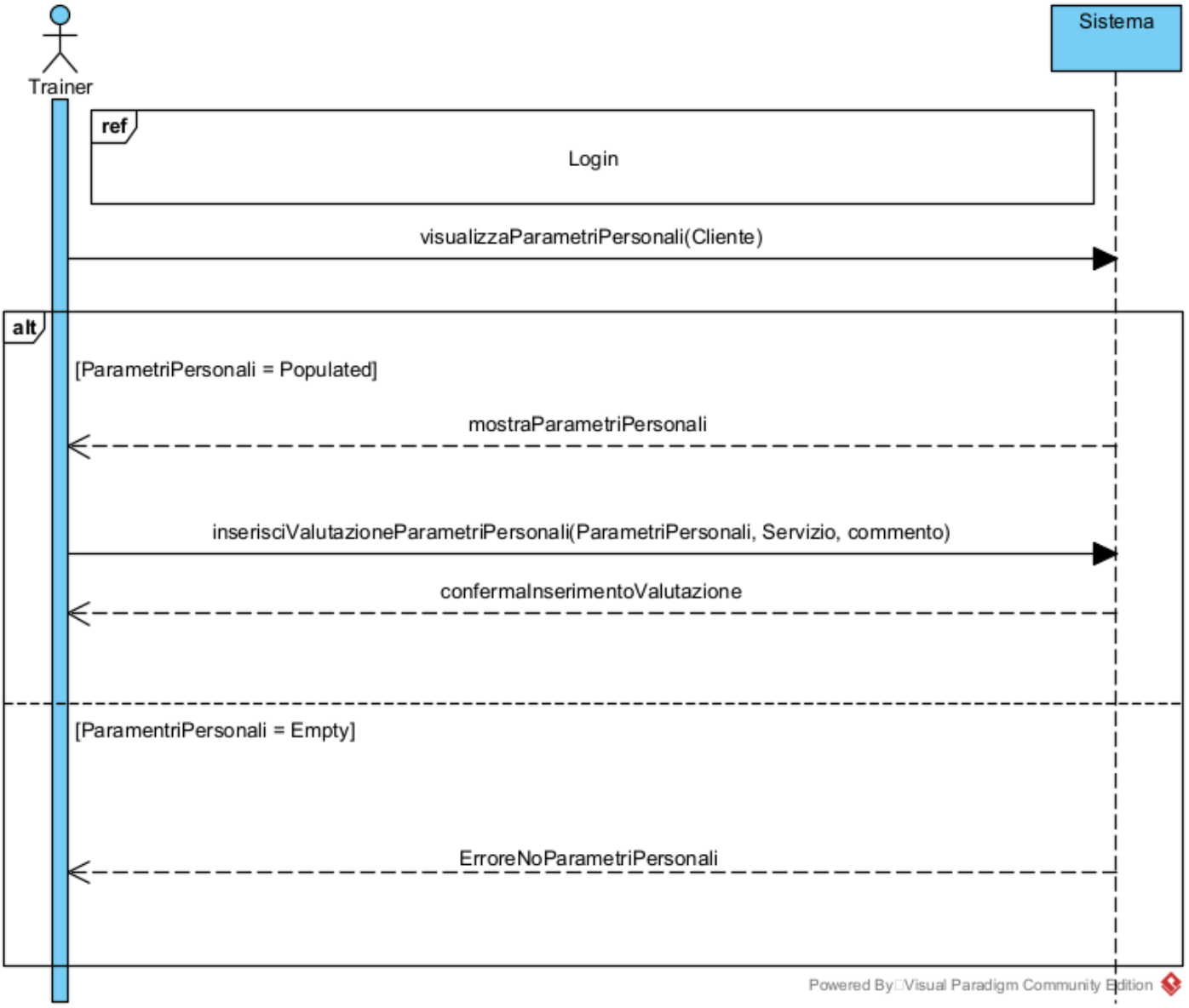
CompilareQuestionario



AggiungereParametriPersonali



ValutareParametriPersonaliCliente



contratti

Contratto: InserisciParametri

Campo	Descrizione
Operazione	InserisciParametri(cliente: Cliente, peso: Float, indiceDiMassa: String, frequenzaCardiacaRiposo: String, età: Int, OreAttivitàFisica: Int, descrizioneApprofondimento: String)
Riferimenti	Use Cases: AggiungereParametriPersonali

Pre-condizioni

- Il Cliente possiede un'iscrizione attiva presso la palestra.
- Esiste un'interfaccia per inserire i dati.

