UNIVERSITY OF TRENTO

Progetto EpiOpera

Gruppo T18

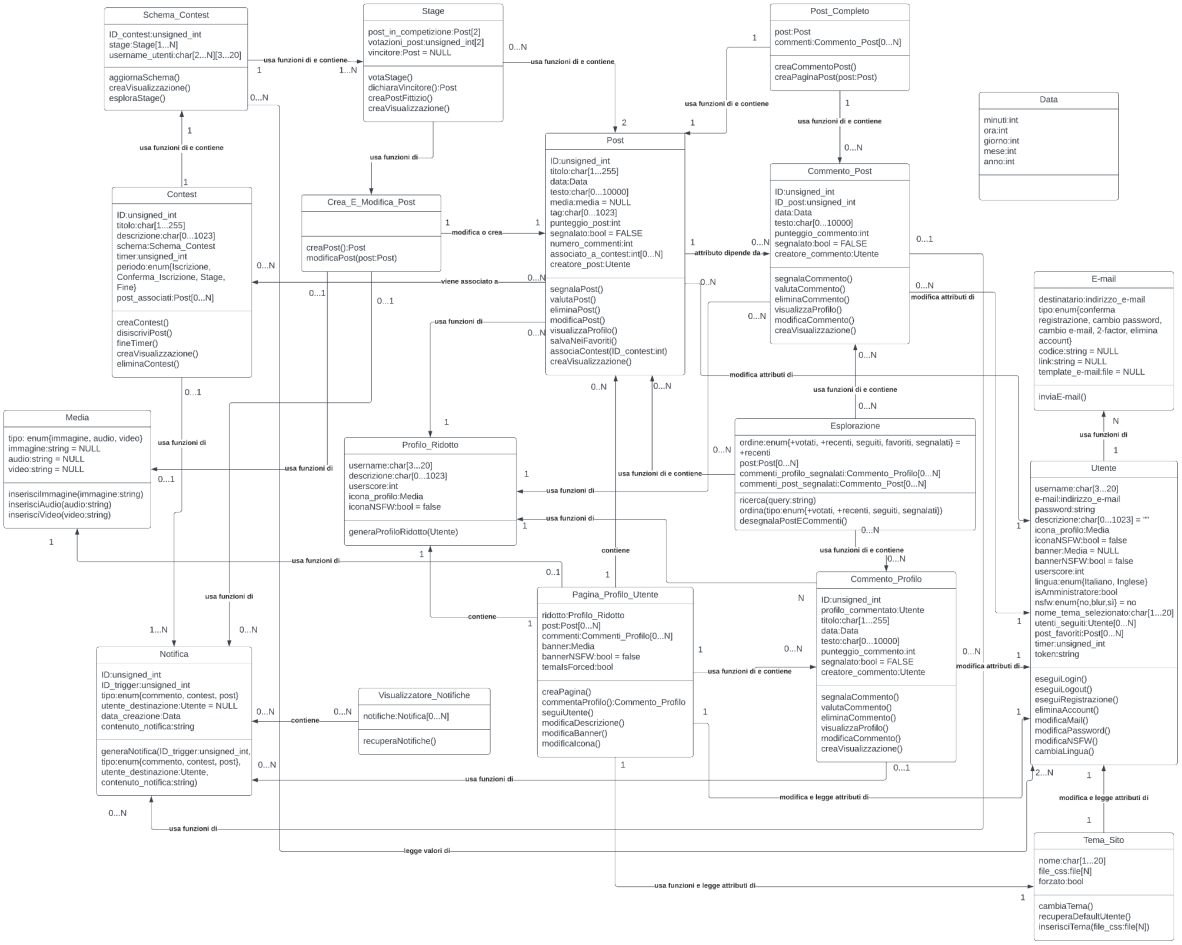
Alessandro Marostica
Nicolò Marchini
Nicolò Masellis

