

Indice

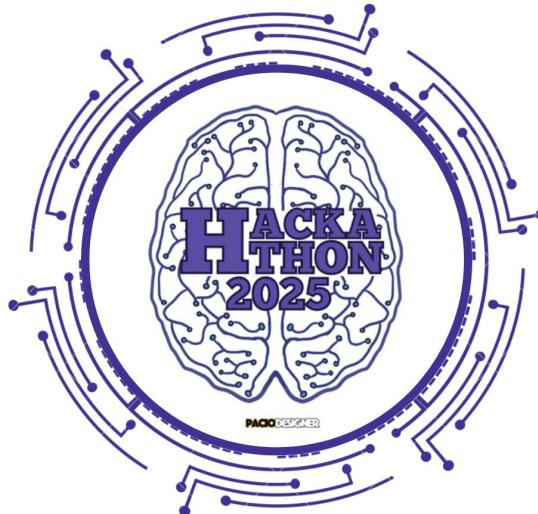
1	Introduzione	3
2	Glossario	4
3	User Requirements Definition	5
3.1	Use Case Giudice	5
3.1.1	Diagramma	5
3.1.2	Documentazione	6
3.2	Use Case Organizzatore	9
3.2.1	Diagramma	9
3.2.2	Documentazione	10
3.3	Use Case Utente Registrato	12
3.3.1	Diagramma	12
3.3.2	Documentazione	12
3.4	Use Case Utente Non Registrato	14
3.4.1	Diagramma	14
3.4.2	Documentazione	14
3.5	Use Case Squadra	16
3.5.1	Diagramma	16
3.5.2	Documentazione	16
4	System Requirements	19
4.1	Requisiti Funzionali	19
4.2	Requisiti Non Funzionali	20
4.3	Requisiti di Dominio	21
5	System Architectural Models	22
5.1	Activity Diagrams	22
5.1.1	Activity Diagrams Giudice	22
5.1.2	Activity Diagrams Organizzatore	27
5.1.3	Activity Diagrams Utente Registrato	29
5.1.4	Activity Diagrams Utente Non Registrato	32
5.1.5	Activity Diagrams Team	34
5.2	Sequence Diagrams	39

5.2.1	Sequence Diagrams Giudice	39
5.2.2	Sequence Diagrams Organizzatore	41
5.2.3	Sequence Diagrams Utente Registrato	42
5.2.4	Sequence Diagrams Utente Non Registrato	44
5.2.5	Sequence Diagrams Team	45
5.3	Class Diagrams	48
5.3.1	Class Diagram Unrefined	48
5.3.2	Class Diagram Refined	49
6	Design Patterns	50
6.1	Observer	50
6.2	Decorator	51
6.3	Strategy	52
6.4	Factory	53
6.5	Singleton	53

HACKATHON

Giugno, 2025

Nome	Email	Matricola
Giorgia Pazienza	giorgia.pazienza@students.uniroma2.eu	0327936
Mario Chiappini	mario.chiappini@students.uniroma2.eu	0322135
Pietro Galotto	pietro.galotto@students.uniroma2.eu	0325553
Francesco Cosciotti	francesco.cosciotti@students.uniroma2.eu	0323545
Ionut Georgian Zbirciog	ionutgeorgian.zbirciog@students.uniroma2.eu	0308984



1 Introduzione

Nel corso dei nostri studi in Informatica, abbiamo avuto l'opportunità di confrontarci con sfide che mettono davvero alla prova le nostre capacità di problem solving e programmazione. Uno tra questi è l'hackathon, una competizione della durata di 18 ore in modalità full immersion, durante la quale i partecipanti, suddivisi in team, devono risolvere problemi complessi, mettendo alla prova le proprie capacità in ambito problem solving.

Alla base di questo evento vi è un'organizzazione dettagliata e strutturata, finalizzata a garantire il corretto svolgimento della competizione. Al fine di migliorare e semplificare la gestione degli hackathon universitari, si propone lo sviluppo di un software dedicato, capace di supportare l'intero processo organizzativo in modo efficiente ed efficace.

Il software è progettato per creare un sistema completo, dove il comitato organizzatore può definire le sfide, gestire le iscrizioni, monitorare in tempo reale i progressi dei team e valutare le soluzioni grazie all'intervento di giudici esperti.

All'interno di questa piattaforma software si avrà una classifica dinamica, dove ogni aggiornamento modifica le sorti della classifica.

- Gli organizzatori della competizione, singole persone identificate come un'unica entità avente come mansione la stesura delle sfide, gestione della giornata, relazioni con gli sponsor etc... . Durante l'evento questi si occuperanno della gestione generale del campionato e del coordinamento delle attività.
- I partecipanti alla competizione, devono confermare la propria partecipazione all'evento iscrivendosi e formando un team o partecipando ad uno già esistente.
- Le squadre sono formate da 1 fino ad un massimo di 4 utenti registrati che collaboreranno fra di loro per la risoluzione dei problemi.
- Gli utenti non registrati possono accedere ad una sezione pubblica, in cui possono consultare il calendario degli hackathon e i risultati degli eventi passati.
- Il giudice ha come ruolo principale all'interno della competizione quello di valutare e assegnare punteggi/penalità alle soluzioni presentate dalle varie squadre. Si interfaccia con il sistema pubblicando i risultati e nell'eventualità di una contestazione presentare ha l'obbligo di esaminarla.

L'obiettivo principale della piattaforma è quello di automatizzare le principali attività organizzative, riducendo il rischio di errori e rendendo l'evento più fluido e coinvolgente. Infatti, la piattaforma offrirà la possibilità di monitorare in tempo reale i progressi e la posizione in classifica, incentivando così la partecipazione attiva e competitiva.

2 Glossario

Termine	Sinonimo	Descrizione
Organizzatore	Admin	Utente con privilegi elevati per la gestione del sistema.
Utente	Partecipante	Qualsiasi persona che interagisce con la piattaforma.
Utente Autenticato	Partecipante	Utente che ha effettuato login e ha sessione valida.
Competizione	Evento	Evento in cui utenti o team si sfidano risolvendo problemi.
Modulo	Componente	Componente software autonomo con interfacce definite.
Contestazione	Ricorso	Richiesta di revisione di un risultato o decisione.
Credenziali	Dati di accesso	Dato che attesta l'identità o i permessi di un utente.
Valutazione	Giudizio	Processo di giudizio di una soluzione o performance.
Partecipante	Concorrente, Utente	Persona che prende parte a un evento o competizione.
Soluzione	Risoluzione	Idea proposta da una squadra sulla base delle richieste del problema.

