

Traccia

Titolo

Sviluppare una applicazione Web per la gestione di una palestra.

Utenti

L'applicazione può essere utilizzata da parte di utenti di diverso tipo: utenti non autenticati, utenti autenticati, clienti abbonati, personal trainer e gestori. Le funzionalità disponibili variano in base al tipo di utente che sta utilizzando l'applicazione.

Utenti autenticati e non autenticati

Gli utenti non autenticati potranno registrarsi o accedere alla piattaforma, visualizzare i piani di abbonamento disponibili, il calendario settimanale dei corsi di gruppo e le informazioni sui vari personal trainer.

Gli utenti autenticati, in aggiunta alle funzionalità disponibili agli utenti non autenticati, potranno effettuare il logout dal proprio account, creare e modificare un profilo personale e sottoscrivere un piano di abbonamento, diventando clienti della palestra. Un utente autenticato non può sottoscrivere un piano di abbonamento qualora non avesse creato un profilo personale. Gli utenti autenticati hanno la possibilità di caricare una foto profilo in fase di creazione o di modifica del proprio profilo.

Clienti abbonati

Un cliente abbonato è un utente autenticato a cui è associato un piano di abbonamento ancora in corso di validità. In aggiunta a tutte le funzionalità disponibili per utenti autenticati e non autenticati, i clienti abbonati potranno:

- Gestire il proprio abbonamento: sarà possibile visualizzare le informazioni relative al piano di abbonamento sottoscritto, come data di scadenza e prezzo mensile. Risulta inoltre possibile cancellare e rinnovare il piano di abbonamento. Se l'abbonamento dell'utente venisse cancellato o fosse scaduto, l'utente perderebbe temporaneamente (fino al rinnovo o alla sottoscrizione di un nuovo piano) l'accesso alle funzionalità riservate ai clienti abbonati.
- Gestire le prenotazioni di allenamenti e corsi di gruppo: i clienti abbonati possono iscriversi ai corsi di gruppo disponibili e prenotare allenamenti personalizzati, in base a obiettivi di fitness specifici, con i personal trainer. Siccome il calendario dei corsi è settimanale, il cliente non potrà prenotarsi per eventi svolti in una data passata nel corso della settimana corrente. Un cliente non potrà prenotarsi a corsi di gruppo qualora il massimo numero di iscritti fosse stato raggiunto. Inoltre, per evitare sovrapposizioni, non sarà possibile prenotarsi contemporaneamente per corsi di gruppo e allenamenti con i personal trainer all'interno della stessa finestra temporale. I clienti hanno la possibilità di disiscriversi dagli eventi non ancora cominciati.
- Visualizzare il proprio programma settimanale: i clienti hanno a disposizione una dashboard nella quale possono visualizzare gli allenamenti e i corsi di gruppo prenotati nella settimana corrente.
- Gestire le recensioni: se un evento presente nella dashboard (allenamento o corso di gruppo) è terminato, il cliente ha la possibilità di lasciare una recensione. Le recensioni possono essere modificate o cancellate dal cliente che le ha rilasciate.

Personal trainer

I Personal Trainer possono creare e modificare il proprio profilo personale, specificando gli obiettivi di fitness di loro competenza, caricando un file contenente il CV e, eventualmente, la foto che verrà mostrata all'interno della pagina di presentazione dei Personal Trainer. Ciascun Personal Trainer ha a disposizione una dashboard personale nella quale, come nel caso dei clienti, è possibile visualizzare gli impegni settimanali. Per i corsi di gruppo sarà possibile visualizzare il profilo di tutti i partecipanti, mentre per gli allenamenti personali sarà possibile visualizzare il profilo del cliente che ha prenotato l'allenamento. Sempre all'interno della dashboard, i Personal Trainer potranno visualizzare le recensioni relative agli eventi in cui sono coinvolti.

Manager

I manager, ossia i gestori della palestra, hanno a disposizione le seguenti funzionalità:

- Gestione dei piani di abbonamento: è possibile creare nuovi piani di abbonamento, oltre che modificare e rimuovere quelli preesistenti. Inoltre, è possibile applicare sconti speciali, in base alla durata del piano sottoscritto e in base all'età del sottoscrittore.
- Gestione dei corsi di gruppo: è possibile creare nuovi corsi di gruppo, oltre che modificare e rimuovere quelli preesistenti.
- Gestione degli obiettivi di fitness della palestra: è possibile creare nuovi obiettivi di fitness, oltre che rinominare e rimuovere quelli preesistenti.
- Visualizzazione di tutte le recensioni.

Profili utente

Tutti gli utenti autenticati eccetto i manager hanno la possibilità di creare e modificare un profilo contenente le principali informazioni personali. Come anticipato, la creazione di un profilo personale è obbligatoria per abbonarsi. La seguente tabella mostra come sono strutturati i **profili** degli utenti.

Campo	Tipo	Obbligatorio	Utenti	Vincoli
Nome	Stringa	SI	Tutti	Massimo 100 caratteri alfanumerici
Cognome	Stringa	SI	Tutti	Massimo 100 caratteri alfanumerici
Sesso	Carattere	SI	Tutti	Scelta tra 'M' e 'F'
Data di Nascita	Data	SI	Tutti	<ul style="list-style-type: none">• Età massima: 100 anni• Età minima: 14 anni per i clienti e 18 anni per i Personal Trainer
Peso	Numero	NO	Clienti	Compreso tra 55 e 230 cm
Altezza	Numero	NO	Clienti	Compreso tra 50 e 150 kg
Foto Profilo	File	NO	Tutti	Solo file di tipo immagine
Certificazioni	File	NO	Personal Trainer	Solo file testuali o PDF
Obiettivi di fitness	Fitness Goal	NO	Personal Trainer	Scelta tra gli obiettivi di fitness della palestra
Foto riconoscitiva	File	NO	Personal Trainer	Solo file di tipo immagine

Piani di abbonamento

I piani di abbonamento permettono agli utenti di usufruire dei servizi erogati dalla palestra. Gli utenti possono sottoscrivere un abbonamento scelto tra quelli creati dai manager della palestra. Un generico **piano di abbonamento** è strutturato in questo modo:

Campo	Tipo	Obbligatorio	Vincoli
Nome	Stringa	SI	Massimo 100 caratteri alfanumerici
Tipo	Stringa	SI	Scelta tra 'FULL', 'WEIGHTS' e 'GROUP'
Prezzo mensile	Numero	SI	Compreso tra 0 e 100
Riduzione mensile in base all'età	Numero	SI	Compreso tra 0 (default) e il prezzo mensile

Esistono tre tipi di piano di abbonamento:

- **FULL**: possibilità di frequentare corsi di gruppo e di prenotare allenamenti con Personal Trainer.
- **WEIGHTS**: possibilità di prenotare allenamenti con Personal Trainer.
- **GROUP**: possibilità di prenotare corsi di gruppo.

Per quanto riguarda la durata, l'utente ha la possibilità di sottoscrivere un piano di abbonamento di durata mensile, trimestrale, semestrale oppure annuale. Il prezzo mensile sarà calcolato dinamicamente in base a tre dati:

1. **Piano di abbonamento**: piani di abbonamento diversi hanno prezzi mensili differenti.
2. **Età dell'utente**: il sistema è in grado di supportare eventuali riduzioni di prezzo in base all'età, in particolare per gli utenti minorenni e per gli utenti Over 65.
3. **Durata della sottoscrizione**: Il sistema è in grado di supportare scontistiche che variano in base al tipo di piano scelto e in base alla sua durata.

Gli **sconti** possono essere impostati solamente dai manager, e sono strutturati in questo modo:

Campo	Tipo	Obbligatorio	Vincoli
Durata	Numero	SI	Scelta tra 1,3,6 e 12 mesi
Percentuale	Numero	SI	Compreso tra 0 (default) e 100
Piano di abbonamento	Piano di abbonamento	SI	Scelta tra uno dei piani di abbonamento creati per la palestra

Eventuali modifiche agli sconti da parte dei manager entreranno in vigore da subito per gli utenti non abbonati, e alla scadenza dell'abbonamento per gli utenti abbonati.

Ora che è più chiaro il modo in cui sono gestiti i piani di abbonamento, possiamo parlare delle iscrizioni. Una **iscrizione** associa un utente al piano di abbonamento che ha sottoscritto, ed è strutturata in questo modo:

Campo	Tipo	Obbligatorio	Vincoli
Utente	Utente	SI	<ul style="list-style-type: none">• Unicità• Deve essere autenticato• Non deve possedere un abbonamento in corso di validità
Piano di abbonamento	Piano di abbonamento	SI	Scelta tra uno dei piani di abbonamento creati per la palestra
Data di inizio	Data	SI	Calcolato dinamicamente
Data di fine	Data	SI	Calcolato dinamicamente
Prezzo ridotto	Booleano	SI	Calcolato dinamicamente

La data di inizio e la data di fine sottoscrizione vengono calcolate al momento della sottoscrizione stessa: la data di inizio coincide con il momento della sottoscrizione e la data di fine è calcolata aggiungendo alla data di inizio il numero di mesi corrispondenti alla durata scelta dall'utente.