Indice

Scopo del Documento

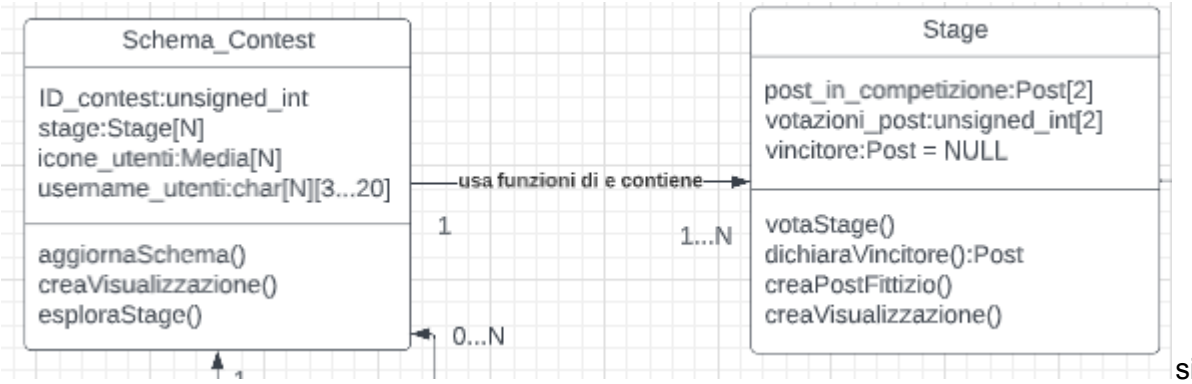
Il presente documento riporta la definizione dell'architettura del progetto EpiOpera usando diagrammi delle classi in Unified Modeling Language (UML) e codice in Object Constraint Language (OCL). Nel precedente documento è stato presentato il diagramma degli use case, il diagramma di contesto e quello dei componenti. Ora, tenendo conto di questa progettazione, viene definita l'architettura del sistema dettagliando da un lato le classi che dovranno essere implementate a livello di codice e dall'altro la logica che regola il comportamento del software. Le classi vengono rappresentate tramite un diagramma delle classi in linguaggio UML. La logica viene descritta in OCL perché tali concetti non sono esprimibili in nessun altro modo formale nel contesto di UML.

Diagramma delle Classi



Nel presente capitolo vengono presentate le classi previste nell'ambito del progetto EpiOpera. Ogni componente presente nel diagramma dei componenti è rappresentato da una o più classi. Tutte le classi individuate sono caratterizzate da un nome, una lista di attributi che identificano i dati gestiti dalla classe e una lista di metodi che definiscono le operazioni previste all'interno della classe. Ogni classe può essere anche associata ad altre classi e, tramite questa associazione, è possibile fornire informazioni su come le classi si

relazionano tra loro. In una associazione la freccia indica chi è che “riceve” l'associazione. Ad esempio questa associazione:



si deve leggere come “Uno Schema_Contest usa funzioni di e contiene uno o più Stage”. Riportiamo di seguito le classi individuate a partire dai diagrammi di contesto e dei componenti.

Utente

Il sito userà questa classe per indicare le caratteristiche di un utente all’interno del sito.
Login contiene una variabile token che indica NULL se nessun login è stato eseguito, altrimenti contiene un token univoco all’utente che ha eseguito il login.
L’attributo timer è un contatore che diminuisce di 1 ogni secondo fino al raggiungimento dello 0 che viene inizializzato a 900 quando viene eseguita la funzione eseguiLogin() o quando l’utente loggato esegue una operazione all’interno del sito, quando il timer raggiunge lo 0 viene eseguita automaticamente la funzione eseguiLogout().
L’attributo utenti_seguiti contiene una lista degli username di tutti gli utenti seguiti da un account.
L’attributo post_favoriti contiene una lista di tutti i post favoriti di un account.
La visibilità globale di questa classe ci permette di sfruttarla per poter accedere a determinate informazioni dalle quali dipende la visualizzazione di altre pagine.
La funzione eseguiLogin() viene eseguita da un utente non autenticato per diventare autenticato e può eseguire la funzione inviaE-mail() della classe [E-mail], per maggiori dettagli sulla funzione vedi lo use case [Login] del D2.
La funzione eseguiLogout() viene eseguita da un utente autenticato per diventare non autenticato o dal sito automaticamente come scritto sopra, per maggiori dettagli sulla funzione vedi lo use case [Logout] del D2.
La funzione eseguiRegistrazione() viene eseguita da un utente non autenticato per creare un account all’interno del sito ed esegue la funzione inviaE-mail() della classe [E-mail], per maggiori dettagli sulla funzione vedi lo use case [Registrazione] del D2.
La funzione eliminaAccount() viene eseguita da un utente autenticato per eliminare il proprio account (oppure da un utente amministratore per eliminare un account altrui) ed esegue la funzione inviaE-mail() della classe [E-mail], per maggiori dettagli sulla funzione vedi lo use case [Elimina Account] del D2.
La funzione modificaMail() viene eseguita da un utente autenticato per modificare la mail associata al suo account, per maggiori dettagli sulla funzione vedi lo use case [Cambio E-mail] del D2.