Tabella 1: Glossario dei termini utilizzati nel documento

3 User Requirements Definition

3.1 Use Case Giudice

3.1.1 Diagramma

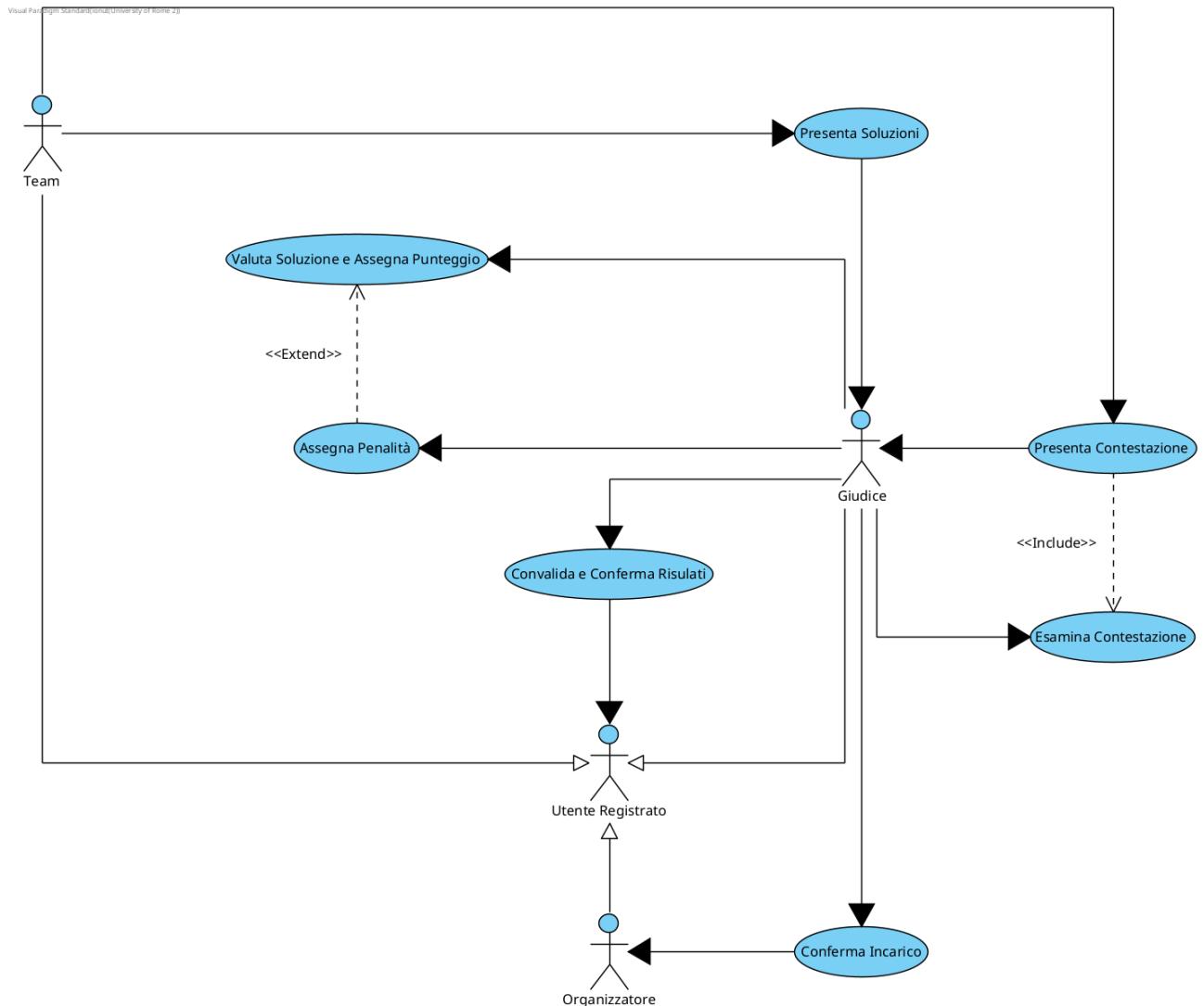


Figura 1: Use Case "Giudice"

3.1.2 Documentazione

Use Case: Valutazione soluzione del team

Use case	Valutazione soluzione del team	
Descrizione	Il giudice valuta la soluzione proposta da un team e assegna un punteggio in base ai criteri stabiliti. Questo processo è fondamentale per aggiornare la classifica e premiare le soluzioni migliori.	
Attori	Giudice, Team	
Precondizioni	Il team ha presentato la propria soluzione. Il giudice ha effettuato l'autenticazione sulla piattaforma.	
Scenario principale	Passo	Azione
	1	Il giudice effettua il login; il sistema verifica le credenziali e consente l'accesso alla dashboard.
	2	Il giudice visualizza la lista dei team in gara e seleziona il team da valutare.
	3	Il sistema mostra i dettagli della soluzione; il giudice analizza il contenuto, verificando i criteri tecnici e qualitativi.
	4	Il giudice inserisce il punteggio; il sistema registra il punteggio e aggiorna la classifica.
	5	Il sistema notifica al team l'esito della valutazione.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Il team non ha presentato la soluzione, il sistema visualizza un messaggio d'errore o avvisa il giudice.
Post-condizioni	Il punteggio del team viene aggiornato e memorizzato. La classifica viene modificata in base alla nuova valutazione. La valutazione viene registrata per eventuali controlli futuri.	

Use Case: Assegnazione di penalità

Use case	Assegnazione di penalità	
Descrizione	Il giudice rileva comportamenti scorretti o infrazioni alle regole da parte di un team e, di conseguenza, assegna una penalità che riduce il punteggio del team.	
Attori	Giudice, Team	
Precondizioni	Il team è attivo nella competizione. Il giudice ha accesso alle informazioni sulle attività e le soluzioni del team.	
Scenario principale	Passo	Azione
	1	Il giudice accede al profilo e alla cronologia delle attività del team.
	2	Durante l'analisi, il giudice rileva un comportamento non conforme (es. uso di codice non autorizzato).
	3	Il giudice seleziona l'opzione per assegnare una penalità.
	4	Il giudice inserisce il valore della penalità e specifica il motivo dell'infrazione.
	5	Il sistema registra la penalità, sottrae punti dal punteggio del team e aggiorna la classifica.
	6	Il team riceve una notifica riguardo l'assegnazione della penalità.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Se il giudice ritiene necessario approfondire la situazione, può sospendere temporaneamente l'assegnazione della penalità e avviare una procedura di riesame, notificando il team dell'avvio della verifica.
Post-condizioni	Il punteggio del team viene aggiornato in base alla penalità. La penalità viene registrata nel sistema per eventuali contestazioni future. Il team è informato ufficialmente della penalità.	

Use Case: Riesame delle contestazioni

Use case	Riesame delle contestazioni	
Descrizione	Il giudice gestisce le richieste di riesame presentate dai team che contestano le decisioni (sia di valutazione che di penalità), garantendo una revisione trasparente e oggettiva.	
Attori	Giudice, Team	
Precondizioni	Un team ha formalmente presentato una contestazione tramite la piattaforma. La richiesta contiene motivazioni ed evidenze.	
Scenario principale	Passo	Azione
	1	Il team invia la richiesta di riesame. Il giudice riceve una notifica.
	2	Il giudice esamina attentamente le evidenze.
	3	Il giudice decide se mantenere la decisione o modificarla (eventualmente aggiornando il punteggio o rimuovendo la penalità).
	4	Il sistema aggiorna la valutazione/penalità e notifica il team dell'esito.
	5	La documentazione del processo viene archiviata per future verifiche.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Se la contestazione è ritenuta infondata, il giudice respinge la richiesta; il sistema registra il rigetto e notifica il team.
Post-condizioni	La decisione finale viene registrata e integrata nel sistema. Il team riceve una comunicazione ufficiale dell'esito. La documentazione è conservata per garantire trasparenza e tracciabilità.	