Post-condizioni

- Controllo della data dell'ultimo inserimento = `controllaDisponibilitàInserimentoMensile(Cliente, Date)`.
- È stato creato un nuovo oggetto ParametriPersonali con i parametri forniti = `creaParametroPersonale(Cliente, peso, indiceDiMassa, frequenzaCardiacaRiposo, età, oreAttivitàFisica, descrizioneApprofondimento)`.
- Il nuovo oggetto ParametriPersonali è associato al Cliente = `aggiungiParametriPersonali(ParametriPersonali)`.
- I parametri inseriti sono stati salvati nel sistema.
- Il sistema ha confermato l'inserimento tramite la funzione = `ConfermaInserimentoParametriPersonali()`.
- Se ci sono errori nell'inserimento, il sistema invia un messaggio di errore tramite = `ErroreInserimentoParametriPersonali()`.

Contratto: InserisciValutazioneParametriPersonaliCliente

Campo	Descrizione
Operazione	InserisciValutazioneParametriPersonaliCliente(parametri: ParametriPersonali, serviziConsigliati: Servizio[], commento: String)
Riferimenti	Use Cases: ValutaParametriPersonaliCliente

Pre-condizioni

- Il Cliente possiede un'iscrizione attiva presso la palestra.
- Il Cliente ha già compilato e caricato i parametri personali mensili.
- Il Trainer è autenticato nel sistema.

Post-condizioni

- È stata creata una nuova istanza di ValutazioneParametriPersonali con i dati forniti = `creaValutazioneParametriPersonali(Servizio, commento)`.
- L'oggetto ValutazioneParametriPersonali è stato associato ai ParametriPersonali e al Servizio = `SetValutazioniParametriPersonali(ValutazioneParametriPeronali)`.
- Il sistema ha confermato l'inserimento della valutazione tramite la funzione = `ConfermaInserimentoValutazionePersonale()`.
- Il sistema ha registrato il commento associato alla valutazione.

Contratto: InserisciRisposteQuestionario

Campo	Descrizione
Operazione	InserisciRisposteQuestionario(cliente: Cliente, questionario: Questionario, risposte: Risposte[])
Riferimenti	Use Cases: CompilareQuestionario

Pre-condizioni

- Il Cliente `cliente` è registrato nel sistema e ha accesso al questionario.
- Il Questionario `questionario` esiste nel sistema.
- Le risposte `risposta` devono essere fornite per tutte le domande chiuse del questionario.

Post-condizioni

- È stata creata una nuova istanza di risposte al questionario associata al Cliente `cliente` e al Questionario `questionario` = `creaRisposteQuestionario(Cliente cliente, Risposte risposte)`.
- Le risposte fornite sono state associate al Questionario = `ritornaRisposteQuestionario`.
- La nuova risposta al questionario è stata aggiunta al sistema attraverso la funzione = `aggiungiRisposteQuestionario(RispostaQuestionario)`.
- Il sistema ha confermato l'aggiunta delle risposte attraverso la funzione = `ConfermaAggiuntaRisposte`.

Contratto: inserisciProdotto

Campo	Descrizione
Operazione	inserisciProdotto(nome: String, prezzo: Float, tipologia: String, coverData: String, descrizione: String, quantitàDisponibile: Integer)
Riferimenti	Use Cases: GestireCatalogoMerchandising

Pre-condizioni

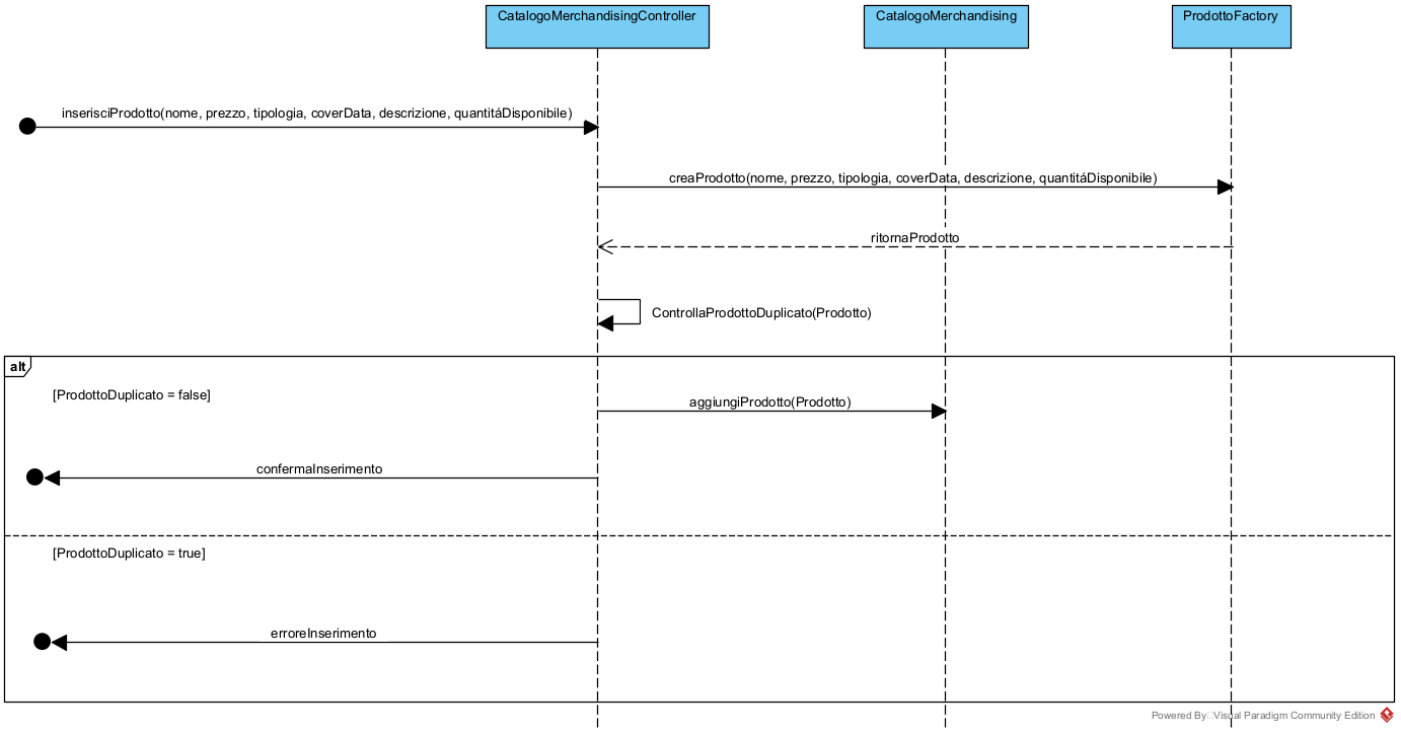
- Il CatalogoMerchandising esiste e l'amministratore ne ha accesso tramite il sistema.
- I dati per il nuovo prodotto (nome, prezzo, tipologia, ecc.) sono validi e completi.
- La quantità disponibile deve essere un numero positivo.