La funzione modificaPassword() viene eseguita da un qualsiasi utente per modificare la password associata al suo account ed esegue la funzione inviaE-mail() della classe [E-mail], per maggiori dettagli sulla funzione vedi lo use case [Cambio Password] del D2.
La funzione modificaNSFW() viene eseguita da un utente autenticato per modificare il proprio valore dell'attributo nsfw, questa funzione è presente per poter soddisfare il requisito non funzionale [Controllo visibilità post NSFW] del D2.
La funzione cambiaLingua(), per maggiori dettagli sulla funzione vedi lo use case [Cambia Lingua del Sito]. del D2.

E-mail

Una funzione fondamentale del sito è quella dello spedire e-mail informative riportanti informazioni riguardanti attività sul profilo.
Tale e-mail ha lo scopo di fornire un appoggio nei processi di login e registrazioni e saranno quindi composte di informazioni riguardo le attività del profilo associato al determinato indirizzo email.
In base al tipo di informazione che deve condividere avrà un template differente e genererà un link con una destinazione differente ed eventualmente un codice (specifico per 2FA).
Si interfaccerà poi con il Servizio E-mail per inviare le e-mail e con il Database MongoDB per ricevere il template.
La funzione inviaE-mail() viene eseguita dal sito per inviare una e-mail a un utente, per maggiori dettagli sulla funzione vedi lo use case [Invio E-mail] del D2.

Notifica

Il sito può inviare a un utente delle notifiche informative riguardanti menzioni e informazioni sui contest in corso.
Ogni notifica avrà un insieme di informazioni specifiche all'evento che lo ha generato e l'utente sarà in grado, all'interazione con essa, di reindirizzare l'utente alla pagina in cui può interagire con l'oggetto della notifica usando un link.
Un'altra classe può richiedere la generazione di una notifica usando la funzione generaNotifica(), se il tipo della notifica da generare è contest allora la notifica non ha un singolo destinatario ma viene mandata a tutti gli utenti del sito.
La funzione generaNotifica() viene eseguita dal sito per generare una notifica, per maggiori dettagli sulla funzione vedi il componente [Genera Notifica] del D2.

Visualizzatore_Notifiche

Questa classe permette ad un utente autenticato di visualizzare la lista di tutte le notifiche da lui ricevute in ordine dalla più recente.
L'attributo notifiche è una lista che contiene tutte le istanze della classe [Notifica] associate ad un utente.
Questo avviene interagendo con il Database MongoDB usando la sessione di login attuale dell'utente con la funzione recuperaNotifiche(), per maggiori dettagli riguardo la funzione vedi lo use case [Visualizza Notifiche] del D2.

Profilo_Ridotto

E' una classe specifica realizzata per la funzione del sito di visualizzare un profilo in maniera "ridotta", cioè omettendo alcune informazioni che creerebbero troppa confusione in altre pagine ma comunque accessibili con un click verso la pagina completa del profilo.
A tale proposito la classe è composta dalle informazioni base di un profilo più un link che porta alla pagina del profilo stesso.
La funzione generaProfiloRidotto() viene eseguita dal sito per creare la visualizzazione del profilo ridotto, per maggiori informazioni su questa funzione vedi l'use case [Visualizza Profilo Ridotto di un Utente] nel D2.

Pagina_Profilo_Utente

Nel sito esistono pagine specifiche alla visualizzazione completa di un profilo utente.
Questa classe sfrutta l'esistenza della soprastante classe [Profilo_Ridotto] per evitare di avere funzioni e attributi ridondanti.
L'attributo commenti è una lista di tutte le istanze della classe [Commento_Profilo] associate al profilo.
L'attributo post è una lista di tutte le istanze della classe [Post] create dall'utente proprietario della pagina profilo.
La funzione commentaProfilo() permette a un utente autenticato di creare una istanza della classe [Commento_Profilo] associato al profilo, per maggiori dettagli sulla funzione vedi lo use case [Commenta Profilo] del D2.
La funzione seguiUtente() permette a un utente autenticato di seguire (o smettere di seguire se lo stava seguendo già) il profilo visualizzato andando ad aggiungere all'attributo utenti seguiti l'username dell'utente di cui si sta visualizzando il profilo all'istanza della classe [Utente] associata all'utente autenticato, per maggiori dettagli sulla funzione vedi lo use case [Segui Utente] del D2.
La funzione creaPagina() viene eseguita dal sito per creare e far visualizzare a un utente la pagina associata ad un profilo, in particolare interagisce con la classe [Tema_Sito] se il parametro tema IsForced è true per recuperare il tema personalizzato dell'utente a cui appartiene la pagina profilo, esegue le funzioni creaVisualizzazione() delle classi [Post] e [Commento_Profilo] e la funzione generaProfiloRidotto() della classe [Profilo_Ridotto], per maggiori dettagli sulla funzione vedi lo use case [Visualizza Profilo Completo di un Utente] del D2.
La funzione modificaDescrizione() permette all'utente possessore del profilo di modificare la descrizione del suo profilo, per maggiori dettagli sulla funzione vedi lo use case [Modifica Profilo] del D2.
La funzione modificaBanner() permette all'utente possessore del profilo di modificare il banner del suo profilo interagendo con la classe [Media] e aggiornando gli attributi anche all'interno della istanza della classe [Utente] relativa all'utente che detiene la pagina profilo. Un utente amministratore può usare questa funzione anche su profili altrui ma in modo ristretto, per maggiori dettagli sulla funzione vedi lo use case [Modifica Profilo] del D2.
La funzione modificaIcona() permette all'utente possessore del profilo di modificare l'icona del suo profilo interagendo con la classe [Media], andando anche a modificare l'istanza associata a tale icona all'interno di eventuali istanze di classe [Contest] e aggiornando gli attributi anche all'interno della istanza della classe [Utente] relativa all'utente che detiene la pagina profilo. Un utente amministratore può usare questa funzione anche su profili altrui ma