Use Case: Conferma Incarico

Use case	Conferma Incarico	
Descrizione	Il giudice manifesta la propria disponibilità e accetta l'incarico di operare durante una specifica giornata in cui si svolge la competizione, formalizzando il proprio impegno a coprire l'evento.	
Attori	Giudice, Organizzatori	
Precondizioni	Il giudice ha ricevuto la richiesta di incarico per una determinata data e competizione. Il giudice ha accesso alla piattaforma e può visualizzare il calendario degli eventi.	
Scenario principale	Passo	Azione
	1	Il sistema invia una notifica o invito contenente data, orario e dettagli della competizione.
	2	Il giudice consulta il proprio calendario per verificare la disponibilità.
	3	Il giudice conferma la propria disponibilità e accetta l'incarico, eventualmente aggiungendo note.
	4	Il sistema registra la conferma, aggiornando lo stato dell'evento e associando il giudice all'incarico.
	5	Il sistema invia una notifica agli organizzatori confermando la copertura dell'evento.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Se il giudice non è disponibile, può rifiutare l'incarico; il sistema notifica il rifiuto agli organizzatori, che potranno cercare un sostituto o riorganizzare la copertura.
Post-condizioni	L'incarico viene registrato come confermato per la data indicata. Gli organizzatori ricevono la conferma e pianificano l'evento in base alla disponibilità del giudice.	

Use Case: Pubblicazione dei risultati

Use case	Pubblicazione dei risultati	
Descrizione	Dopo aver completato le valutazioni e le eventuali penalità, il giudice autorizza la pubblicazione dei risultati della competizione, rendendoli disponibili agli utenti registrati.	
Attori	Giudice, Utenti	
Precondizioni	Tutte le valutazioni, penalità e contestazioni sono state completate e registrate. Il giudice ha verificato la correttezza dei dati e approvato la pubblicazione.	
Scenario principale	Passo	Azione
	1	Il giudice esamina il riepilogo finale delle valutazioni e penalità.
	2	Il giudice seleziona l'opzione per pubblicare i risultati, eventualmente allegando commenti.
	3	Il sistema pubblica i risultati, crea la classifica finale e ne evidenzia i dettagli.
	4	Il sistema invia una notifica a tutti gli utenti registrati.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Se vengono riscontrate discrepanze, la pubblicazione viene sospesa e il giudice avvia una procedura di revisione, bloccando temporaneamente la visualizzazione dei risultati.
Post-condizioni	I risultati sono ufficialmente pubblicati e consultabili. Il sistema registra data e ora della pubblicazione. Gli utenti possono accedere a una documentazione dettagliata dei punteggi e delle penalità.	

3.2 Use Case Organizzatore

3.2.1 Diagramma

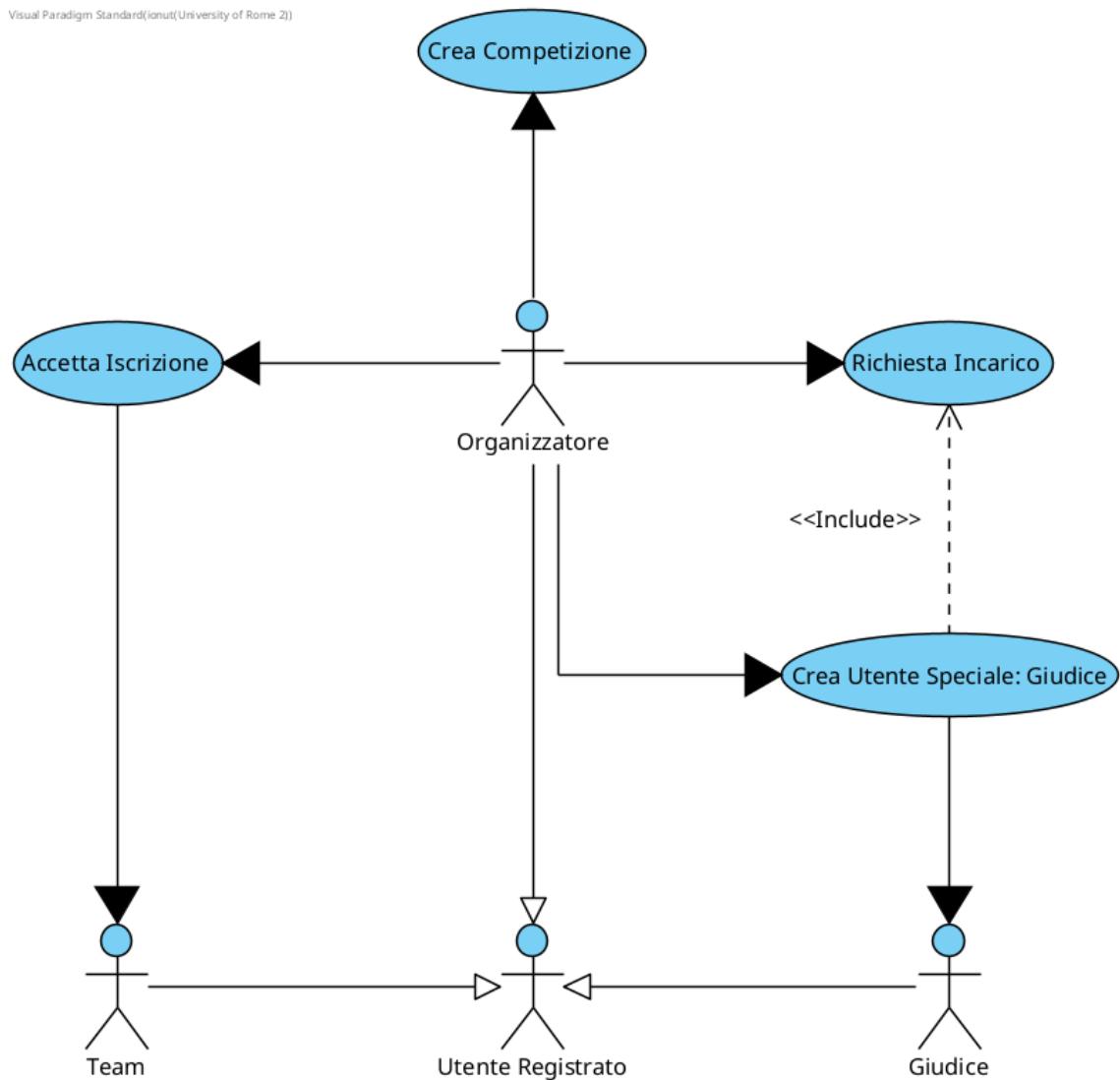


Figura 2: Use Case "Organizzatore"

3.2.2 Documentazione

Use Case: Crea Utente Speciale: Giudice

Use case	Crea Utente Speciale: Giudice	
Descrizione	L'organizzatore crea nel sistema l'account speciale del giudice.	
Attori	Organizzatore, Giudice	
Precondizioni	È stata inviata la richiesta di incarico ed il giudice scelto ha confermato l'incarico.	
Scenario principale	Passo	Azione
	1	L'organizzatore genera le credenziali del giudice e le aggiunge nel sistema.
	2	Il sistema registra l'account del giudice.
	3	Viene inviata una notifica al giudice che le sue credenziali.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Viene creato l'account senza i privilegi dell' utente Giudice. In questo caso l'organizzatore elimina l'account e lo ricrea in modo corretto.
Post-condizioni	Il giudice viene inserito nel sistema.	