Obiettivi di fitness

Un obiettivo di fitness (Fitness Goal) è un traguardo di allenamento specifico, identificato da un nome. I manager hanno la possibilità di creare, rinominare o rimuovere gli obiettivi di fitness che la palestra offre ai propri utenti. In fase di creazione del profilo, ciascun Personal Trainer dovrà selezionare quali obiettivi di fitness mettere a disposizione ai clienti che prenotano un allenamento con loro. In questo modo, quando un cliente prenota un allenamento con un Personal Trainer, potrà scegliere tra gli obiettivi di fitness offerti dal Personal Trainer stesso.

Allenamenti e Prenotazioni

Come anticipato, la palestra permette ai propri clienti di svolgere due tipi di allenamento. Un allenamento può essere un corso di gruppo (tenuto da un Personal Trainer) oppure una seduta personalizzata con un Personal Trainer per obiettivi di fitness specifici. Gli allenamenti hanno una struttura diversa in base al tipo.

Le sedute di **allenamento** con i Personal Trainer sono strutturate in questo modo:

Campo	Tipo	Obbligatorio	Vincoli
Ora di inizio	Numero	SI	Intero da 9 a 19 compresi
Giorno della settimana	Stringa	SI	Scelta tra i giorni della settimana da 'Monday' a 'Sunday'
Personal Trainer	Utente	SI	<ul style="list-style-type: none"> Scelta tra i Personal Trainer della palestra Il Personal Trainer non deve avere altre sedute di allenamento o corsi di gruppo nella fascia oraria scelta
Utente	Utente	SI	<ul style="list-style-type: none"> Deve essere autenticato Deve essere in possesso di un piano di abbonamento in corso di validità e che consenta di svolgere allenamenti con i PT
Obiettivi di fitness	Fitness Goal	SI	Scelta tra gli obiettivi di fitness del Personal Trainer
Informazioni aggiuntive	Stringa	NO	Massimo 500 caratteri

Come si può notare, questa tabella, includendo la voce "Utente", funge anche da prenotazione. Siccome i corsi di gruppo possono essere prenotati da più persone, abbiamo bisogno di due tabelle, una per descrivere il corso di gruppo e una per tenere traccia delle prenotazioni. I **corsi di gruppo** sono strutturati in questo modo:

Campo	Tipo del campo	Obbligatorio	Vincoli
Ora di inizio	Numero	SI	Intero da 9 a 19 compresi
Giorno della settimana	Stringa	SI	Scelta tra i giorni della settimana da 'Monday' a 'Sunday'
Personal Trainer	Utente	SI	<ul style="list-style-type: none"> Scelta tra i Personal Trainer della palestra Il Personal Trainer non deve avere altri corsi di gruppo nella fascia oraria scelta
Titolo	Stringa	SI	Massimo 100 caratteri alfanumerici
Partecipanti massimi	Numero	SI	Numero intero da 1 a 30
Partecipanti totali	Numero	SI	Calcolato dinamicamente, numero intero da 0 al numero massimo di partecipanti
Durata	Numero	SI	Numero intero da 10 a 120
Immagine	File	NO	Solo file di tipo immagine

Sarà importante garantire che i corsi di gruppo non si sovrappongano tra di loro durante la fase di creazione e di modifica.

Le prenotazioni dei corsi di gruppo vengono memorizzate in questo modo:

Campo	Tipo del campo	Obbligatorio	Vincoli
Corso di gruppo	Corso di gruppo	SI	Scelta tra i corsi di gruppo della palestra
Utente	Utente	SI	<ul style="list-style-type: none">• Deve essere autenticato• Deve essere in possesso di un piano di abbonamento in corso di validità e che consenta di svolgere corsi di gruppo• Non può avere un altro corso di gruppo o allenamento prenotato per la stessa fascia orario

Tutte le prenotazioni relative a corsi di gruppo e a sedute di allenamento vengono automaticamente cancellate una volta che si è consumato l'evento.

Recensioni

Come anticipato, i clienti che hanno partecipato a corsi di gruppo o a sedute di personal training possono rilasciare recensioni. Le recensioni sono strutturate in questo modo:

Campo	Tipo del campo	Obbligatorio	Vincoli
Evento	Corso di gruppo o seduta di allenamento con PT	SI	Scelta tra i corsi di gruppo settimanali oppure tra le sedute di allenamento con PT disponibili
Utente	Utente	SI	L'utente deve avere partecipato all'evento oggetto della recensione
Personal Trainer	Utente	SI	Il Personal Trainer deve avere tenuto l'evento oggetto della recensione
Titolo	Stringa	SI	Massimo 100 caratteri
Stelle	Numero	SI	Numero intero da 0 a 5
Recensione	Stringa	SI	Massimo 500 caratteri

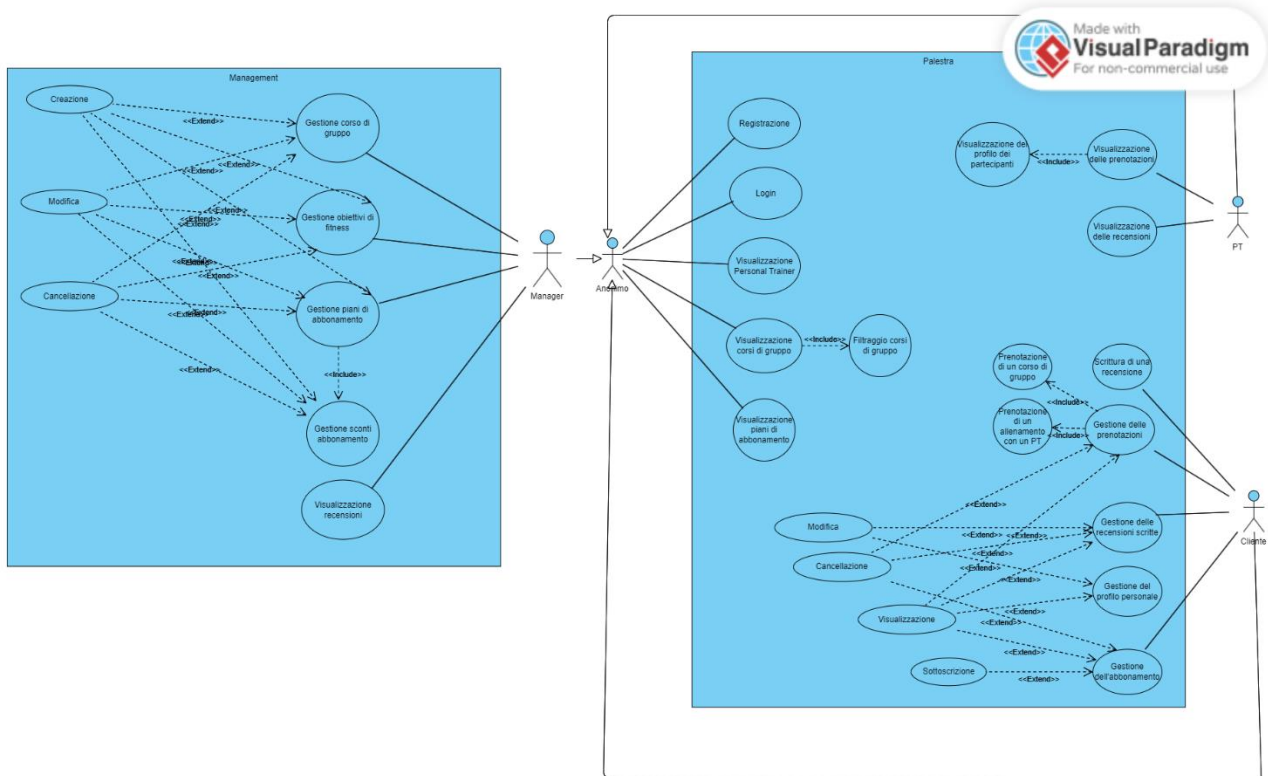
Ciascun cliente potrà visualizzare le recensioni che ha rilasciato, ciascun Personal Trainer potrà visualizzare le recensioni rilasciate dai clienti su di lui, e i manager potranno visualizzare tutte le recensioni.

Descrizione del progetto

Il progetto consiste nello sviluppo di un'applicazione web per la gestione di una palestra, che permette a utenti di diverso tipo di interagire con il sistema. L'applicazione è pensata per migliorare l'efficienza della gestione della palestra, facilitare l'accesso ai servizi offerti e migliorare l'esperienza complessiva degli utenti. Le principali categorie di utenti sono: utenti non autenticati, utenti autenticati, clienti abbonati, personal trainer e gestori. Ciascuna categoria ha accesso a funzionalità specifiche, progettate per soddisfare le diverse esigenze e ruoli all'interno della palestra. Nelle sezioni successive forniremo una panoramica ad alto livello del progetto utilizzando tre tipi di diagramma: il diagramma use case UML, il diagramma delle classi (class diagram) e il diagramma di attività (activity diagram). Nella directory **documentation** è possibile consultare da più vicino tutti questi diagrammi.