Post-condizioni

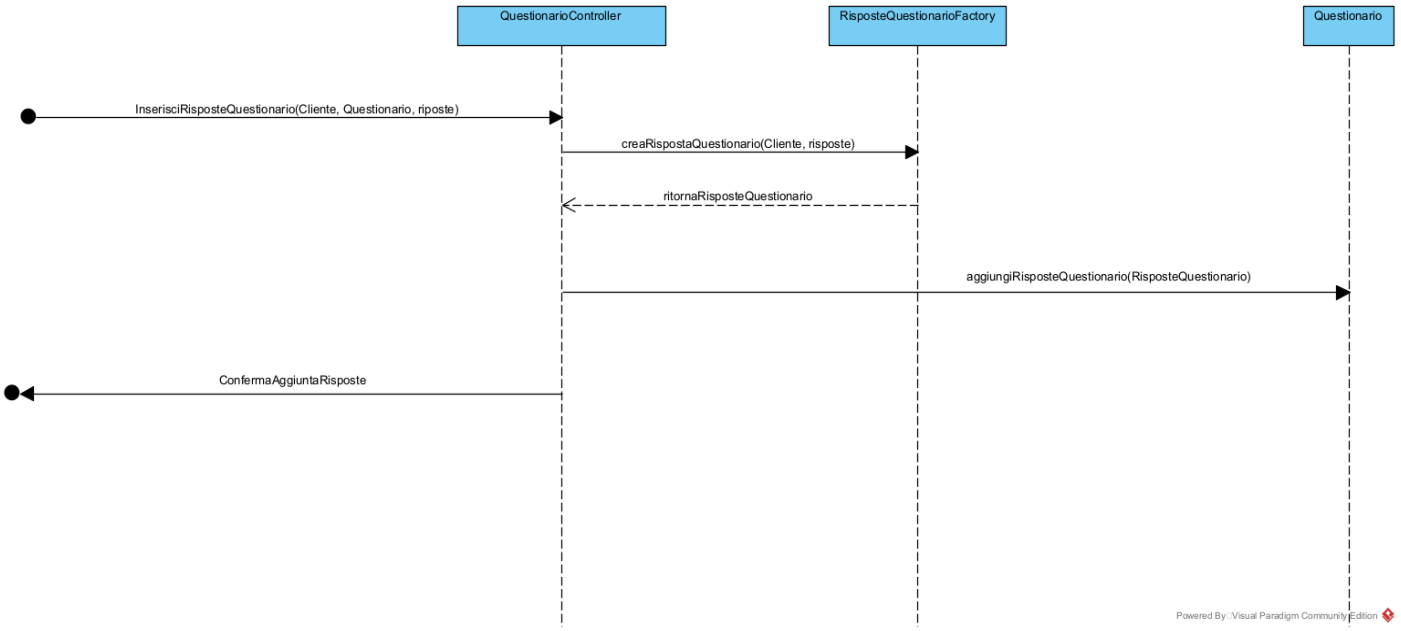
- È stato creato un nuovo oggetto Prodotto con i dettagli forniti = `creaProdotto`.
- Il Prodotto è stato aggiunto al CatalogoMerchandising = `aggiungiProdotto(Prodotto)`.
- Il catalogo è stato aggiornato per includere il nuovo prodotto, con la relativa quantità disponibile.