Use Case: Crea Competizione

Use case	Crea Competizione	
Descrizione	L'organizzatore crea una nuova edizione della competizione	
Attori	Organizzatore, Team	
Precondizioni	Deve essere creata una nuova competizione.	
Scenario principale	Passo	Azione
	1	L'organizzatore inserisce nel sistema una nuova edizione dell'hackathon assegnandogli un nome.
	2	L'organizzatore registra la data della competizione.
	3	L'organizzatore registra la location della competizione.
	4	L'organizzatore registra i problemi per la competizione.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	L'organizzatore assegna all'edizione un nome già utilizzato. Il sistema quindi restituisce un messaggio di errore.
Post-condizioni	Competizione creata, le squadre possono richiedere l'iscrizione.	

Use Case: Accetta Iscrizione

Use case	Accetta Iscrizione	
Descrizione	L'organizzatore accetta o meno la richiesta di una squadra di partecipare.	
Attori	Organizzatore, Team	
Precondizioni	Una squadra ha inviato una richiesta di partecipazione.	
Scenario principale	Passo	Azione
	1	L'organizzatore accetta la richiesta di partecipazione da parte della squadra.
	2	Il sistema registra la squadra nella competizione.
	3	Viene inviata una notifica alla squadra .
Scenari alternativi	Passo	Descrizione
Scenario 1	1	L'organizzatore rifiuta la richiesta di partecipazione da parte della squadra.
Post-condizioni	La squadra è iscritta alla competizione e inserita nel sistema.	

3.3 Use Case Utente Registrato

3.3.1 Diagramma

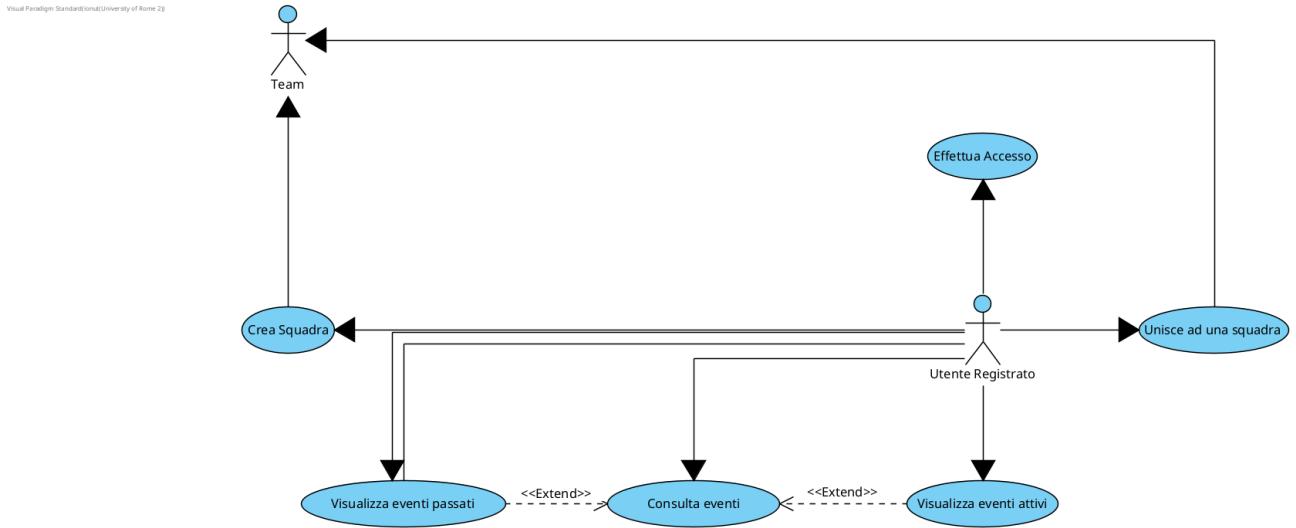


Figura 3: Use Case "Utente Registrato"

3.3.2 Documentazione

Use Case: Effettua Accesso

Use case	Effettua Accesso	
Descrizione	L'utente registrato effettua l'accesso con le proprie credenziali sulla piattaforma.	
Attori	Utente Registrato	
Precondizioni	L'utente registrato, durante la registrazione, ha acconsentito al trattamento dei dati personali.	
Scenario principale	Passo	Azione
2	1	L'utente richiede la pagina principale.
	2	Il sistema visualizza la pagina di accesso. L'utente registrato può selezionare la voce "login".
	3	L'utente registrato effettua il login con le proprie credenziali.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	L'utente inserisce delle credenziali d'accesso errate, gli vengono forniti altri due tentativi prima che venga messo in "timeout" dal servizio per 15 minuti di tempo.
Scenario 2	2	L'utente seleziona la voce "recupera password", un messaggio di recupero verrà inviato all'indirizzo email associato al suo account
Post-condizioni	L'utente effettua con successo il login nell'applicazione, l'utente già loggato può effettuare il logout.	

Use Case: Crea Squadra

Use case	Crea Squadra	
Descrizione	L'utente registrato crea una propria squadra per poter partecipare alla competizione.	
Attori	Utente Registrato, Team	
Precondizioni	L'utente è autenticato nel sistema.	
Scenario principale	Passo	Azione
	1	L'utente richiede la pagina per creazione della squadra.
	2	Il sistema mostra il form di creazione squadra.
	3	L'utente inserisce il nome della squadra e conferma.
	4	Il sistema verifica la disponibilità del nome e crea la squadra.
	5	Il sistema conferma all'utente l'avvenuta creazione della squadra.
Scenari alternativi	Passo	Descrizione
Scenario 1	4	Il nome della squadra è già esistente. Il sistema mostra un messaggio di errore.
Scenario 2	3	L'utente lascia il campo del nome vuoto. Il sistema richiede di completarlo.
Post-condizioni	La squadra è registrata nel sistema e associata all'utente registrato	

Use Case: Unisce ad una squadra

Use case	Unisce ad una squadra	
Descrizione	L'utente riceve un'invito per unirsi ad una squadra e accetta.	
Attori	Utente Registrato, Team	
Precondizioni	L'utente è autenticato nel sistema e non fa parte già di un'altra squadra.	
Scenario principale	Passo	Azione
	1	Il sistema notifica all'utente che ha ricevuto un invito per unirsi a una squadra.
	2	L'utente apre la notifica e visualizza i dettagli dell'invito.
	3	L'utente accetta l'invito.
	4	Il sistema aggiunge l'utente alla squadra e aggiorna lo stato del team.
	5	Il sistema conferma l'avvenuta unione all'utente.
Scenari alternativi	Passo	Descrizione
Scenario 1	3	L'utente rifiuta l'invito. Il sistema registra il rifiuto e non modifica la squadra.
Scenario 2	4	L'invito è scaduto o non più valido. Il sistema notifica l'utente dell'impossibilità di unirsi.
Post-condizioni	L'utente fa parte della squadra a cui ha accettato di unirsi.	