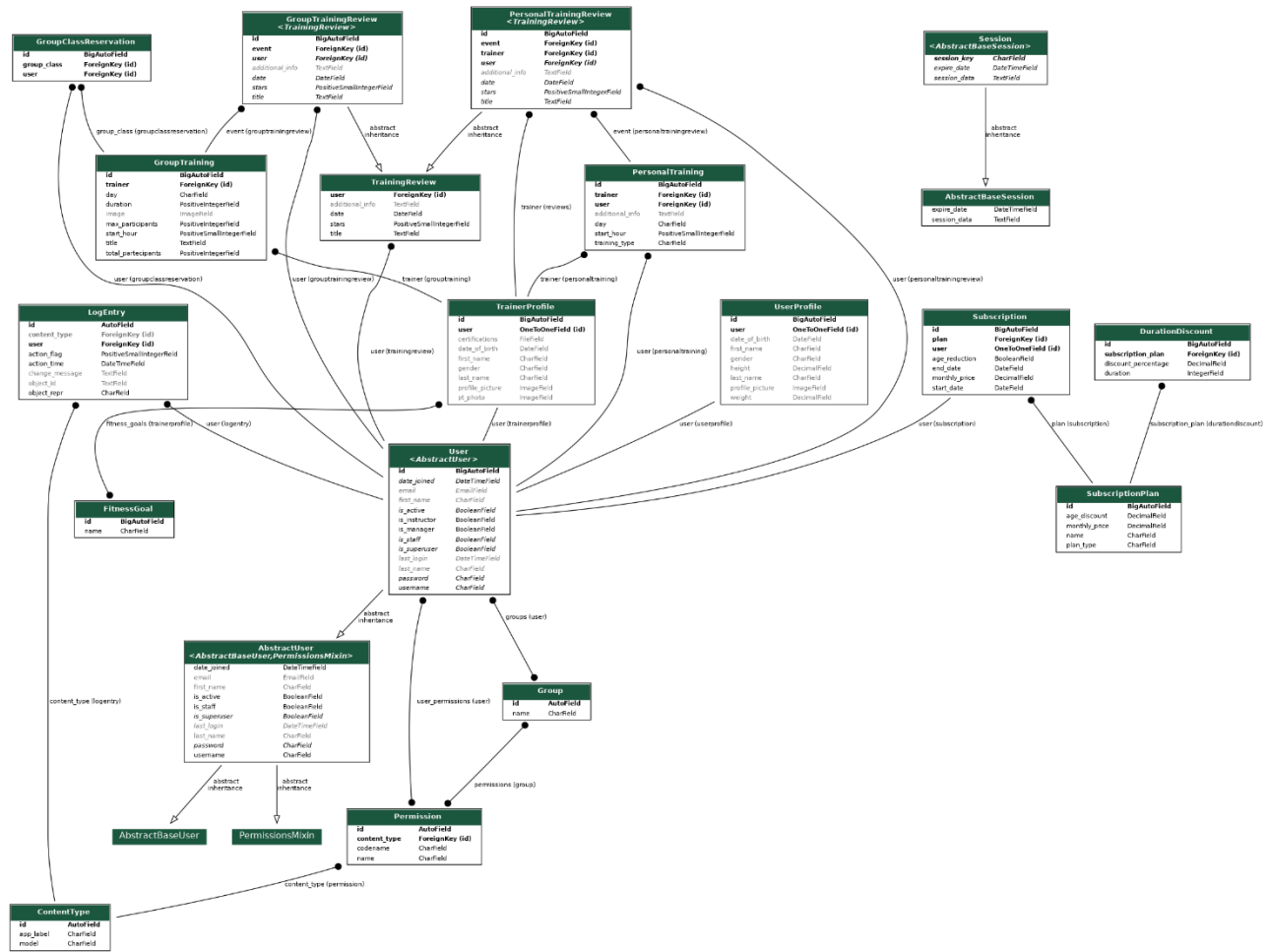
Diagramma use case UML

Il diagramma dei casi d'uso (Use Case Diagram) rappresenta le interazioni tra gli attori (utenti) e le funzionalità del sistema. Questo diagramma aiuta a visualizzare come gli utenti diversi possono utilizzare l'applicazione per svolgere attività specifiche. Ad esempio, un utente non autenticato può visualizzare i piani di abbonamento e il calendario dei corsi, mentre un cliente abbonato può anche gestire il proprio abbonamento e prenotare corsi di gruppo o allenamenti personalizzati. I personal trainer possono visualizzare e gestire i propri impegni, mentre i gestori hanno la possibilità di creare e modificare piani di abbonamento e corsi di gruppo. Ecco il diagramma use case UML del nostro progetto.



Descrizione del progetto - Diagramma delle classi

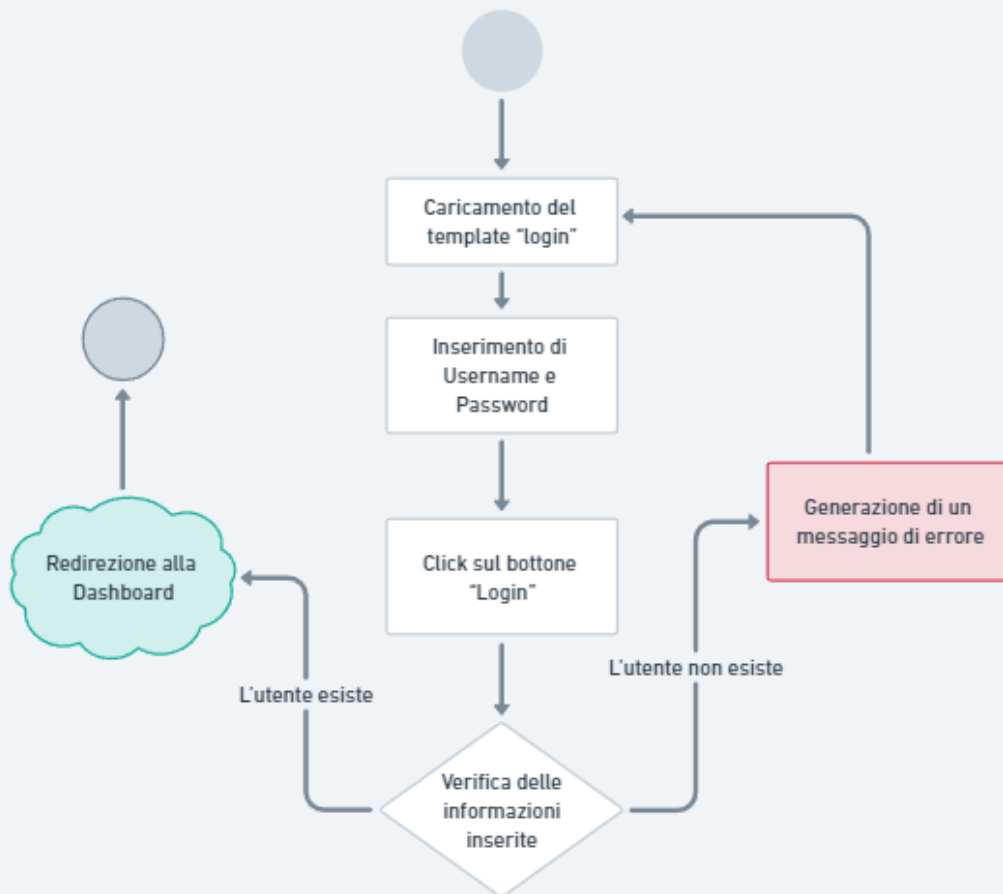
Il diagramma delle classi (Class Diagram) descrive la struttura della base dati dell'applicazione, utilizzando il linguaggio UML per rappresentare le classi, i loro attributi e le relazioni tra di esse. Questo diagramma è essenziale per comprendere la progettazione del database e come le diverse entità del sistema sono interconnesse. Ad esempio, una classe "User" può avere sottoclassi come "Customer" e "Supplier", ognuna con attributi specifici. Le relazioni tra le classi, come quelle tra "Product" e "Category" o tra "Order" e "Customer", sono fondamentali per la corretta implementazione delle funzionalità del sistema. Ecco il diagramma delle classi del nostro progetto:

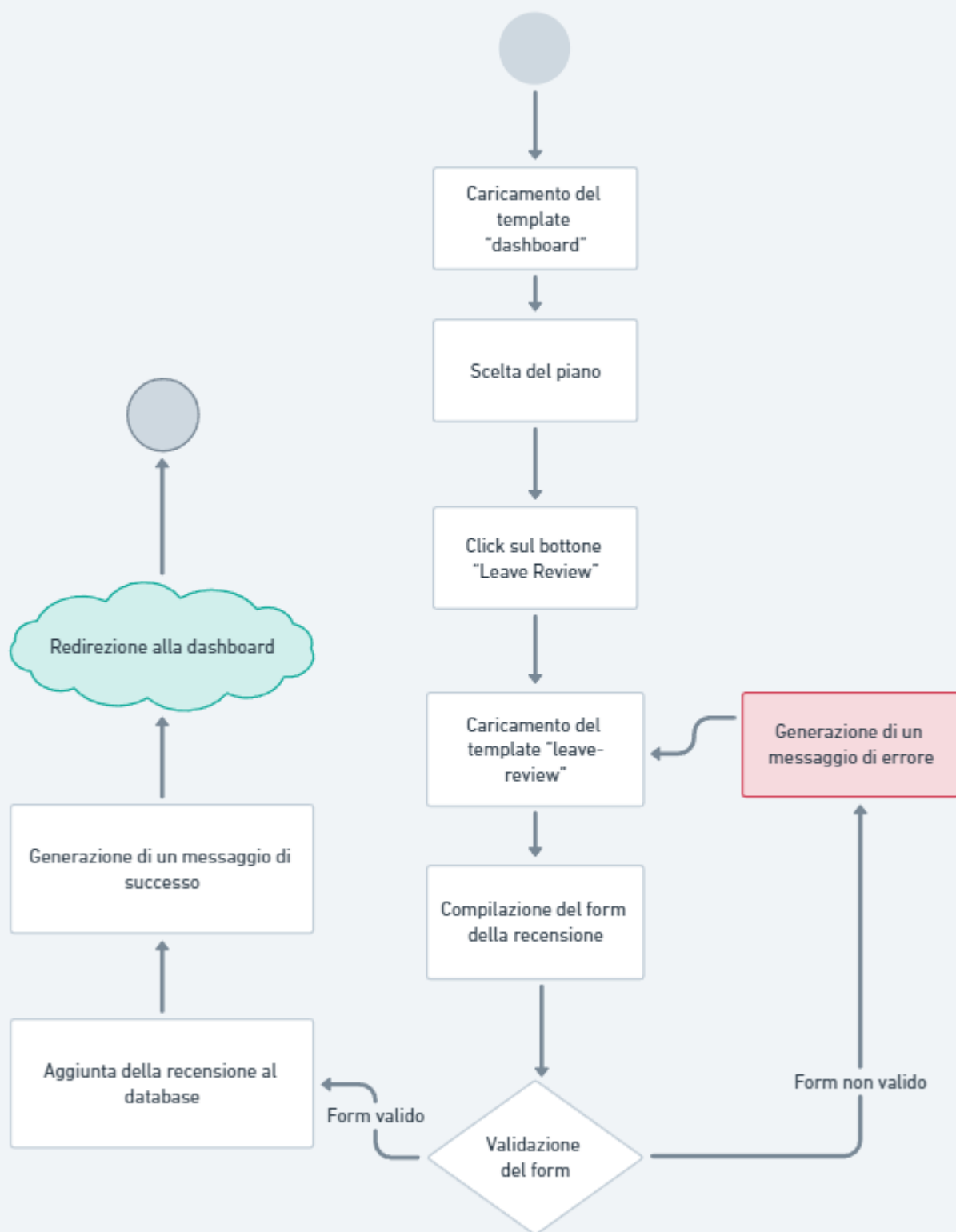


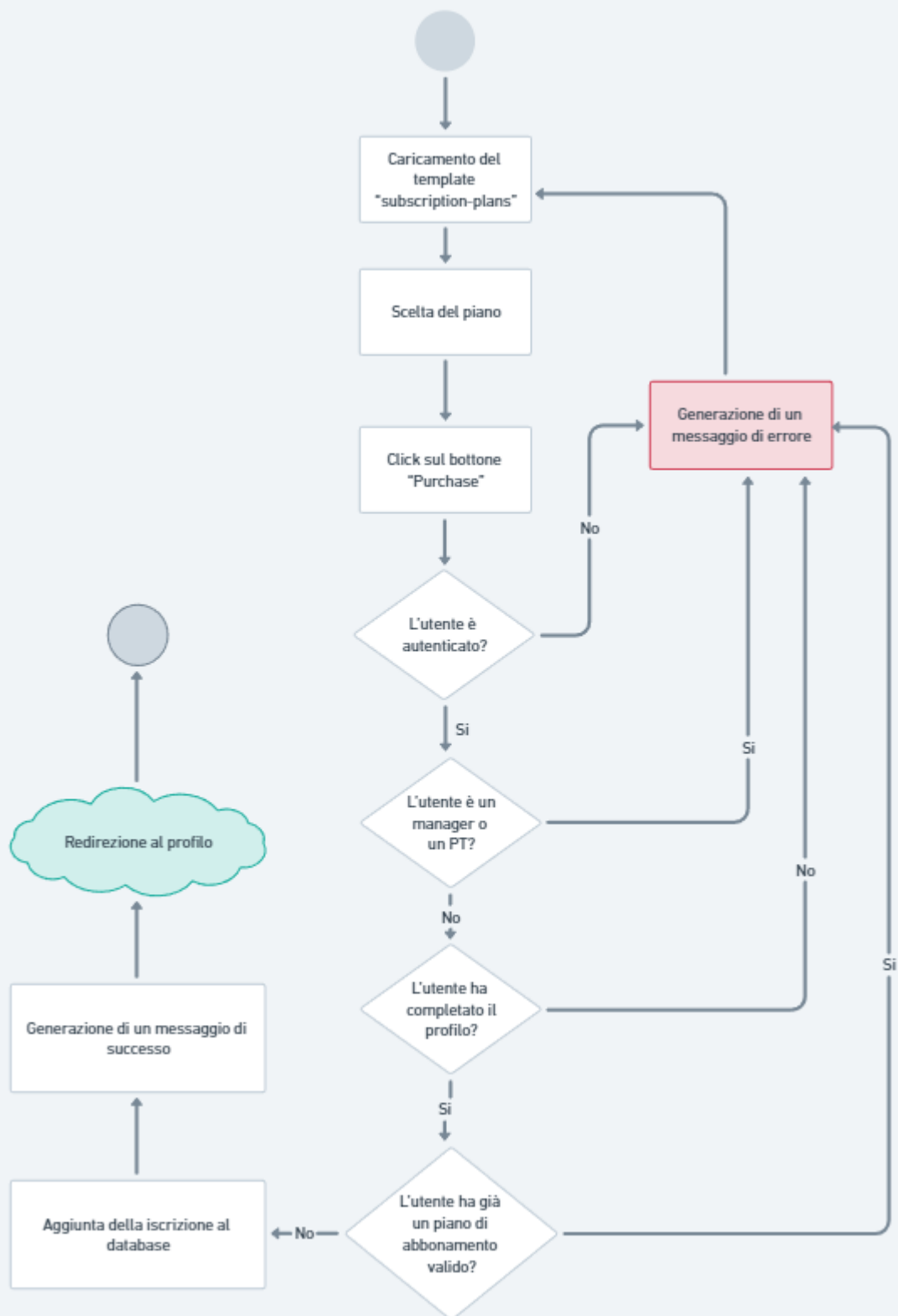
Descrizione del progetto - Diagramma di attività

Il diagramma di attività (Activity Diagram) rappresenta il flusso di lavoro delle principali operazioni del sistema, descrivendo la sequenza delle attività e le decisioni che influenzano il percorso di esecuzione. Questo tipo di diagramma è utile per modellare processi complessi e comprendere il comportamento dinamico dell'applicazione. Nel contesto del progetto, un diagramma di attività potrebbe illustrare il processo di registrazione di un nuovo utente, la gestione delle prenotazioni dei corsi di gruppo o il flusso per la sottoscrizione e il rinnovo di un piano di abbonamento. Questi diagrammi aiutano a garantire che tutti i possibili scenari siano considerati e gestiti correttamente.

Nello spazio seguente abbiamo inserito i diagrammi di attività relativi alle attività di login, sottoscrizione di un piano di abbonamento e rilascio di una recensione.







Tecnologie impiegate

Introduzione

In questo capitolo forniremo una panoramica delle tecnologie impiegate nello sviluppo del progetto. Discuteremo brevemente i framework principali utilizzati, l'architettura backend e frontend, gli strumenti di sviluppo adottati e le altre tecnologie e librerie di supporto.

Framework principale

Il nucleo del nostro progetto si basa sul framework web Django. Django fornisce una struttura solida per lo sviluppo di applicazioni web, offrendo funzionalità potenti per la gestione dei dati, la sicurezza e la scalabilità.

Backend

Il backend del nostro progetto è stato costruito utilizzando le seguenti tecnologie:

- Python v3.11.2: Il linguaggio di programmazione principale utilizzato per lo sviluppo del progetto. Python è conosciuto per la sua leggibilità, versatilità e vasta gamma di librerie disponibili.
- Django v5.0.6: Come già menzionato, Django è il framework principale. Esso include moduli per la gestione di utenti, autenticazione, sessioni e amministrazione.
- Celery: Utilizzati per la gestione di task asincroni e schedulati, migliorando la scalabilità e la gestione dei processi di background. Nel nostro progetto, Celery viene utilizzato per cancellare le prenotazioni dai corsi di gruppo quando questi terminano, in modo tale da consentire nuove iscrizioni per la settimana successiva.
- SQLite3: scelto come database relazionale per la sua affidabilità e compatibilità con Django.

Frontend

Il frontend del progetto utilizza le seguenti tecnologie:

- HTML, CSS, JavaScript: Per la struttura, lo stile e l'interattività della pagina web.
- Django Template Engine: Utilizzato per rendere dinamici i template HTML.
- Bootstrap (django-bootstrap4): Per creare un'interfaccia utente responsiva e moderna.