in modo ristretto, per maggiori dettagli sulla funzione vedi lo use case [Modifica Profilo] del D2.

Post

Funzione centrale del sito è l'esplorazione di contenuti caricati da altri utenti, è quindi necessaria una classe Post che contenga le informazioni sul contenuto.
In particolare la classe Post si riferisce alla visualizzazione ridotta dei post che viene utilizzata da tutte quelle parti del sito che richiedono la visualizzazione di un post.
La visualizzazione di un post cambia a seconda della presenza o meno del tag #NSFW al suo interno e delle impostazioni dell'utente che ha eseguito il login (se non ci sono sessioni di login allora di default i post NSFW vengono nascosti).
L'attributo numero_commenti viene ricavato interrogando il Database MongoDB riguardo il numero di commenti associato a un post.
L'attributo associato_a_contest indica a quali istanze della classe [Contest] viene associato il post usando gli id delle istanze di [Contest].
La funzione segnalaPost() viene eseguita da un utente autenticato e segnala un post per la revisione mettendo a TRUE l'attributo segnalato, per maggiori dettagli sulla funzione vedi lo use case [Segnala Post] del D2.
La funzione valutaPost() viene eseguita da un utente autenticato che assegna un voto a un post modificando l'attributo punteggio_post, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del post, per maggiori dettagli sulla funzione vedi lo use case [Valuta Post] del D2.
La funzione eliminaPost() viene eseguita dall'utente che ha creato il post in questione o da un utente amministratore per eliminare il post dal sito, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del post, per maggiori dettagli sulla funzione vedi lo use case [Elimina Post] del D2.
La funzione visualizzaProfilo() esegue la funzione generaProfiloRidotto() della classe [Profilo_Ridotto] per mostrare a schermo il profilo ridotto dell'autore del post.
La funzione salvaNeiFavoriti() viene eseguita da un utente autenticato aggiungendo questo post alla sua lista post_favoriti all'interno della sua istanza della classe [Utente], per maggiori dettagli sulla funzione vedi lo use case [Salva Post nei Favoriti] del D2.
La funzione associaContest() viene eseguita dall'utente creatore del post che associa il post ad una istanza della classe [Contest], per maggiori dettagli sulla funzione vedi lo use case [Commenta Post] del D2.
La funzione creaVisualizzazione() viene eseguita da una parte del sito quando vuole visualizzare un post e interagisce con l'Orologio di Sistema del dispositivo che interagisce con il sito, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Post Ridotto] del D2.
La funzione modificaPost() viene eseguita dall'utente creatore del post o da un utente amministratore e richiama la funzione modificaPost() della classe [Crea_E_Modifica_Post].

Post_Completo

Questa classe serve ad unire in un unico luogo i commenti associati a un post e il post stesso.

L'attributo commenti è una lista di tutte le istanze della classe [Commento_Post] associate al profilo.
La funzione creaCommentoPost() viene eseguita da un utente autenticato che crea un'istanza della classe [Commento_Post] e la comunica al Database MongoDB, per maggiori dettagli sulla funzione vedi lo use case [Commenta Post] del D2.
La funzione creaPaginaPost() viene eseguita dal sito per creare la visualizzazione di un post completo, questa funzione richiama la funzione creaVisualizzazione() della istanza della classe [Post] che si vuole vedere e creaVisualizzazione() delle istanze della classe [Commento_Post] associate al post, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Post Completo] del D2.