3.4 Use Case Utente Non Registrato

3.4.1 Diagramma

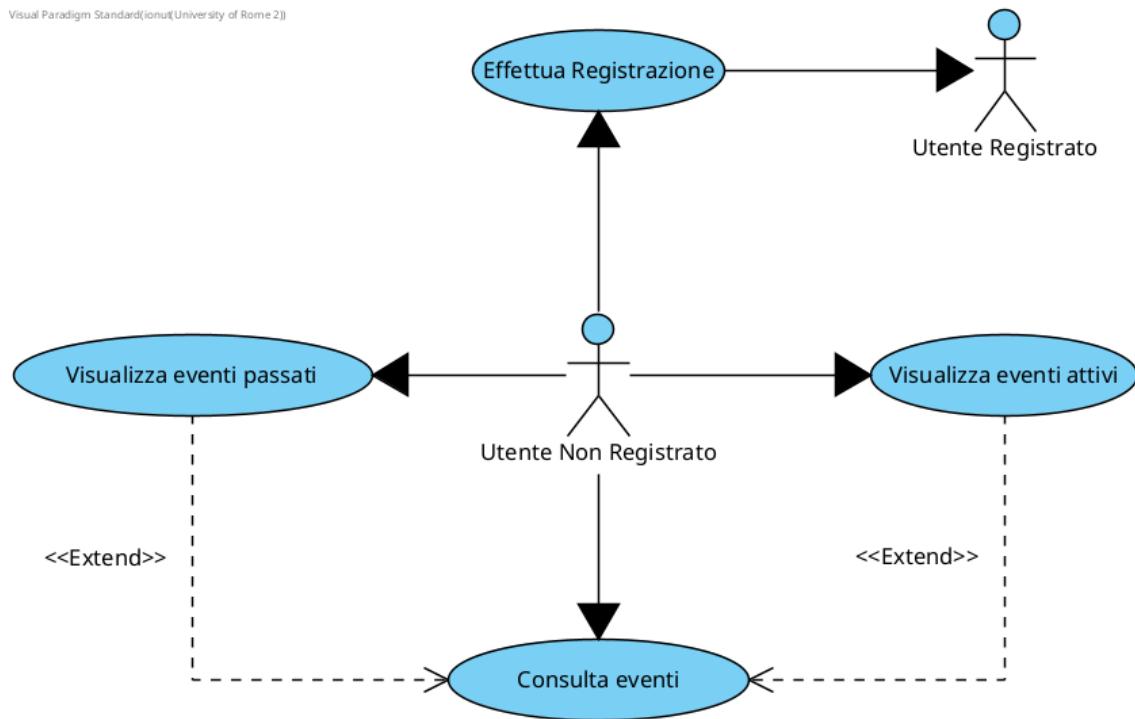


Figura 4: Use Case "Utente Non Registrato"

3.4.2 Documentazione

Use Case: Effettua registrazione

Use case	Effettua registrazione	
Descrizione	Un utente non registrato esegue il processo di registrazione.	
Attori	Utente non registrato	
Precondizioni	L'utente deve accettare il trattamento dei dati.	
Scenario principale	Passo	Azione
	1	L'utente sceglie di registrarsi.
	2	Il sistema mostra il form di registrazione.
	3	L'utente inserisce i dati personali richiesti.
	4	Il sistema verifica i dati e li memorizza.
	5	L'utente diventa registrato e ottiene i privilegi.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	L'utente inserisce dati non validi. Il sistema mostra un errore e richiede la correzione.
Scenario 2	1	Il sistema rileva che l'email è già registrata e avvisa l'utente.
Post-condizioni	L'utente è ora registrato nel sistema.	

Use Case: Consulta eventi

Use case	Consulta eventi	
Descrizione	L'utente visualizza gli eventi disponibili.	
Attori	Utente non registrato	
Precondizioni	Il sistema deve contenere eventi programmati.	
Scenario principale	Passo	Azione
	1	L'utente accede alla funzione di consultazione eventi.
	2	Il sistema mostra il calendario degli eventi.
	3	L'utente può selezionare un evento passato o in corso per visualizzarne i dettagli.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Non ci sono eventi nel calendario. Il sistema mostra un messaggio di avviso.
Scenario 2	1	L'utente seleziona un evento che è stato rimosso. Il sistema notifica che l'evento non è più disponibile.
Post-condizioni	L'utente ha visualizzato il calendario eventi.	

3.5 Use Case Squadra

3.5.1 Diagramma

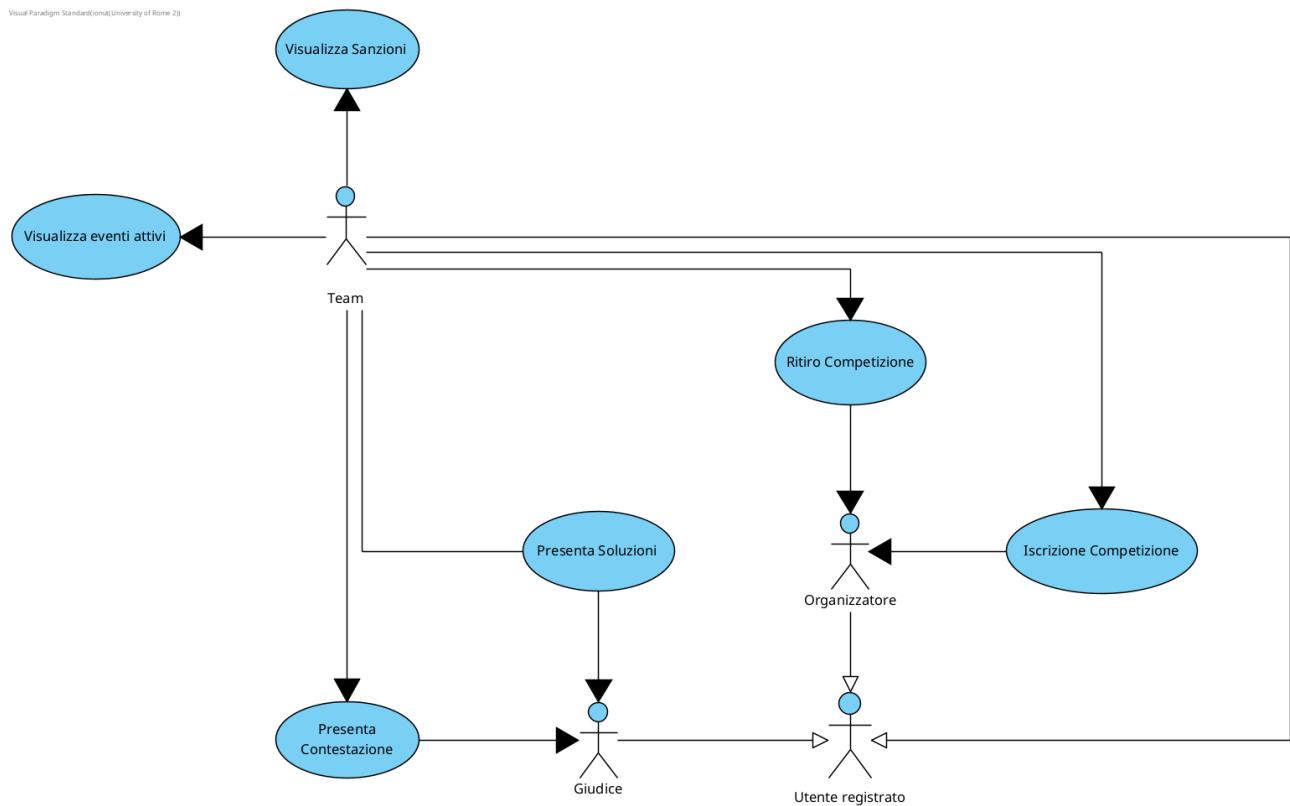


Figura 5: Use Case "Team"