Diagramma di sistema (SD)

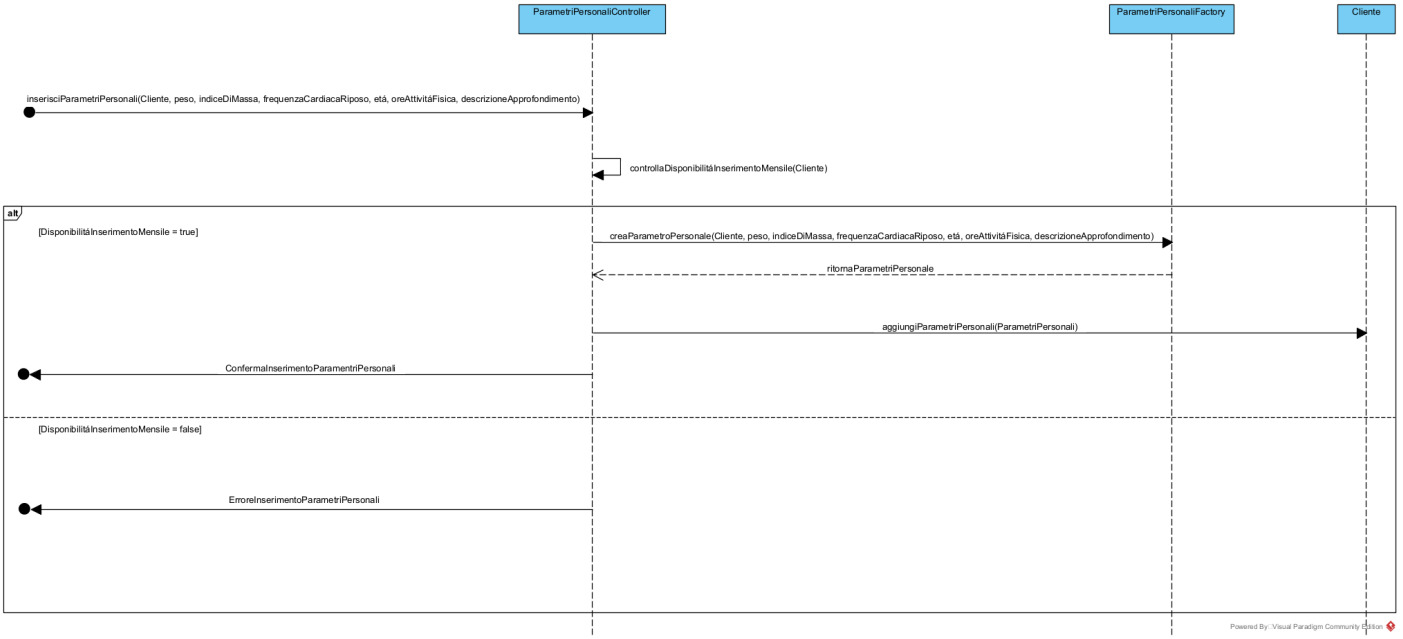
InserisciProdotto



InserisciRisposteQuestionario



InserisciParametri



InserisciValutazioneParametriPersonaliCliente