Esplorazione

Questa classe si rende necessaria nella creazione delle pagine di esplorazione. L'attributo ordine viene usato per indicare l'ordine scelto dall'utente per visualizzare i risultati di una ricerca e viene modificato dalla funzione ordina().
La funzione ricerca() viene eseguita da un utente per ricercare istanze della classe [Post] all'interno del sito eseguendo anche la funzione creaVisualizzazione() della classe [Post], per maggiori dettagli sulla funzione vedi lo use case [Cerca Post] del D2.
La funzione ordina() viene eseguita da un utente per ordinare la visualizzazione delle istanze della classe [Post] all'interno di una ricerca (o, nel caso di utenti amministratore, anche istanze delle classi [Commento_Profilo] e [Commento_Post]), per maggiori dettagli sulla funzione vedi lo use case [Ordina Post] del D2.
La funzione desegnalaPostECommenti() è disponibile solo agli utenti amministratori quando usano la funzione ordina() per visualizzare istanze di [Post], [Commento_Profilo] e [Commento_Post] segnalati, per maggiori dettagli sulla funzione vedi l'use case [De-segnala Post/Commento] del D2.

Crea_E_Modifica_Post

La classe permette all'utente di creare o modificare un'istanza della classe [Post].
La funzione creaPost() viene eseguita da un utente autenticato o dal sito per generare un'istanza della classe [Post] alle volte interagendo con la classe [Media], per maggiori dettagli sulla funzione vedi l'use case [Crea Post] del D2.
La funzione modificaPost() viene eseguita da un utente autenticato per modificare un'istanza della classe [Post] alle volte interagendo con la classe [Media], per maggiori dettagli sulla funzione vedi l'use case [Modifica Post] del D2.

Contest

La classe Contest permette di creare e interagire, insieme alle classi [Schema_Contest] e [Stage], con un contest all'interno del sito.
L'attributo periodo va a indicare in che periodo si trova attualmente il contest (es. periodo sarà uguale a "Fine" se il contest è concluso).
L'attributo timer è un contatore che diminuisce di 1 ogni secondo fino al raggiungimento dello 0 e che sta ad indicare quanto tempo manca fino alla fine del periodo attuale.

L'attributo post_associati contiene una lista di tutte le istanze della classe [Post] associate al contest.
La funzione creaContest() viene eseguita da un utente amministratore per indire un contest per maggiori dettagli sulla funzione vedi l'use case [Crea Contest] del D2.
La funzione disiscriviPost() viene eseguita da un utente amministratore durante il periodo di Conferma_Iscrizione escludendo una istanza della classe [Post] dal contest, per maggiori dettagli sulla funzione vedi lo use case [Disiscrivi Post] del D2.
La funzione fineTimer() viene eseguita dal sito quando il timer raggiunge 0, può avere diversi effetti a seconda del periodo attuale del contest ed allo stato attuale della istanza di [Schema_Contest] associata al contest, andando anche alle volte a chiamare la funzione aggiornaSchema() della classe [Schema_Contest] o a creare una istanza di questa, per maggiori dettagli sulla funzione vedi gli use case [Fine Periodo Iscrizione], [Fine Periodo Conferma Iscrizione], [Fine Stage] e [Fine Contest] del D2.
La funzione creaVisualizzazione() viene eseguita da un utente per poter visualizzare la pagina del contest e può eseguire le funzioni creaVisualizzazione() delle classi [Post] e [Schema_Contest], per maggiori dettagli sulla funzione vedi lo use case [Visualizza Contest] del D2.
La funzione eliminaContest() viene eseguita da un utente amministratore per eliminare un contest, per maggiori dettagli sulla funzione vedi lo use case [Elimina Contest] del D2.

Schema_Contest

La classe contiene tutte le informazioni necessarie alla corretta visualizzazione dell'intera struttura del contest e alla navigazione dello stesso.
L'attributo stage contiene una lista di tutte le istanze della classe [Stage] che sono attualmente presenti nel contest.
L'attributo username_utenti contiene una lista di tutti gli utenti che sono iscritti al contest.
La funzione aggiornaSchema() viene eseguita dal sito quando si conclude un periodo di Stage, eseguendo la funzione dichiaraVincitore() per ogni istanza della classe [Stage] non ancora conclusa del contest e andando a creare nuove istanze della classe [Stage] ove necessario, per maggiori dettagli sulla funzione vedi lo use case [Fine Stage] del D2.
La funzione creaVisualizzazione() viene eseguita dal sito per creare una visualizzazione dello schema andando anche a leggere le icone profilo dalle istanze della classe [Utente] che partecipano alla competizione e i dati delle istanze della classe [Stage] attualmente presenti nel contest, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Schema di Sfide, Vittorie e Sconfitte] del D2.
La funzione esploraStage() viene eseguita da un utente andando a cliccare sulla visualizzazione dello schema contest per selezionare l'istanza della classe [Stage] che vuole visualizzare, eseguendo la funzione creaVisualizzazione() dell'istanza della classe [Stage] selezionata dall'utente (ma anche eseguendo nuovamente la funzione creaVisualizzazione() della istanza della classe Schema_Contest appartenente al contest che si sta visualizzando), per maggiori dettagli sulla funzione vedi lo use case [Visualizza Stage] del D2.