3.5.2 Documentazione

Use Case: Visualizza classifica

Use case	Visualizza classifica	
Descrizione	Il team visualizza la classifica in tempo reale.	
Attori	Team	
Precondizioni	Il team è iscritto alla competizione.	
Scenario principale	Passo	Azione
	1	Il team accede alla funzione di visualizzazione delle classifiche.
	2	Il sistema mostra gli eventi disponibili.
	3	Il team può selezionare un'evento per visualizzarne i dettagli.
Scenari alternativi	Passo	Descrizione
//	//	//
Post-condizioni	Il team ha visualizzato le classifiche.	

Use Case: Iscrizione Competizione

Use case	Iscrizione Competizione	
Descrizione	Il team si iscrive alla competizione.	
Attori	Team, Organizzatore	
Precondizioni	Il team è registrato nel sistema.	
Scenario principale	Passo	Azione
	1	Il team richiede il form di iscrizione.
	2	Sceglie a quale competizione iscriversi.
	3	Compila e invia richiesta d'iscrizione.
Scenari alternativi	Passo	Descrizione
Scenario 1	2	La competizione selezionata non è più disponibile (esaurimento posti o scadenza): il sistema mostra un messaggio di errore.
Post-condizioni	La richiesta viene inviata all'organizzatore.	

Use Case: Ritiro Competizione

Use case	Ritiro Competizione	
Descrizione	Il team comunica il ritiro dalla competizione.	
Attori	Team, Organizzatore	
Precondizioni	Il team è attivamente iscritto alla competizione.	
Scenario principale	Passo	Azione
	1	Il team accede al portale.
	2	Richiede il modulo di ritiro, lo compila e lo invia.
Scenari alternativi	Passo	Descrizione
Scenario 1	2	Il ritiro non è consentito perché si è superata la data limite prevista dal regolamento: il sistema blocca l'azione e notifica il team.
Post-condizioni	Il team non compare più tra i partecipanti.	

Use Case: Visualizza Sanzioni

Use case	Visualizza Sanzioni	
Descrizione	Il team visualizza le sanzioni ricevute.	
Attori	Team	
Precondizioni	Il team ha ricevuto sanzioni precedentemente.	
Scenario principale	Passo	Azione
	1	Il team accede al portale.
	2	Accede alla sezione "Visualizza Sanzioni".
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Nessuna sanzione da visualizzare
Post-condizioni	Le sanzioni sono visualizzate.	

Use Case: Presenta Soluzioni

Use case	Presenta Soluzioni	
Descrizione	La squadra presenta le soluzioni dei problemi al Giudice.	
Attori	Team, Giudice	
Precondizioni	La squadra deve essere iscritta alla competizione.	
Scenario principale	Passo	Azione
	1	Uno dei membri della squadra accede alla pagina dedicata.
	2	Le soluzioni vengono caricate sulla piattaforma
	3	Il sistema ha registrato le soluzioni della squadra.
Scenari alternativi	Passo	Descrizione
Scenario 1	1	Le soluzioni non vengono caricate correttamente e il sistema restituisce un messaggio di errore.
Post-condizioni	Le soluzioni della squadra sono state inviate al giudice.	

Use Case: Presenta Contestazione

Use case	Presenta Contestazione	
Descrizione	La squadra non è d'accordo con il giudizio/sanzione del giudice, e decide di fare ricorso.	
Attori	Team, Giudice	
Precondizioni	La squadra ha ricevuto un giudizio o una sanzione.	
Scenario principale	Passo	Azione
	1	Dopo aver analizzato il contenuto del giudizio, la squadra valuta di non essere d'accordo con la decisione.
	2	Il team compila un modulo di ricorso, indicando chiaramente le motivazioni e le eventuali prove a sostegno della propria posizione.
	3	Il ricorso viene inviato formalmente al giudice attraverso il sistema.
Scenari alternativi	Passo	Descrizione
Scenario 1	2	Il modulo di ricorso risulta incompleto o contiene errori formali: il sistema lo rifiuta e chiede alla squadra di correggerlo e ripresentarlo.
Scenario 2	2	Il termine per la presentazione del ricorso è scaduto: il sistema blocca l'invio e notifica alla squadra che non è più possibile fare ricorso.
Post-condizioni	Il giudice riceve il ricorso e ne conferma l'avvenuta ricezione.	

4 System Requirements

4.1 Requisiti Funzionali

Attore	ID	Requisiti funzionali	Descrizione
Giudice	1.1	Valuta soluzione del team	Esamina le soluzioni proposte e assegna punteggi in base alla qualità, originalità e coerenza.
	1.2	Assegna penalità	Applica penalità in caso di violazioni delle regole definite per l'evento.
	1.3	Annulla o riprogramma la competizione	Consente di annullare o rinviare la competizione per cause tecniche o organizzative.
	1.4	Riesame delle contestazioni	Riesamina eventuali reclami per assicurare equità e trasparenza.
	1.5	Conferma incarico	Conferma ufficialmente la sua partecipazione come giudice all'evento.
	1.6	Pubblica i risultati	Rende pubblici i risultati finali dopo le valutazioni.
Organizzatore	2.1	Nomina e crea credenziali giudice	Seleziona e assegna credenziali di accesso ai giudici.
	2.2	Ricerca sito e struttura evento	Trova location e definisce la struttura organizzativa dell'hackathon.
	2.3	Programma evento	Definisce il calendario dell'evento, scadenze e attività principali.
	2.4	Sponsorship e premi	Firma accordi con sponsor e definisce premi per i vincitori.
Utente Registrato	3.1	Effettua accesso	Accede alla piattaforma utilizzando credenziali valide.
	3.3	Consulta eventi	Visualizza il calendario degli eventi.
	3.5	Consulta classifica aggiornata	Visualizza la classifica aggiornata dei team.
	3.6	Crea squadra	Crea una nuova squadra per partecipare all'hackathon.
	3.7	Unisce a squadra esistente	Richiede di entrare in una squadra già creata.
Utente Non Registrato	4.1	Effettua registrazione	Compila il modulo di iscrizione per creare un account.
	4.2	Consulta eventi	Visualizza liberamente il calendario delle competizioni.
	4.5	Visualizza classifica	Osserva la classifica generale pubblica dei team.
Team	5.1	Iscrizione alla competizione	Effettua l'iscrizione del team alla gara tramite la piattaforma.
	5.2	Visualizza classifica	Controlla la propria posizione nella classifica.
	5.3	Visualizza sanzioni	Consulta le eventuali penalità assegnate.
	5.4	Ritiro competizione	Permette al team di ritirarsi ufficialmente dalla competizione.
	5.5	Fornisce soluzione al giudice	Invia la soluzione tecnica da sottoporre alla valutazione del giudice.
	5.6	Presenta ricorso al giudice	Inoltra reclami formali contro decisioni ritenute non corrette.