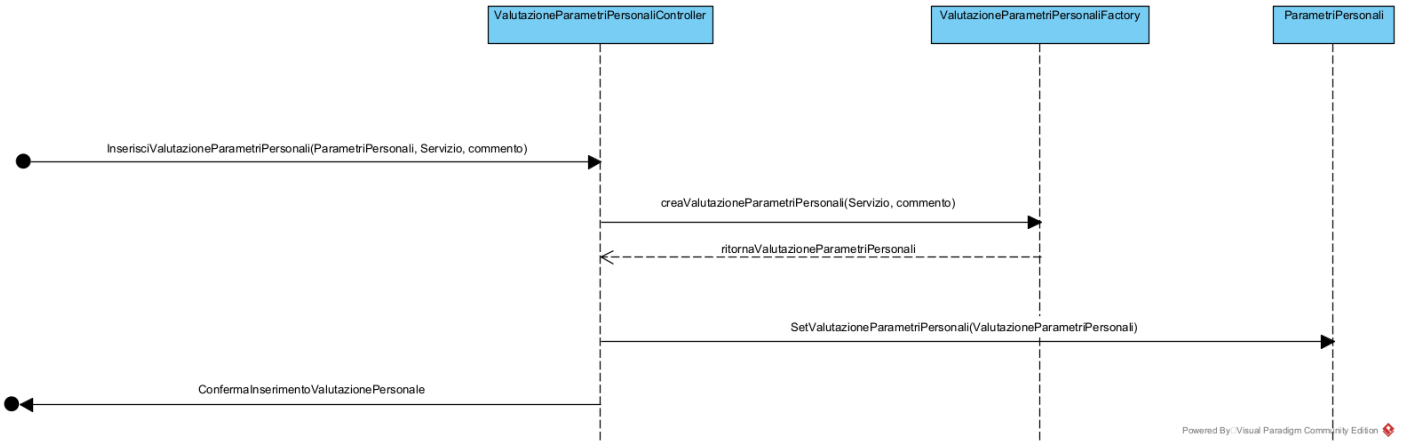


Diagramma delle classi di progetto

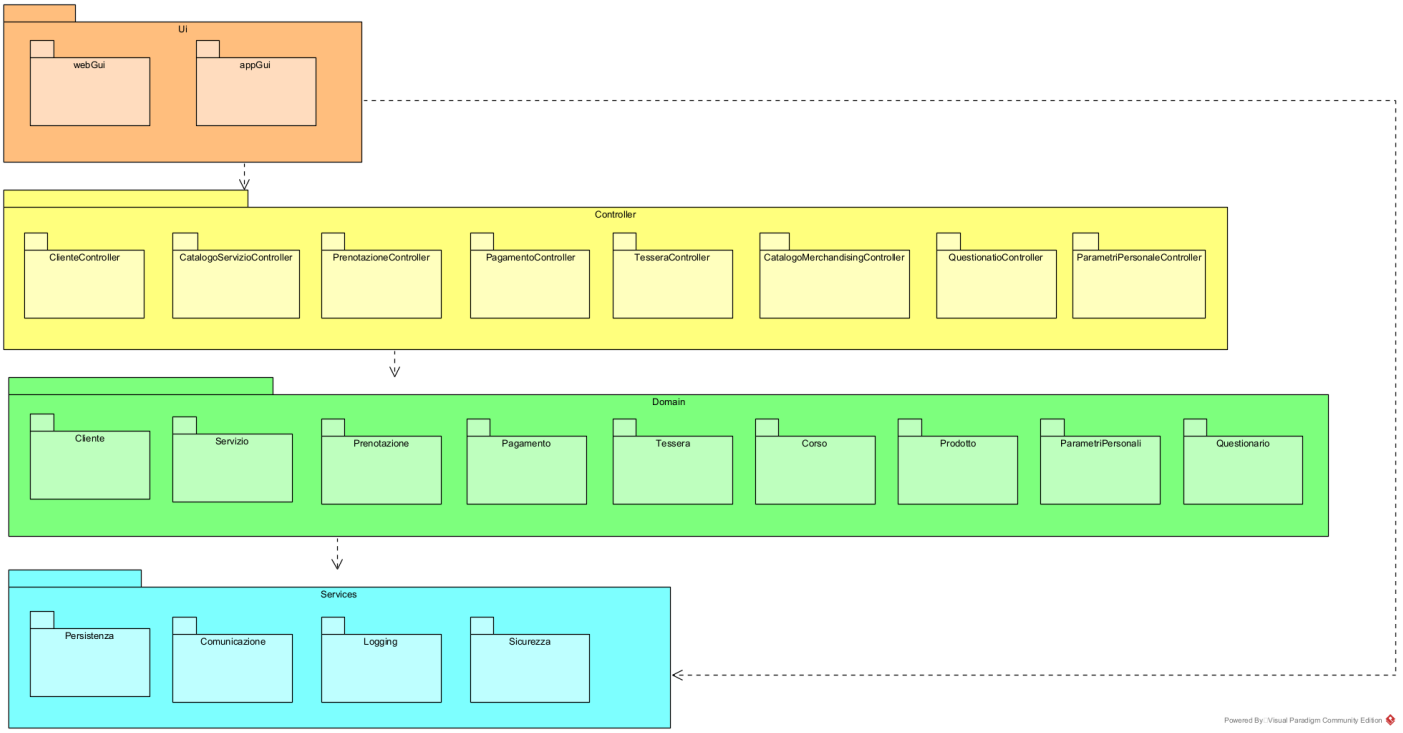


Diagramma Classe Prodotti