Stage

La classe stage gestisce l'entità minima della competizione all'interno del contest. Descrive il duello in corso tra due post e tiene traccia delle votazioni eseguite dagli utenti.
La funzione votaStage() viene eseguita dagli utenti autenticati per assegnare un unico voto ad uno dei due post in competizione, per maggiori dettagli sulla funzione vedi lo use case [Vota Stage] del D2.
La funzione dichiaraVincitore() viene eseguita dal sito per concludere lo stage e dichiarare un vincitore, per maggiori dettagli sulla funzione vedi lo use case [Genera Schema di Sfide, Vittorie e Sconfitte] del D2.
La funzione creaPostFittizio() viene eseguita dal sito e crea, usando la funzione creaPost() della classe [Crea_E_Modifica_Post], un post segnaposto nel caso in cui uno dei post partecipanti dovessero venire eliminati dal sito, per maggiori dettagli sulla funzione vedi lo use case [Elimina Post] del D2.
La funzione creaVisualizzazione() viene eseguita dal sito e crea una visualizzazione dello stato attuale di uno stage del contest, usando anche la funzione creaVisualizzazione delle istanze della classe [Post] che partecipano allo stage, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Stage] del D2.

Commento_Profilo

Questa classe permette agli utenti di lasciare commenti a un profilo utente, si differenzia dalla classe [Commento_Post] anche per la presenza dell'attributo titolo.
La funzione segnalaCommento() viene eseguita da un utente autenticato e segnala un commento per la revisione mettendo a TRUE l'attributo segnalato, per maggiori dettagli sulla funzione vedi lo use case [Segnala Commento] del D2.
La funzione valutaCommento() viene eseguita da un utente autenticato e permette di assegnare un voto unico a un commento, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del commento, per maggiori dettagli sulla funzione vedi lo use case [Valuta Commento] del D2.
La funzione eliminaCommento() viene eseguita da un utente autenticato che possiede il commento(ed eventualmente da un utente amministratore su un commento altrui) per rimuovere il commento, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del commento, per maggiori dettagli sulla funzione vedi lo use case [Elimina Commento] del D2.
La funzione visualizzaProfilo() esegue la funzione generaProfiloRidotto() della classe [Profilo_Ridotto] per mostrare a schermo il profilo ridotto dell'autore del commento.
La funzione modificaCommento() permette dall'utente autenticato che ha creato il commento di modificare i contenuti del commento, per maggiori dettagli sulla funzione vedi lo use case [Modifica Commento] del D2.
La funzione creaVisualizzazione() viene eseguita dal sito per mostrare a schermo il commento e interagisce con l'Orologio di Sistema del dispositivo che interagisce con il sito, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Commento] del D2.

Commento_Post

Questa classe permette agli utenti di lasciare commenti a un post, si differenzia dalla classe [Commento_Profilo] anche per l'assenza dell'attributo titolo.
La funzione segnalaCommento() viene eseguita da un utente autenticato e segnala un commento per la revisione mettendo a TRUE l'attributo segnalato, per maggiori dettagli sulla funzione vedi lo use case [Segnala Commento] del D2.
La funzione valutaCommento() viene eseguita da un utente autenticato e permette di assegnare un voto unico a un commento, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del commento, per maggiori dettagli sulla funzione vedi lo use case [Valuta Commento] del D2.
La funzione eliminaCommento() viene eseguita da un utente autenticato che possiede il commento(ed eventualmente da un utente amministratore su un commento altrui) per rimuovere il commento, questa funzione va anche a modificare l'attributo userscore della istanza della classe [Utente] associata all'utente autore del commento, per maggiori dettagli sulla funzione vedi lo use case [Elimina Commento] del D2.
La funzione visualizzaProfilo() esegue la funzione generaProfiloRidotto() della classe [Profilo_Ridotto] per mostrare a schermo il profilo ridotto dell'autore del commento.
La funzione modificaCommento() permette dall'utente autenticato che ha creato il commento di modificare i contenuti del commento, per maggiori dettagli sulla funzione vedi lo use case [Modifica Commento] del D2.
La funzione creaVisualizzazione() viene eseguita dal sito per mostrare a schermo il commento e interagisce con l'Orologio di Sistema del dispositivo che interagisce con il sito, per maggiori dettagli sulla funzione vedi lo use case [Visualizza Commento] del D2.