4.2 Requisiti Non Funzionali

Requisito	Descrizione
Performance	Le operazioni principali (come l'iscrizione, il caricamento degli eventi e i punteggi in tempo reale) devono essere eseguite in maniera rapida. Il sistema deve essere in grado di supportare fino a 500 utenti per ora e fornire le pagine richieste in meno di 2 secondi, includendo la formattazione del testo e la visualizzazione delle immagini, per garantire un'esperienza fluida durante l'evento.
Scalabilità	In prospettiva, l'organizzazione degli hackathon potrebbe richiedere l'aggiunta di nuove funzionalità (ad esempio, integrazioni con API di social media o sistemi di ranking in tempo reale), un incremento del traffico e la gestione di un volume maggiore di dati, mantenendo invariata la qualità delle prestazioni. L'obiettivo è facilitare questi ampliamenti senza stravolgere il codice già esistente.
Portabilità	Il sistema deve essere facilmente accessibile sia tramite sito web che tramite app mobile. Quest'ultima dovrà essere compatibile con Android e iOS (le versioni correnti e le ultime tre versioni), mentre il sito garantirà la compatibilità con i principali browser (Chrome, Firefox, Opera, Safari, Edge) per assicurare l'accesso a tutti i partecipanti, indipendentemente dal dispositivo utilizzato.
Affidabilità	Il sistema deve operare senza guasti nel 98% dei casi, assicurando il regolare svolgimento dell'evento e la gestione stabile dei processi di partecipazione e valutazione. È garantita la manutenzione per i successivi 5 anni a partire dal rilascio.
Manutenibilità	La struttura modulare del sistema deve consentire una manutenzione semplice, agevolando l'aggiunta di nuove funzionalità o l'automatizzazione di procedure correlate in futuro. In caso di guasti, il ripristino del sistema deve avvenire entro poche ore, minimizzando l'impatto sull'evento.
Disponibilità	Le funzionalità del sistema devono essere operative per il 90% del tempo durante le fasi dell'hackathon, garantendo un accesso continuo e affidabile a tutti i partecipanti.
Usabilità	Dato che non tutti gli utenti avranno competenze informatiche avanzate, l'interfaccia deve essere intuitiva e semplice da utilizzare senza necessità di formazione approfondita. Il design, inoltre, dovrà essere responsive, adattandosi automaticamente a diverse dimensioni e tipologie di dispositivi, e prevedere un supporto tecnico dedicato. L'obiettivo è mantenere il tasso di errore degli utenti sotto il 10% durante i test.
Sicurezza	La gestione dei dati sensibili (come email e numeri di telefono) e delle operazioni critiche (quali l'invio dei progetti e il voto dei giudici) deve essere strettamente protetta. Ogni account dovrà essere autenticato tramite username e password, e le funzionalità critiche saranno accessibili esclusivamente a personale autorizzato, al fine di prevenire accessi non autorizzati e garantire l'integrità delle competizioni.

4.3 Requisiti di Dominio

In questa sezione vengono elencati i principali requisiti normativi e regolamentari che devono essere considerati nell'ambito dello sviluppo del progetto. Tali requisiti sono fondamentali per garantire la conformità legale, organizzativa e procedurale dell'iniziativa.

1. Conformità al Regolamento Generale sulla Protezione dei Dati (GDPR).

È richiesto il rispetto del Regolamento (UE) 2016/679, che disciplina il trattamento dei dati personali delle persone fisiche e ne garantisce la libera circolazione all'interno dell'Unione Europea. Il progetto dovrà pertanto adottare tutte le misure necessarie per assicurare la protezione dei dati, in conformità agli obblighi previsti dalla normativa vigente.

Link di riferimento: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>

2. Esempio di Regolamento di Campionato

A supporto dell'organizzazione di eventi competitivi, viene fornito un esempio pratico di regolamento di campionato. Questo documento può essere utilizzato come modello per definire le regole di partecipazione, i criteri di valutazione e le modalità di gestione delle competizioni.

Link di riferimento: <https://www.example.com/regolamento-campionato>

3. Esempio di Circolare Normativa

Per la definizione di procedure organizzative interne e per la gestione efficace dell'Hackathon, viene messo a disposizione un esempio di circolare normativa. Questo documento rappresenta un utile riferimento per l'impostazione di criteri, modalità operative e linee guida procedurali.

Link di riferimento: <https://www.example.com/circolare-normativa>

5 System Architectural Models

5.1 Activity Diagrams

5.1.1 Activity Diagrams Giudice

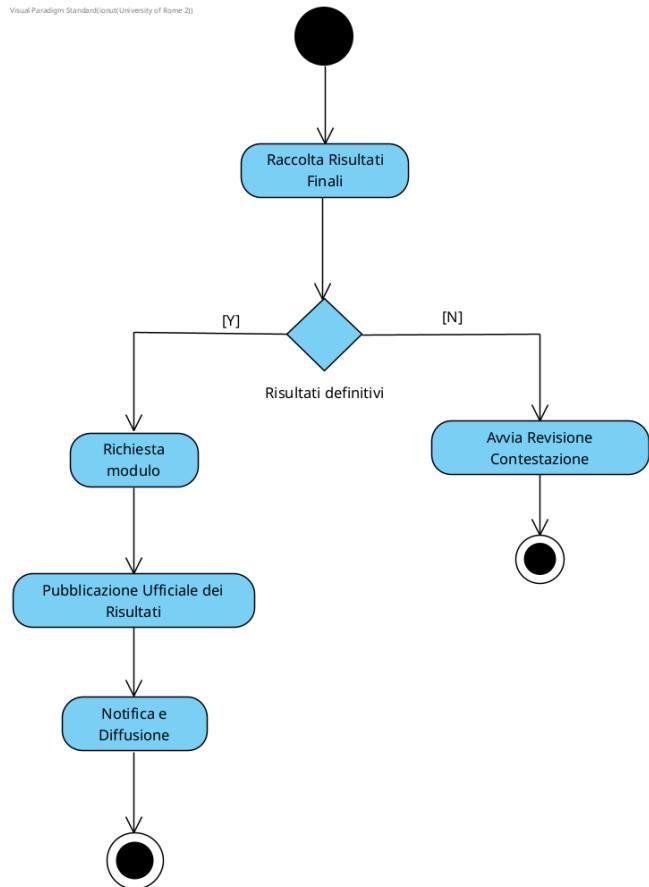


Figura 6: Activity Diagram "Pubblicazione Risultati"