Altre tecnologie e librerie

Oltre alle tecnologie principali, il progetto ha fatto uso di altre librerie di supporto:

- Pytz v2024.1: per la gestione dei fusi orari.
- python_dateutil v2.9.0: utilizzo di "relativedelta" per gestire correttamente le durate degli abbonamenti.
- Pillow: libreria per la gestione delle immagini
- Django Extensions Fornisce utili estensioni e comandi aggiuntivi per Django, e consente di estrarre i modelli UML dal database.

La lista completa di dipendenze è consultabile tramite il file "requirements.txt", presente nella directory principale del progetto. Inoltre, come illustrato nel file "README.md", il file "requirements.txt" è necessario per l'installazione e il corretto funzionamento del progetto.

Motivazione delle Scelte

Python e Django

La scelta di Python come linguaggio di programmazione e Django come framework principale è stata in parte determinata dal curriculum del corso, che si focalizza su queste tecnologie. Ad ogni modo, Python, come accennato in precedenza, è noto per la sua leggibilità e semplicità, caratteristiche che facilitano l'apprendimento e la collaborazione tra sviluppatori. Inoltre, la vasta gamma di librerie e strumenti disponibili per Python rende questo linguaggio estremamente versatile. Per quanto riguarda Django, essendo un framework ad alto livello per lo sviluppo di applicazioni web, permette di accelerare lo sviluppo e di mantenere un codice pulito e manutenibile, consentendo una rapida implementazione delle funzionalità principali del progetto.

Altre Tecnologie Backend

Celery è stato scelto per la gestione dei task asincroni e schedulati, come la cancellazione automatica delle prenotazioni per i corsi di gruppo terminati. Questo aspetto è fondamentale per il corretto funzionamento del progetto, in quanto evita di cancellare manualmente le prenotazioni dai corsi di gruppo alla fine di ogni settimana.

SQLite3 è stato selezionato come database relazionale principalmente per la sua semplicità e compatibilità con Django. Un'ulteriore motivazione nella scelta di SQLite come database relazionale proviene dal seguente estratto (tradotto) della documentazione ufficiale di Django: “Di default, la configurazione utilizza SQLite. Se non hai dimestichezza con i database, o se sei semplicemente interessato nel provare Django, questa è la scelta migliore. Inoltre, SQLite è incluso in Python, di conseguenza non avrai bisogno di installare nient'altro per supportare il tuo database”.

Tecnologie Frontend

Per il frontend, abbiamo optato per HTML, CSS e JavaScript, tecnologie standard del web che garantiscono una solida base per la struttura e l'interattività delle pagine web. L'uso del motore di template di Django ha semplificato la generazione dinamica dei contenuti HTML, migliorando l'integrazione tra frontend e backend.

JavaScript ha svolto un ruolo cruciale nell'implementazione di funzionalità interattive avanzate:

- La generazione dinamica dei prezzi dei piani di abbonamento in base all'età dell'utente e al piano scelto ha migliorato l'esperienza utente, rendendo il processo di selezione del piano più intuitivo.
- I popup nella dashboard, attivati dai personal trainer, hanno fornito un'interfaccia immediata per visualizzare i partecipanti ai corsi di gruppo e agli allenamenti, migliorando l'efficienza nella gestione delle sessioni.
- La visualizzazione dinamica dei corsi di gruppo basata sui filtri inseriti dagli utenti ha reso la ricerca dei corsi più flessibile e personalizzata.

Bootstrap (tramite django-bootstrap4) è stato scelto per creare un'interfaccia utente moderna e responsiva. Questo framework CSS facilita lo sviluppo di layout che si adattano automaticamente a diverse risoluzioni e dispositivi, migliorando l'esperienza utente senza dover scrivere codice CSS complesso.

Altre librerie

Pytz è stato utilizzato per la gestione avanzata delle date e dei fusi orari, critici per la corretta gestione delle durate degli abbonamenti e delle prenotazioni. Senza l'utilizzo di quest'ultimo, abbiamo

riscontrato una discrepanza di due ore (probabilmente dovuta al fuso orario di Django) nella gestione delle ore di inizio dei corsi di gruppo e degli allenamenti.

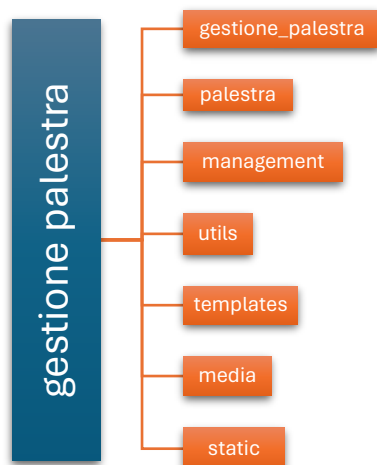
Pillow è stata scelta per la manipolazione delle immagini, permettendo agli utenti di caricare una immagine del profilo, ai personal trainer di caricare una propria foto personale da mostrare nell'apposito template, e ai manager di caricare le foto dei corsi di gruppo da mostrare nell'apposito template.

Django Extensions ha fornito comandi aggiuntivi e utili estensioni che hanno semplificato lo sviluppo e la manutenzione del progetto. Queste estensioni hanno anche reso possibile l'estrazione di modelli UML dal database, facilitando la comprensione e la documentazione della struttura del progetto.

Organizzazione logica dell'applicazione

Introduzione

L'immagine raffigura come è strutturato logicamente il nostro progetto.



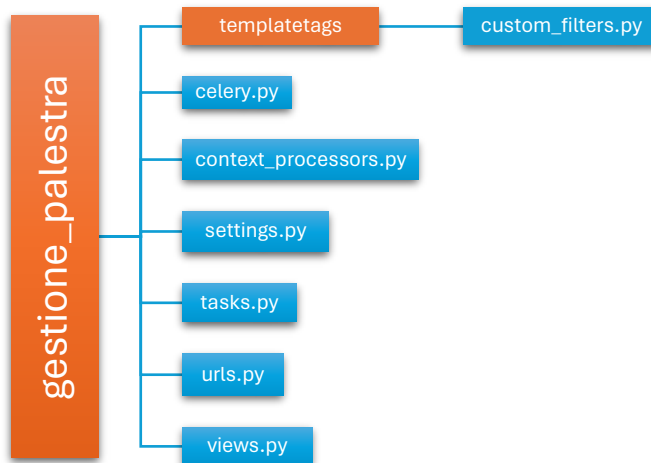
Questa struttura è composta da diverse directory e moduli, ognuno con una specifica funzione nell'applicazione:

- **Gestione_palestra:** questa è la directory principale del progetto Django. Contiene le configurazioni globali del progetto, come **settings.py**, **urls.py**, e **wsgi.py**. È il punto di partenza per l'intero progetto Django.
- **Palestra:** questa directory rappresenta un'applicazione Django dedicata alla gestione degli utenti, dei profili e delle prenotazioni di corsi di gruppo e allenamenti.
- **Management:** questa directory è un'altra applicazione Django che consente ai manager di effettuare tutte le operazioni descritte nella sezione [manager](#).
- **Utils:** contiene funzioni di codice applicativo (.py) utilizzate da tutte le app.
- **Templates:** questa directory contiene i template HTML utilizzati per il rendering delle pagine web. È organizzata in sottodirectory corrispondenti alle diverse applicazioni (palestra, management) per mantenere i file template ben organizzati.

- Media: contiene i file caricati dagli utenti, come le foto profilo, i CV, le immagini dei corsi di gruppo.
- Static: contiene file statici, come i fogli di stile (.css) e le immagini non destinate ad essere modificate da parte degli utenti.

Nelle prossime sezioni esamineremo nel dettaglio le sottodirectory più importanti del progetto.

Sottodirectory gestione_palestra

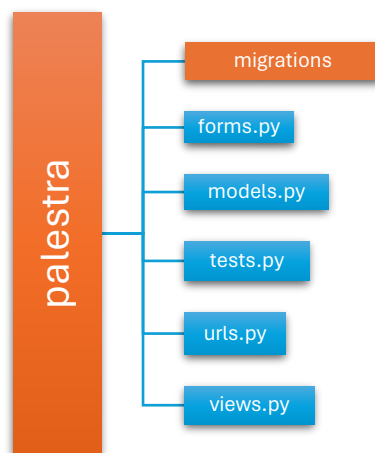


Come anticipato, questa è la directory principale del progetto Django. I componenti principali sono i seguenti:

- **Templatetags:** questa directory contiene i tag personalizzati per i template Django. Questi tag possono essere utilizzati nei file HTML per aggiungere logica personalizzata nelle viste. Nel nostro caso, all'interno di questa directory abbiamo inserito il file **custom_filters.py**, che contiene i seguenti filtri personalizzati: sottrazione, moltiplicazione, divisione, `custom_range`. Questi filtri vengono utilizzati per manipolare i dati prima di essere visualizzati direttamente all'interno del template.
- **Celery.py:** questo file è utilizzato per configurare Celery, un task queue che permette di eseguire operazioni in background (configurate in **tasks.py**) in modo asincrono.
- **Context_processors.py:** questo file contiene funzioni di processori di contesto personalizzati. I processori di contesto sono utilizzati per aggiungere variabili di contesto alle viste Django, rendendole disponibili in tutti i template. Nel nostro caso, abbiamo definito un contesto globale rendendo disponibili in tutte le viste i seguenti elementi: nome della palestra, ora attuale, obiettivi di fitness, nomi dei Personal Trainer, orari della palestra ecc.
- **Settings.py:** Questo è il file di configurazione principale per l'applicazione Django. Contiene le impostazioni del database, applicazioni installate, middleware, e altre configurazioni globali necessarie per il funzionamento del progetto. In aggiunta alle usuali configurazioni inserite in questo file, come directory dei template e dei file statici, abbiamo aggiunto la configurazione di **Celery**, tramite la definizione di **CELERY_IMPORTS** e **CELERY_BEAT_SCHEDULE**.
- **Tasks.py:** questo file contiene definizioni di task che possono essere eseguiti in background usando Celery. All'interno di **tasks.py** abbiamo definito due task:

- **Delete_reservations**: questo task, che viene eseguito ogni domenica alle 18, elimina tutte le prenotazioni dei corsi di gruppo, permettendo nuove iscrizioni la settimana successiva.
 - **Reset_training_info**: questo task, che viene eseguito ogni ora, imposta il numero totale di partecipanti a 0 per i corsi di gruppo terminati.
- **Urls.py**: questo file contiene le definizioni delle rotte URL per l'applicazione. Mappa gli URL alle viste corrispondenti, definendo quali viste vengono chiamate per specifici URL richiesti. In **urls.py** abbiamo inserito:
 - tutti gli url di **management** e di **palestra** utilizzando due namespace
 - url per il login, il logout e per la homepage.
 - File media, specificati tramite **MEDIA_ROOT**.
- **Views.py**: questo file contiene le definizioni delle viste che non fanno parte di un'app specifica, come homepage, login e logout.

Sottodirectory palestra

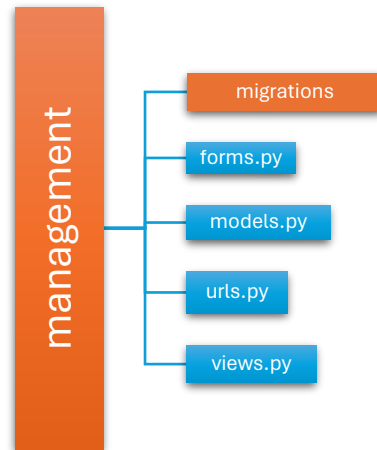


Questa directory contiene l'implementazione dell'app **palestra**:

- **Migrations**: contiene file che tengono traccia delle modifiche alla struttura del database. Questi file vengono generati automaticamente con i comandi **makemigrations** e **migrate**.
- **Forms.py**: contiene definizioni di form Django. Utilizzato per creare form HTML e gestire la validazione e la conversione dei dati degli input utente. I form inclusi in **forms.py** sono molteplici, i più importanti sono i form relativi alla creazione/modifica del profilo personale, e quelli che consentono di rilasciare recensioni.
- **Models.py**: contiene definizioni di modelli Django. Un modello è una classe che mappa una tabella del database. Definisce i campi e i comportamenti dei dati che si desidera archiviare. I modelli inseriti in **models.py** sono quelli relativi alla gestione degli utenti e dei loro profili, quelli relativi alle prenotazioni di allenamenti e corsi di gruppo, quelli per le recensioni e per l'iscrizione alla palestra.
- **Tests.py**: questo file contiene i **test** automatizzati per l'app **palestra**.
- **Urls.py**: questo file contiene le definizioni degli URL specifici per l'app **palestra**.

- Views.py: questo file contiene le definizioni delle viste specifiche per l'app **palestra**.

Sottodirectory management



Questa directory contiene l'implementazione dell'app **management**:

- Migrations: contiene file che tengono traccia delle modifiche alla struttura del database. Questi file vengono generati automaticamente con i comandi **makemigrations** e **migrate**.
- Forms.py: contiene definizioni di form Django relativi all'app **management**. I form inclusi in **forms.py** sono quelli relativi alla creazione/modifica dei corsi di gruppo, obiettivi di fitness, piani di abbonamento e sconti.
- Models.py: contiene definizioni di modelli Django relativi all'app **management**. I modelli inseriti in **models.py** sono quelli relativi alla definizione dei corsi di gruppo, dei piani di abbonamento, degli obiettivi di fitness e degli sconti.
- Urls.py: questo file contiene le definizioni degli URL specifici per l'app **management**.
- Views.py: questo file contiene le definizioni delle viste specifiche per l'app **management**.

Scelte implementative

Introduzione

In questa sezione sono descritte le scelte implementative principali che sono state adottate nello sviluppo del progetto. Tali decisioni sono state prese per garantire un'esperienza utente ottimale e una gestione efficiente del sistema. Di seguito vengono elencate e motivate alcune delle scelte più significative effettuate durante lo sviluppo.

Filtraggio dei corsi di gruppo

Per la visualizzazione dei corsi di gruppo, è stata implementata una tabella dinamica all'interno del template "**classes-schedule.html**". Questa scelta permette agli utenti di filtrare i corsi in base a diversi criteri (giorno, ora, Personal Trainer, disponibilità), migliorando notevolmente la navigabilità e la

personalizzazione dell'esperienza utente. Un corso di gruppo è disponibile se il numero totale di partecipanti è minore del numero massimo di partecipanti e se la data di inizio del corso è nel futuro. Questo approccio è motivato da una migliore usabilità del sistema, permettendo agli utenti di trovare i corsi che meglio si adattano ai loro orari e preferenze.

Generazione dinamica dei prezzi dei piani di abbonamento

All'interno del template “**subscription-plans.html**”, i prezzi dei piani di abbonamento vengono generati dinamicamente. Questo include sia il prezzo standard che quello scontato per fasce d'età specifiche (under 18 e over 65). I prezzi variano in base alla durata dell'abbonamento selezionato. Questo approccio permette una visualizzazione compatta dei piani di abbonamento, evitando di mostrare troppi prezzi in un unico riquadro. Infatti, per ogni piano di abbonamento, vengono mostrati 4 costi: il prezzo mensile intero, il prezzo mensile ridotto, il costo totale intero e il costo totale ridotto.

Dashboard Differente in Base al Tipo di Utente

Il template della dashboard (“**dashboard.html**”) è stato progettato per adattarsi al ruolo dell'utente (manager, istruttore, membro). Ogni tipologia di utente ha accesso a funzionalità specifiche e personalizzate, come la gestione delle classi per i manager e la visualizzazione del proprio programma settimanale per i membri. Questo approccio è efficiente, in quanto evita di creare un template per ogni tipo di utente, e sicuro, in quanto limita l'accesso a funzionalità specifiche in base al ruolo, impedendo, ad esempio, ad un cliente di modificare corsi di gruppo.

Disponibilità dinamica nei form di prenotazione

Per la gestione delle prenotazioni delle sessioni di allenamento con i Personal Trainer, è stata implementata una logica di disponibilità dinamica all'interno del form di prenotazione. Questa funzionalità permette agli utenti di vedere in tempo reale le fasce orarie disponibili per un determinato personal trainer, migliorando notevolmente l'esperienza utente e ottimizzando la gestione delle risorse. Tuttavia, questo approccio non elimina la necessità di effettuare dei controlli backend, in quanto un utente potrebbe cambiare i campi del form di prenotazione tramite lo strumento “ispeziona”. Ad ogni modo, la logica di disponibilità dinamica è stata implementata nel template “**book-workout.html**”.

Test effettuati

Lo unit testing è una tecnica di verifica del software in cui singole unità di codice, come funzioni, metodi, o classi, vengono testate in isolamento dal resto del programma. L'obiettivo è assicurarsi che ogni unità funzioni correttamente secondo le specifiche e le aspettative. I test unitari assicurano agli sviluppatori che ogni parte di codice si comporti in maniera corretta, facilitando la manutenzione del codice e migliorandone la qualità. Gli unit test rendono infatti possibile catturare e correggere errori nelle prime fasi dello sviluppo di un progetto e assicurarsi che le modifiche al codice esistente non introducano nuovi errori.

Django fornisce vari strumenti per implementare unit test. Ecco i principali:

1. `django.test.TestCase`: per creare test, si definiscono classi di test che ereditano da questa classe.
2. `django.test.Client`: è una classe che permette di simulare richieste HTTP a viste Django e di controllare le risposte.
3. Django crea automaticamente un database di test per eseguire i test, garantendo che i test non influiscano sul database di sviluppo o produzione.

Nel nostro progetto abbiamo deciso di testare una funzione di codice applicativo e una vista.

Test sulla funzione di codice applicativo

La funzione di codice applicativo che è stata testata si occupa di restituire le recensioni da presentare nella dashboard. Tale funzione, chiamata “get_reviews()”, prende in input un oggetto “User” e restituisce l’insieme di recensioni da mostrare nella dashboard per quell’utente. Ricordando che:

- Un utente non autenticato non può accedere alla dashboard.
- Un utente autenticato può visualizzare le recensioni che ha scritto.
- Un Personal Trainer può visualizzare tutte le recensioni su di lui.
- Un Manager può visualizzare tutte le recensioni della palestra.