Tema_Sito

Questa classe gestisce la visualizzazione da parte di tutto il sito dei temi.
Si occupa anche di permettere ad un utente autenticato di aggiungere un tema personalizzato al sito.
La classe cambiaTema() viene eseguita da un utente qualsiasi per applicare un tema alla visualizzazione del sito andando a modificare l'attributo nome_tema_selezionato della classe [Utente], per maggiori dettagli sulla funzione vedi lo use case [Cambia Tema del Sito] del D2.
La classe recuperaDefaultUtente() viene eseguita dal sito per recuperare il tema che l'utente ha selezionato come default per la visualizzazione delle pagine del sito.
La classe inserisciTema() viene eseguita da un utente autenticato per caricare dei file css contenenti un tema personalizzato per il sito, per maggiori dettagli sulla funzione vedi lo use case [Carica Tema Personalizzato] del D2.

Media

Questa classe gestisce le informazioni riguardante i media non testuali presenti nel sito.
La funzione inserisciImmagine() viene eseguita da un utente autenticato per inserire una immagine nel sito, per maggiori dettagli sulla funzione vedi lo use case [Inserisci Media] del D2.
La funzione inserisciAudio() viene eseguita da un utente autenticato per inserire un audio nel sito, per maggiori dettagli sulla funzione vedi lo use case [Inserisci Media] del D2.

La funzione inserisciVideo() viene eseguita da un utente autenticato per inserire un video nel sito, per maggiori dettagli sulla funzione vedi lo use case [Inserisci Media] del D2.

Data

Informazioni principali riguardanti numerosi contenuti sono data e ora di pubblicazione o modifica, viene quindi creata una classe Data contenente tali informazioni.

OCL

Profilo_Ridotto

La classe necessita che il media dell'immagine di profilo sia esclusivamente un'immagine.

context Profilo_Ridotto : self.icona_profilo.tipo = immagine

Post_Completo

Il commento può essere creato esclusivamente da un utente autenticato.

context Post_Completo :: creaCommentoPost()
pre: Utente.token != NULL

Esplorazione

Nella pagina di esplorazione un utente amministratore può visualizzare una lista dei post segnalati, questo è quindi un requisito, un utente qualsiasi può invece visualizzare una lista di post in base all'ordine selezionato eccetto per l'ordine "seguiti" che può essere selezionato solo da utenti autenticati.

context Esplorazione :: ordina()
pre: if (!Utente.isAmministratore AND self.ordine = segnalati) then (self.ordine = +recenti)

context Esplorazione :: ordina()
pre: if (Utente.token = NULL AND self.ordine = seguiti) then (self.ordine = +recenti)

context Esplorazione :: desegnalaPostECommenti()
pre: Utente.isAmministratore

Pagina_Profilo_Utente

Il banner del profilo utente deve essere un'immagine, per poter seguire un profilo l'utente deve essere autenticato.
context Pagina_Profilo_Utente : self.banner.tipo = immagine

context Pagina_Profilo_Utente :: seguiUtente()

pre: Utente.token != NULL

context Pagina_Profilo_Utente :: modificaDescrizione()
pre: Utente.username = self.ridotto.username

context Pagina_Profilo_Utente :: modificaBanner()
pre: Utente.username = self.ridotto.username OR Utente.isAmministratore

context Pagina_Profilo_Utente :: modificaIcona()
pre: Utente.username = self.ridotto.username OR Utente.isAmministratore

Crea_E_Modifica_Post

Per creare un post un utente deve essere autenticato, inoltre per modificare un post il nome dell'utente deve corrispondere al nome dell'autore del post oppure l'utente deve essere amministratore.

context Crea_E_Modifica_Post :: creaPost()
pre: Utente.token != NULL

context Crea_E_Modifica_Post :: modificaPost()
pre: Utente = self.post.creatore_post OR Utente.isAmministratore

Media

La particolare combinazione di attributi all'interno della classe permette di stabilire il tipo del media inserito.