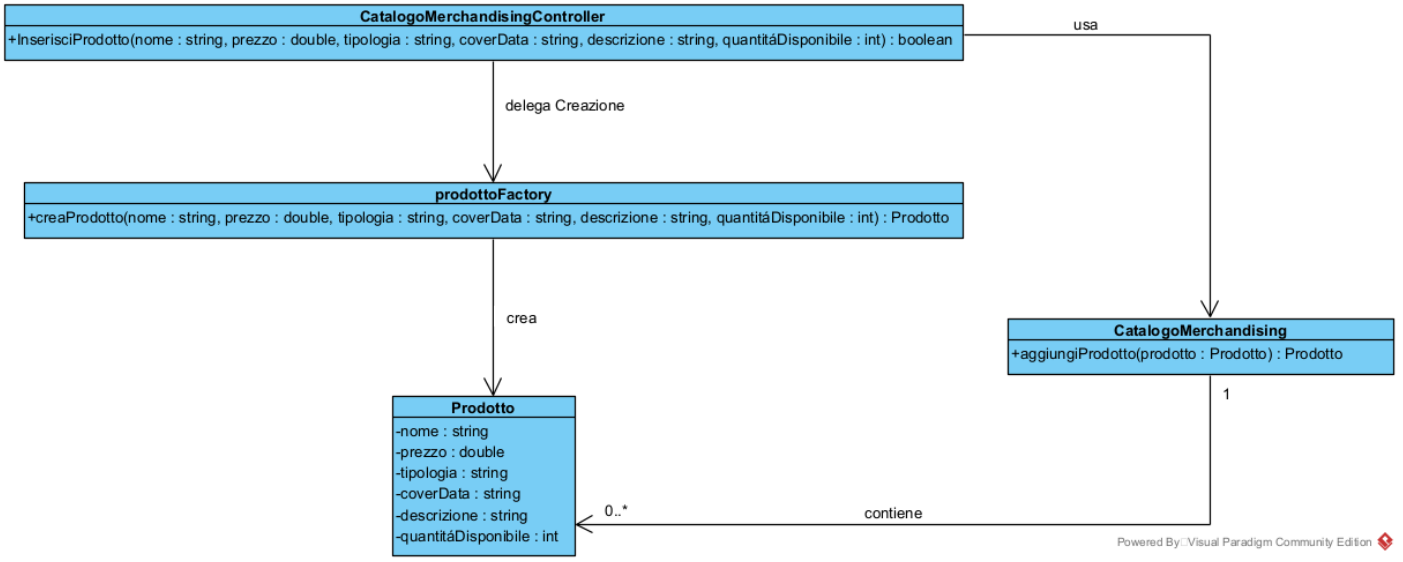


Diagramma Classe Parametri Personali

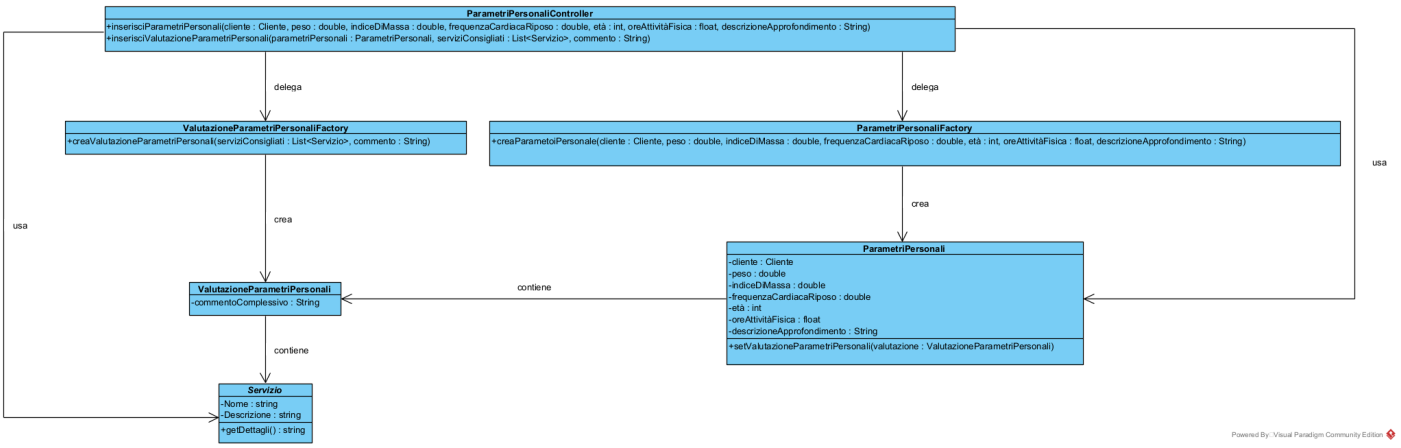
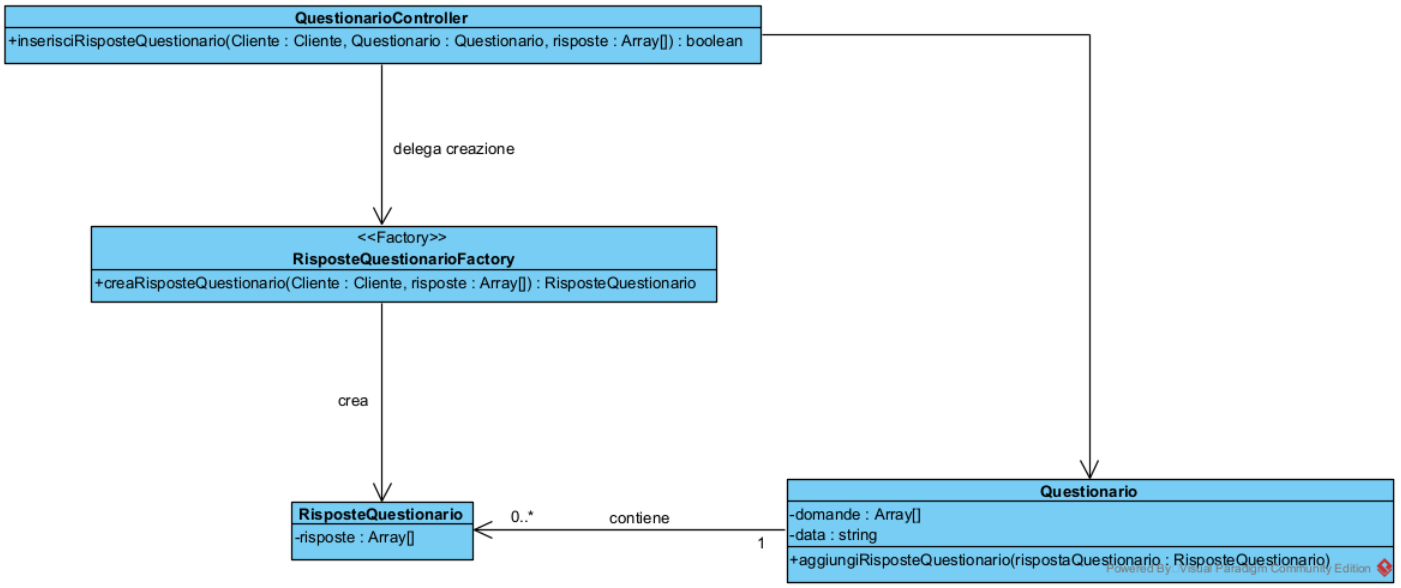