La funzione “get_reviews()” deve essere in grado di restituire correttamente le recensioni in base al tipo di utente che tenta di accedere alla dashboard. La fase di setup per questi test è avvenuta in questo modo:

1. Generazione di un utente di tipo Manager (self.manager).
2. Generazione di una lista casuale di 100 utenti (self.normal_users).
3. Generazione di una lista casuale di 10 Personal Trainer (self.trainers).
4. Generazione di una lista casuale di 500 recensioni (self.reviews, di cui 250 relative a corsi di gruppo e 250 relative a sedute di allenamento)

Ecco una tabella riassuntiva dei test che abbiamo effettuato sulla funzione “get_reviews”:

Descrizione del test:	Comportamenti desiderati
test_user_not_authenticated: viene chiamata “get_reviews” con “AnonymousUser” come input.	L’output di “get_reviews” deve coincidere con un insieme vuoto.
test_user_authenticated: viene chiamata “get_reviews” con un utente normale scelto casualmente come input.	L’output di “get_reviews” deve coincidere con l’insieme di recensioni in “self.reviews” scritte dall’utente.
test_user_authenticated_after_edit: viene chiamata “get_reviews” con un utente normale scelto casualmente come input e si modifica una recensione scelta casualmente tra quelle scritte dall’utente scelto.	1) L’output di “get_reviews” deve coincidere con l’insieme di recensioni in “self.reviews” scritte dall’utente. 2) Deve essere presente il messaggio di conferma “Your review has been successfully edited”. 3) L’output di “get_reviews” deve contenere la recensione con le modifiche.
test_user_authenticated_after_deletion: viene chiamata “get_reviews” con un utente normale scelto casualmente come input e si cancella una recensione scelta casualmente tra quelle scritte dall’utente scelto.	1) Il numero di recensioni “get_reviews” deve corrispondere con il numero di recensioni con l’insieme di recensioni in “self.reviews” scritte dall’utente – 1. 2) A seconda del tipo di recensione, l’istruzione “review.refresh_from_db()” deve generare l’eccezione “DoesNotExist”.
test_trainer: viene chiamata “get_reviews” con un Personal Trainer scelto casualmente come input.	L’output di “get_reviews” deve coincidere con l’insieme di recensioni in “self.reviews” il cui Personal Trainer coincide con il Personal Trainer scelto casualmente.
test_trainer_after_edit: un utente normale viene selezionato casualmente. Si modifica una recensione scelta casualmente tra quelle scritte dall’utente scelto. Viene chiamata “get_reviews” utilizzando come input il Personal Trainer associato alla recensione modificata.	1) L’output di “get_reviews” deve coincidere con l’insieme di recensioni in “self.reviews” il cui Personal Trainer coincide con il Personal Trainer associato alla recensione modificata. 2) L’output di “get_reviews” deve contenere la recensione con le modifiche.
test_trainer_after_deletion: un utente normale viene selezionato casualmente. Si cancella una recensione scelta casualmente tra quelle scritte dall’utente scelto. Viene chiamata “get_reviews” utilizzando come input il Personal Trainer associato alla recensione cancellata.	1) Il numero di recensioni “get_reviews” deve corrispondere con il numero di recensioni con l’insieme di recensioni in “self.reviews” in cui è coinvolto il Personal Trainer – 1. 2) A seconda del tipo di recensione, l’istruzione “review.refresh_from_db()” deve generare l’eccezione “DoesNotExist”.
test_manager: viene chiamata “get_reviews” con il manager “self.manager” scelto come input.	Il numero totale di recensioni in “self.reviews” deve coincidere con il numero totale di recensioni restituito da “get_reviews”.
test_manager_after_edit: un utente normale viene selezionato casualmente. Si modifica una recensione scelta casualmente tra quelle scritte dall’utente scelto. Viene chiamata “get_reviews” utilizzando come input il manager “self.manager”.	1) Il numero totale di recensioni in “self.reviews” deve coincidere con il numero totale di recensioni restituito da “get_reviews”. 2) L’output di “get_reviews” deve contenere la recensione con le modifiche.
test_manager_after_deletion: un utente normale viene selezionato casualmente. Si cancella una recensione scelta casualmente tra quelle scritte dall’utente scelto. Viene chiamata “get_reviews” utilizzando come input il manager “self.manager”.	1) Il numero di recensioni “get_reviews” deve corrispondere con il numero di recensioni con l’insieme di recensioni in “self.reviews” – 1. 2) A seconda del tipo di recensione, l’istruzione “review.refresh_from_db()” deve generare l’eccezione “DoesNotExist”.

Test sulla vista

I test sulla vista sono stati compiuti su “GymClassesView”. Tale vista permette agli utenti di prenotare corsi di gruppo. In generale, per quanto riguarda le prenotazioni dei corsi di gruppo, saranno verificati i comportamenti della vista in queste casistiche:

- L’utente non è autenticato oppure è un membro dello staff (Personal Trainer o Manager) oppure non ha completato il proprio profilo personale.
- L’utente non è abbonato o ha un abbonamento scaduto o che non consente di iscriversi a corsi di gruppo.
- L’utente ha già un allenamento prenotato per la stessa ora dello stesso giorno del corso di gruppo a cui desidera iscriversi oppure sta tentando di prenotarsi per la seconda volta ad un corso a cui è già iscritto.
- Il corso di gruppo che si desidera prenotare è pieno oppure è terminato.
- La prenotazione va a buon fine.

Inoltre, siccome la vista “GymClassesView” mostra anche le informazioni sui vari corsi di gruppo, sarà necessario verificare la veridicità delle informazioni mostrate a seguito di cambiamenti dei corsi di gruppo nel database, soprattutto in seguito alle cancellazioni o alle aggiunte di prenotazioni.

La fase di setup per questi test è avvenuta in questo modo:


1. Generazione di un utente di tipo Personal Trainer (self.trainer) e del relativo profilo (self.trainer_profile).
2. Generazione di un corso di gruppo disponibile (self.group_class), di uno non disponibile (self.expired_group_class) e di uno pieno (self.full_group_class), tutti sostenuti dal trainer (self.trainer).
3. Generazione di un piano di abbonamento che consente l’iscrizione a corsi di gruppo (self.subscription_plan), di uno non valido (self.wrong_subscription_plan), che consente soltanto le iscrizioni alle sedute con i Personal Trainer.
4. Generazione di tre abbonamenti: un abbonamento valido (self.valid_subscription), un abbonamento scaduto (self.expired_subscription). Questi due abbonamenti fanno riferimento al piano valido (self.subscription_plan). Il terzo abbonamento (self.wrong_subscription) fa riferimento al piano sbagliato (self.wrong_subscription_plan).
5. Generazione di cinque utenti: ai primi due utenti, il primo con profilo (self.user_with_profile) e il secondo senza (self.user_without_profile), è stato assegnato il piano di abbonamento valido (self.valid_subscription). Al terzo utente (user_with_wrong_subscription) è stato assegnato un profilo e l’abbonamento sbagliato (self.wrong_subscription). Al quarto utente (self.user_with_expired_subscription) è stato assegnato un profilo e una iscrizione scaduta (self.expired_subscription). Al quinto utente (self.user_with_no_subscription) è stato assegnato un profilo e nessun abbonamento.

Ecco una tabella riassuntiva dei test che abbiamo effettuato sulla vista “GymClassesView”:

Descrizione del test:	Comportamenti desiderati
Test_user_not_authenticated: un utente non autenticato, ottenuto tramite “self.client.logout()” cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “login”. 2) Presenza di un messaggio di errore “You must login first”.
test_user_is_not_a_member: un utente autenticato con profilo e senza piano di abbonamento (self.user_with_no_subscription) cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “subscription-plans”. 2) Presenza di un messaggio di errore “You are not a member”.
test_user_is_staff_member: un membro dello staff (self.trainer) cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “classes-schedule”. 2) Presenza di un messaggio di errore “You are a staff member”.
test_no_profile_info: un utente autenticato che non ha completato il profilo (self.user_without_profile) cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “profile”. 2) Presenza di un messaggio di errore “You must complete your profile first”.
test_user_wrong_subscription: un utente con un abbonamento che non consente di prenotarsi ai corsi di gruppo (self.user_with_wrong_subscription) cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “profile”. 2) Presenza di un messaggio di errore contenente la stringa “does not allow that”.
test_user_subscription_expired: un utente autenticato con profilo ma con un piano di abbonamento scaduto (self.user_with_expired_subscription) cerca di iscriversi a un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “profile”. 2) Presenza di un messaggio di errore “Your subscription has expired”.
test_group_class_full: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) tenta di iscriversi ad un corso di gruppo pieno (self.full_group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “classes-schedule”. 2) Presenza di un messaggio di errore “The group class you are trying to join is full”.
test_group_class_ended_for_week: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) tenta di iscriversi ad un corso di gruppo terminato (self.expired_group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “classes-schedule”. 2) Presenza di un messaggio di errore “This group class has ended”.
test_user_already_has_personal_training: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) tenta di iscriversi ad un corso di gruppo valido (self.group_class) ma ha già un allenamento prenotato nella stessa ora dello stesso giorno.	1) Status code della risposta 302 , reindirizzamento alla pagina “classes-schedule”. 2) Presenza di un messaggio di errore contenente la stringa “You have an upcoming personal training session already booked”.
test_user_already_booked: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) tenta di iscriversi due volte ad un corso di gruppo valido (self.group_class)	1) Status code della risposta 302 , reindirizzamento alla pagina “classes-schedule”. 2) Presenza di un messaggio di errore “You are already booked in this class”.
test_user_cancels_reservation: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) si disiscrive da un corso di gruppo valido (self.group_class)	1) Status code della risposta 302 , reindirizzamento alla pagina “dashboard”. 2) Presenza di un messaggio di conferma “The Group Class reservation has been cancelled”. 3) Il nuovo numero totale di partecipanti del corso di gruppo (self.group_class) coincide con il vecchio numero totale di partecipanti – 1.
test_successful_reservation: un utente autenticato con profilo e con piano di abbonamento valido (self.user_with_profile) tenta di iscriversi ad un corso di gruppo valido (self.group_class).	1) Status code della risposta 302 , reindirizzamento alla pagina “dashboard”. 2) Presenza di un messaggio di conferma “Group class joined successfully”. 3) Il nuovo numero totale di partecipanti del corso di gruppo (self.group_class) coincide con il vecchio numero totale di partecipanti + 1.