context Media:
if (tipo = immagine) then (immagine != NULL AND audio = NULL AND video = NULL)
else if (tipo = audio) then (immagine = NULL AND audio != NULL AND video = NULL)
else if (tipo = video) then (immagine = NULL AND audio = NULL AND video != NULL)

context Media :: inserisciImmagine(immagine:string)
post: self.tipo = immagine AND self.immagine != NULL

context Media :: inserisciAudio(audio:string)
post: self.tipo = audio AND self.audio != NULL

context Media :: inserisciVideo(video:string)
post: self.tipo = video AND self.video != NULL

Contest

Solo un utente amministratore può creare, eliminare o disiscrivere post da un contest, inoltre quest'ultimo può avvenire solo durante il periodo di conferma iscrizione.

context Contest :: creaContest()
pre: Utente.isAmministratore
post: self.timer >= 86400 AND self.timer <= 2678400

context Contest :: disiscriviPost()
pre: Utente.isAmministratore AND self.periodo = Conferma_Iscrizione

context Contest :: eliminaContest()
pre: Utente.isAmministratore

Schema_Contest

La classe necessita che il media dell'immagine di profilo sia esclusivamente un'immagine.

context Schema_Contest : icone_utenti[N].tipo = immagine

Stage

Solo un utente autenticato può votare uno stage.

context Stage :: votaStage()
pre: Utente.token != NULL

Tema_Sito

La personalizzazione del tema può essere effettuata solo da un utente autenticato.

context Tema_Sito :: inserisciTema(file_css:file[N])
pre: Utente.token != NULL

Utente

Un utente autenticato può modificare solamente la propria mail, password e NSFW.
Inoltre un utente autenticato può eliminare solamente il proprio account.
Tuttavia un utente amministratore può eliminare un account altrui.
Infine solo un utente non autenticato può eseguire la registrazione al sito o il login e solo un utente autenticato può eseguire il logout.

context Utente : icona_profilo.tipo = immagine

context Utente : banner.tipo = immagine

context Utente :: eseguiLogin()
pre: self.utente.token = NULL
post: self.utente.token != NULL

context Utente :: eseguiLogout()
pre: self.utente.token != NULL
post: self.utente.token = NULL

context Utente :: eseguiRegistrazione()
pre: self.utente.token = NULL

context Utente :: eliminaAccount()
pre: self.token != NULL OR self.isAmministratore

context Utente :: modificaMail()
pre: self.token != NULL

context Utente :: modificaPassword()
pre:self.token != NULL

context Utente :: modificaNSFW()
pre:self.token != NULL

Commento_Profilo

La creazione, la modifica e l'interazione con i commenti possono essere effettuate solo da utenti autenticati e, ove necessario, sotto la specifica condizione di proprietà del commento. Un utente amministratore inoltre ha la possibilità di rimuovere commenti al fine della moderazione.

context Commento_Profilo :: eliminaCommento()
pre: Utente = self.creatore_commento OR Utente.isAmministratore

context Commento_Profilo :: modificaCommento()
pre: Utente = self.creatore_commento

context Commento_Profilo :: segnalaCommento()
pre: Utente.token != NULL
post: self.segnalato = true

context Commento_Profilo :: valutaCommento()
pre: Utente.token != NULL

Commento_Post

La creazione, la modifica e l'interazione con i commenti possono essere effettuate solo da utenti autenticati e, ove necessario, sotto la specifica condizione di proprietà del commento. Un utente amministratore inoltre ha la possibilità di rimuovere commenti al fine della moderazione.

context Commento_Post :: eliminaCommento()

pre: Utente = self.creatore_commento OR Utente.isAmministratore

context Commento_Post :: modificaCommento()
pre: Utente = self.creatore_commento

context Commento_Post :: segnalaCommento()
pre: Utente.token != NULL
post: self.segnalato = true

context Commento_Post :: valutaCommento()
pre: Utente.token != NULL

Post

La creazione e l'interazione con i post possono essere effettuate solo da utenti autenticati e, ove necessario, sotto la specifica condizione di proprietà del post.
Un utente amministratore ha inoltre la possibilità di rimuovere post al fine della moderazione.

context Post :: eliminaPost()
pre: Utente = self.creatore_post OR Utente.isAmministratore

context Post :: segnalaPost()
pre: Utente.token != NULL
post: self.segnalato = true

context Post :: valutaPost()
pre: Utente.token != NULL

context Post :: salvaNeiFavoriti()
pre: Utente.token != NULL

context Post :: associaContest()
pre: Utente = self.creatore_post

Visualizzatore_Notifiche

Solo un utente autenticato può visualizzare le sue notifiche.

context Visualizzatore_Notifiche :: recuperaNotifiche()
pre: Utente.token != NULL