Diagramma Classe Questionario



Pattern GRASP Identificati nel Diagramma Questionario

1. Controller

La classe `QuestionarioController` implementa il pattern **Controller**, fungendo da coordinatore principale che:

- Gestisce le richieste dall'interfaccia utente per l'inserimento delle risposte al questionario
- Coordina le operazioni tra Cliente, Questionario e le risposte
- Delega la creazione delle risposte questionario alla factory appropriata

2. Creator

Il pattern **Creator** è chiaramente visibile nelle relazioni "crea" e "delega creazione":

- `QuestionarioController` delega la creazione alla `RisposteQuestionarioFactory`
- `RisposteQuestionarioFactory` crea istanze di `RisposteQuestionario`
- La factory è responsabile della creazione centralizzata degli oggetti RisposteQuestionario

3. Information Expert

- `RisposteQuestionario` contiene l'array delle risposte ed è l'esperto delle informazioni sulle risposte
- `Questionario` contiene le domande (Array[]) e la data, ed è l'esperto delle informazioni sul questionario stesso
- Ogni classe mantiene e manipola le informazioni di cui è responsabile

4. Low Coupling

Design contiene **accoppiamento ridotto**:

- Il Controller non crea direttamente le risposte ma delega alla Factory
- La separazione tra Factory e Controller riduce le dipendenze dirette
- Il Questionario mantiene una relazione di composizione con le RisposteQuestionario

5. High Cohesion

Ogni classe ha una **responsabilità ben definita**:

- `QuestionarioController`: controllo del flusso applicativo per la gestione questionari
- `RisposteQuestionarioFactory`: creazione specializzata delle risposte
- `RisposteQuestionario`: gestione delle risposte fornite
- `Questionario`: gestione delle domande e metadati del questionario

6. Polymorphism/Indirection

L'uso della Factory class implementa il pattern di **Indirection**:

- Il Controller non dipende direttamente dalla classe RisposteQuestionario
- La Factory astrae il processo di creazione delle risposte
- Permette variazioni future nell'implementazione senza modificare il Controller

7. Pure Fabrication

La `RisposteQuestionarioFactory` rappresenta il pattern **Pure Fabrication**:

- Non rappresenta un concetto del dominio problema
- È una classe di supporto creata per mantenere alta coesione e basso accoppiamento
- Centralizza la logica di creazione delle risposte

Pattern GRASP Identificati nel Diagramma Prodotti

1. Controller

La classe `CatalogoMerchandisingController` implementa il pattern **Controller**, fungendo da coordinatore principale che:

- Gestisce le richieste dall'interfaccia utente per l'inserimento di prodotti
- Coordina le operazioni tra le diverse classi del sistema
- Delega la creazione di prodotti alla factory appropriata

2. Creator

Il pattern **Creator** è chiaramente visibile nelle relazioni "crea" e "delega Creazione":

- `CatalogoMerchandisingController` delega la creazione alla `prodottoFactory`
- `prodottoFactory` crea istanze di `Prodotto`
- La factory è responsabile della creazione centralizzata degli oggetti Prodotto

3. Information Expert

- `Prodotto` contiene tutti gli attributi specifici (nome, prezzo, tipologia, coverData, descrizione, quantitàDisponibile) ed è l'esperto delle informazioni sui prodotti
- `CatalogoMerchandising` gestisce la collezione di prodotti e le operazioni di aggiunta
- Ogni classe mantiene e manipola le informazioni di cui è esperta