Risultati ottenuti


Trainers List



First Name: John
Last Name: Doe
Gender: M
Date of Birth: May 10, 1999
Fitness Goals:

- BodyBuilding
- Functional Fitness
- Rehabilitation


[Book a workout](#)



First Name: Jessica
Last Name: Smart
Gender: F
Date of Birth: May 11, 1989
Fitness Goals:

- Weight Loss
- Flexibility and Mobility


[Book a workout](#) [Download CV](#)



First Name: Michael
Last Name: Will
Gender: M
Date of Birth: Sept. 10, 1980
Fitness Goals:

- Strength Training

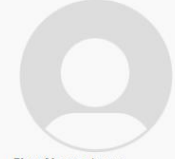
[Book a workout](#)



First Name: Jennifer
Last Name: Kendall
Gender: F
Date of Birth: Dec. 11, 1980
Fitness Goals:

- Weight Loss
- Functional Fitness
- Rehabilitation

[Book a workout](#)



First Name: Jamar
Last Name: Smith
Gender: M
Date of Birth: April 7, 1987
Fitness Goals:

- Strength Training
- Functional Fitness

[Book a workout](#)

Filter the group classes

Day:

All



Hour:

All



Trainer:

All



Available: ☐

Day	Title	Trainer	Start Hour	Duration	Max Participants	Remaining Spots	Status	Actions
Monday	HIIT	John Doe	10	45	15	0	Ended	Join Class
	Active Yoga	John Doe	9	30	10	0	Ended	Join Class
Tuesday	Cardio Attack	Jennifer Kendall	11	40	12	0	Ended	Join Class
	Postural	Jennifer Kendall	15	50	20	0	Ended	Join Class
Thursday	Pilates	John Doe	18	45	15	0	Ended	Join Class
Saturday	Zumba	Jessica Smart	11	60	30	29	Available	Join Class
Sunday	Tabata	Michael Will	11	50	15	13	Available	Join Class

Group Classes Plan

Monthly price:	25.99 \$
Monthly price with age discount (U18/O65):	20.99 \$
Full price:	25.99 \$
Full price with age discount (U18/O65):	20.99 \$
Duration:	<div><div></div><div>1 Month</div><div></div><div>3 Months</div><div></div><div>6 Months</div><div></div><div>12 Months</div></div>

Purchase

Gym Access Plan

Monthly price:	29.99 \$
Monthly price with age discount (U18/O65):	24.99 \$
Full price:	29.99 \$
Full price with age discount (U18/O65):	24.99 \$
Duration:	<div><div></div><div>1 Month</div><div></div><div>3 Months</div><div></div><div>6 Months</div><div></div><div>12 Months</div></div>

Purchase

Full Access Plan

Monthly price:	49.99 \$
Monthly price with age discount (U18/O65):	39.99 \$
Full price:	49.99 \$
Full price with age discount (U18/O65):	39.99 \$
Duration:	<div><div></div><div>1 Month</div><div></div><div>3 Months</div><div></div><div>6 Months</div><div></div><div>12 Months</div></div>

Purchase

af41's Dashboard

[Home](#)[Our trainers](#)[Gym classes schedule](#)[Subscription plans](#)

Your Schedule for this week:

Monday	10:00	HIIT Group Class with John Doe This event has ended.
Tuesday	11:00	Cardio Attack Group Class with Jennifer Kendall This event has ended.
Thursday	15:00	Weight Loss Workout with Jessica Smart This event has ended.
Thursday	18:00	Pilates Group Class with John Doe This event has ended.
Friday	9:00	BodyBuilding Workout with John Doe This event has ended.
Saturday	11:00	Zumba Group Class with Jessica Smart
Sunday	11:00	Tabata Group Class with Michael Will

Your reviews

Review #1 - score:5 written on May 16, 2024
Event: Weight Loss Workout with Jessica Smart
Title: Terrific workout

Edit Review

Delete Review

Review #2 - score:4 written on May 14, 2024
Event: Cardio Attack Group Class with Jennifer Kendall
Title: It was a good experience
Text: I would do it again!

Edit Review

Delete Review

pt1's Dashboard

[Home](#)[Our trainers](#)[Gym classes schedule](#)[Subscription plans](#)

Your Schedule for this week:

Event Participants

Name	Alessandro Franceschini
Date of Birth	2002-05-04
Gender	M
Height	184 cm
Weight	91 kg
Profile Picture	

Reviews about you

Review #1 - score:5 written by af41 on May 14, 2024
Event: HIIT Group Class
Title: Good

Monday	9:00	Active Yoga Group Class <div>View Participants</div>
Monday	10:00	HIIT Group Class <div>View Participants</div>
Thursday	9:00	Rehabilitation Workout with Davide Casarin This event has ended.
Thursday	18:00	Pilates Group Class <div>View Participants</div>
Friday	9:00	BodyBuilding Workout with Alessandro Franceschini This event has ended.

all group classes

Zumba

Edit Class Delete Class

Pilates

Edit Class Delete Class

Tabata

Edit Class Delete Class

HIIT

Edit Class Delete Class

Cardio Attack

Edit Class Delete Class

Postural

Edit Class Delete Class

Active Yoga

All Subscription Plans

Plan #1

Plan Name	Group Classes Plan
Full 1 Month Price	25.99 \$
Reduced 1 Month Price	20.99 \$
Full 3 Month Price	74.07 \$
Reduced 3 Month Price	59.07 \$
Full 6 Month Price	140.35 \$
Reduced 6 Month Price	110.35 \$
Full 1 Year Price	265.10 \$
Reduced 1 Year Price	205.10 \$

Edit Plan Edit Discounts Delete Plan

Plan #2

Plan Name	Gym Access Plan
Full 1 Month Price	29.99 \$
Reduced 1 Month Price	24.99 \$
Full 3 Month Price	85.47 \$
Reduced 3 Month Price	70.47 \$
Full 6 Month Price	161.95 \$
Reduced 6 Month Price	131.95 \$
Full 1 Year Price	302.30 \$
Reduced 1 Year Price	242.30 \$

Edit Plan Edit Discounts Delete Plan

Plan #3

Plan Name	Full Access Plan
Full 1 Month Price	49.99 \$
Reduced 1 Month Price	39.99 \$
Full 3 Month Price	134.97 \$
Reduced 3 Month Price	104.97 \$
Full 6 Month Price	254.95 \$
Reduced 6 Month Price	194.95 \$
Full 1 Year Price	479.90 \$
Reduced 1 Year Price	359.90 \$

Edit Plan Edit Discounts Delete Plan

Create new Subscription Plan

All reviews

Review #1 - scores:5

written by af41 on May 16, 2024

Event: Workout with Jessica Smart

Title: Terrific workout

Review #2 - scores:4

written by af41 on May 14, 2024

Event: Cardio Attack Group Class with Jennifer Kendall

Title: It was a good experience

Text: I would do it again!

Review #3 - scores:5

written by af41 on May 14, 2024

Event: HIIT Group Class with John Doe

All Fitness Goals

BodyBuilding

Strength Training

Weight Loss

Flexibility and Mobility

Functional Fitness

Rehabilitation

Sport Specific Training

Add new Fitness Goal

Delete

Rename

Delete

Rename

Delete

Rename

Delete

Rename

Delete

Rename

Delete

Rename

Delete

Rename

Type the name here

Add

First Name:

John

Last Name:

Doe

Gender:

Male

Date of Birth:

10/05/1999

Profile Picture:

Scegli file

Nessun file selezionato

Your Photo:

Scegli file

Nessun file selezionato

CV:

Scegli file

Nessun file selezionato

fitness goals:

☒ BodyBuilding

☐ Strength Training

☐ Weight Loss

☐ Flexibility and Mobility

☒ Functional Fitness

☒ Rehabilitation

☐ Sport Specific Training

Submit