4. Low Coupling

Design contiene **accoppiamento ridotto**:

- Il Controller non crea direttamente i prodotti ma delega alla Factory
- La separazione tra Factory e Controller riduce le dipendenze
- Il catalogo mantiene una relazione di composizione pulita con i prodotti

5. High Cohesion

Ogni classe ha una **responsabilità ben definita**:

- `CatalogoMerchandisingController`: controllo del flusso applicativo
- `prodottoFactory`: creazione specializzata di prodotti
- `Prodotto`: rappresentazione e gestione dei dati del prodotto
- `CatalogoMerchandising`: gestione della collezione di prodotti

6. Polymorphism/Indirection

L'uso della Factory class implementa il pattern di **Indirection**:

- Il Controller non dipende direttamente dalla classe Prodotto
- La Factory astrae il processo di creazione
- Permette variazioni future nell'implementazione senza modificare il Controller

Pattern GRASP Identificati nel Diagramma ParametriPersonali

1. Controller

La classe `ParametriPersonaliController` implementa il pattern **Controller**, fungendo da coordinatore principale che:

- Gestisce le richieste dall'interfaccia utente
- Coordina le operazioni tra le diverse classi
- Delega le responsabilità specifiche alle classi appropriate

2. Creator

Il pattern **Creator** è visibile nelle relazioni "crea":

- `ParametriPersonaliController` crea istanze di `ValutazioneParametriPersonaliFactory`
- `ParametriPersonaliFactory` crea istanze di `ParametriPersonali`
- Ogni classe è responsabile della creazione degli oggetti che utilizza direttamente

3. Information Expert

- `ParametriPersonali` contiene tutti gli attributi specifici (cliente, peso, indici, età, ecc.) e presumibilmente i metodi per manipolarli
- `ValutazioneParametriPersonali` contiene il commento complessivo, essendo l'esperto delle informazioni di valutazione
- `Servizio` gestisce le proprie informazioni (nome, descrizione, dettagli)

4. Low Coupling

design con **accoppiamento ridotto**:

- Le classi sono ben separate con responsabilità distinte
- L'uso del pattern Factory riduce le dipendenze dirette
- Il Controller centralizza le dipendenze evitando connessioni multiple tra classi

5. High Cohesion

Ogni classe ha una **responsabilità ben definita**:

- `ParametriPersonali`: gestione dei dati personali del cliente
- `ValutazioneParametriPersonali`: gestione della valutazione
- `Servizio`: gestione delle informazioni del servizio
- Factory classes: creazione di oggetti specifici

6. Polymorphism/Indirection

Anche se non completamente visibile nel diagramma, l'uso delle Factory classes suggerisce l'applicazione di questi pattern per:

- Astrarre la creazione di oggetti
- Permettere variazioni nell'implementazione senza modificare il client

Pattern GOF Identificati Nei Diagrammi Di Progetto

Diagramma Classe Progetto ParametriPersonali

1. Factory Method

Classe: `ParametriPersonaliFactory`

Descrizione: Incapsula la creazione di oggetti **Parametri Personali** per semplificare l'istanziazione e la logica di creazione

Diagramma Classe Progetto Questionario

1. Factory Method

Classe: `RisposteQuestionarioFactory`

Descrizione: Incapsula la creazione di oggetti **Risposte Questionario** per semplificare l'istanziazione e la logica di creazione

Diagramma Classe Progetto Prodotti

1. Factory Method

Classe: `ProdottiFactory`

Descrizione: Incapsula la creazione di oggetti **Prodotto** per semplificare l'istanziazione e la logica di creazione

Diagramma Classe Progetto ValutazioneParametriPersonali

1. Factory Method

Classe: `ValutazioneParametriPersonaliFactory`

Descrizione: Incapsula la creazione di oggetti **Valutazione Parametri Personali** per semplificare l'istanziazione e la logica di creazione

Conclusione e Responsabilità

Il presente elaborato ha affrontato l'estensione del sistema di gestione della palestra integrando le nuove funzionalità richieste: merchandising, monitoraggio del benessere, invito amici e questionari di valutazione.

Sono stati prodotti e analizzati i referti richiesti ed ogni membro del gruppo ha contribuito allo sviluppo del progetto secondo il seguente schema di responsabilità:

- Ardò Cosimo Andrea:
  - ValutareParametriPersonaliCliente, InserisciValutazioneParametriPersonali, Diagramma Classe Parametri Personali(parte Valutazione)
- D'adda Tommaso:
  - AggiungereMerchandising, InserisciProdotto, Diagramma Classe Prodotti
- Ferrari Pagini Alessandro:
  - CompilareQuestionario, InserisciRisposteQuestionario, Diagramma Classe Questionario
- Lagae Nicole:
  - AggiungereParametriPersonali, InserisciParametriPersonali, Diagramma Classe Parametri Personali(parte ParametriPersonali)