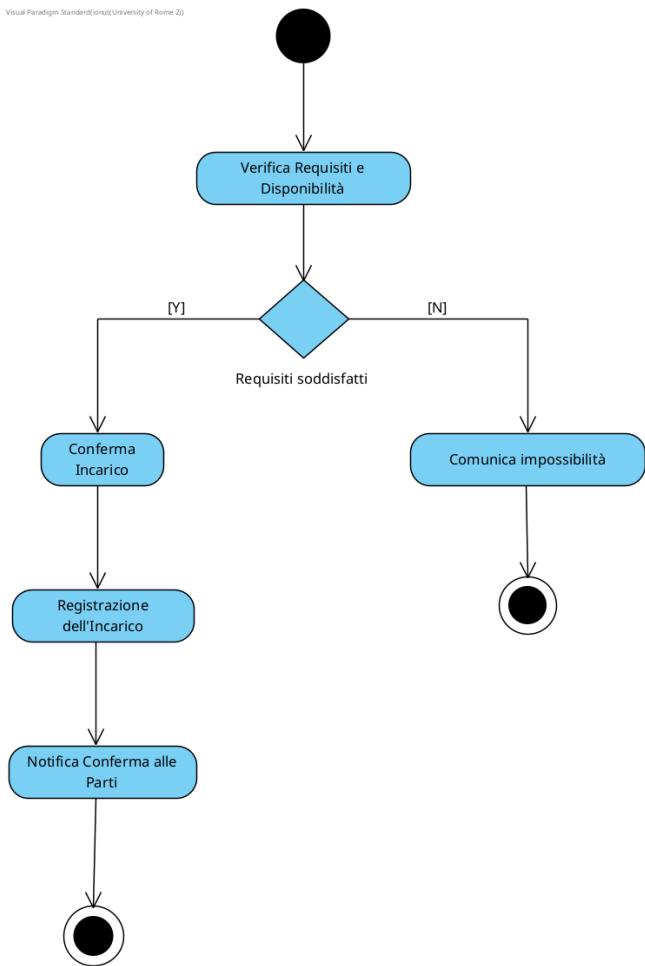


Figura 7: Activity Diagram "Conferma Incarico"

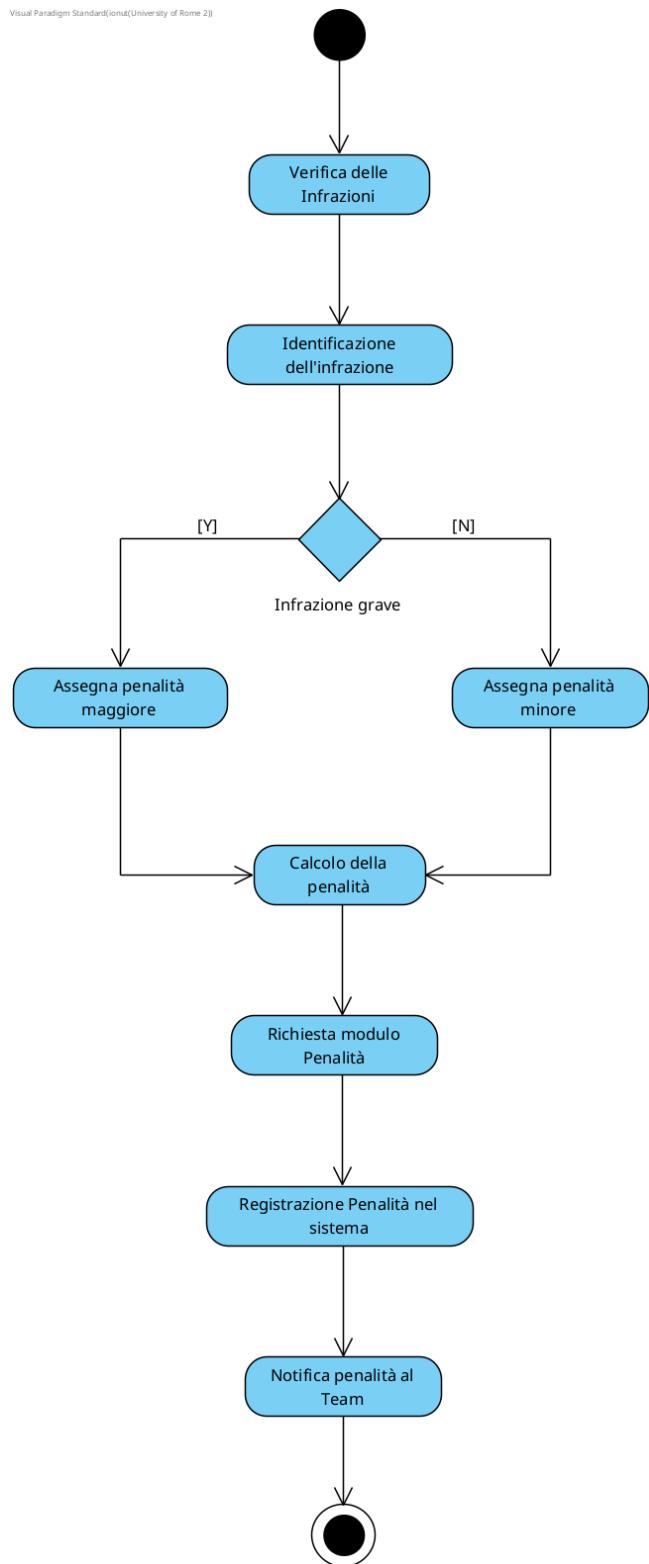


Figura 8: Activity Diagram "Assegnazione Penalità"

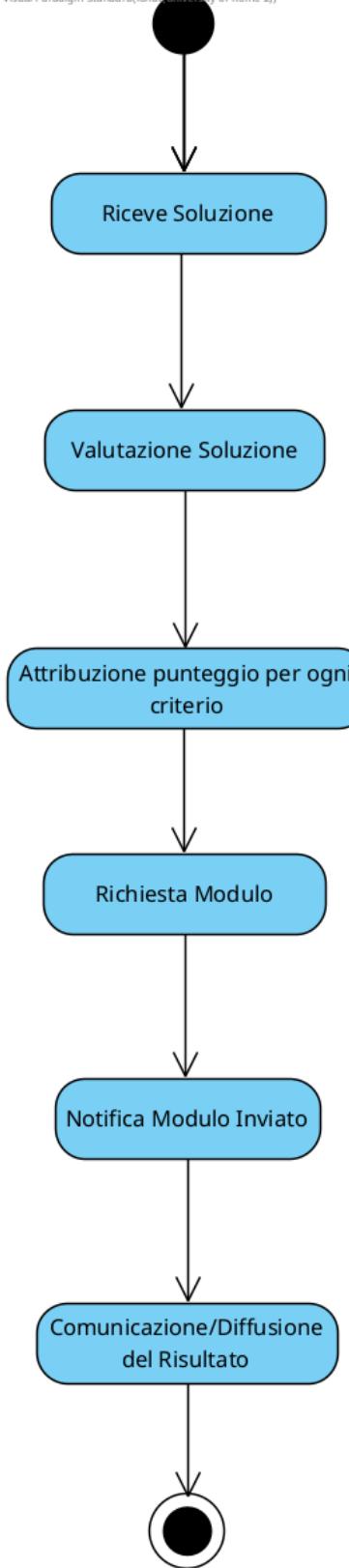


Figura 9: Activity Diagram "Valutazione Soluzione"

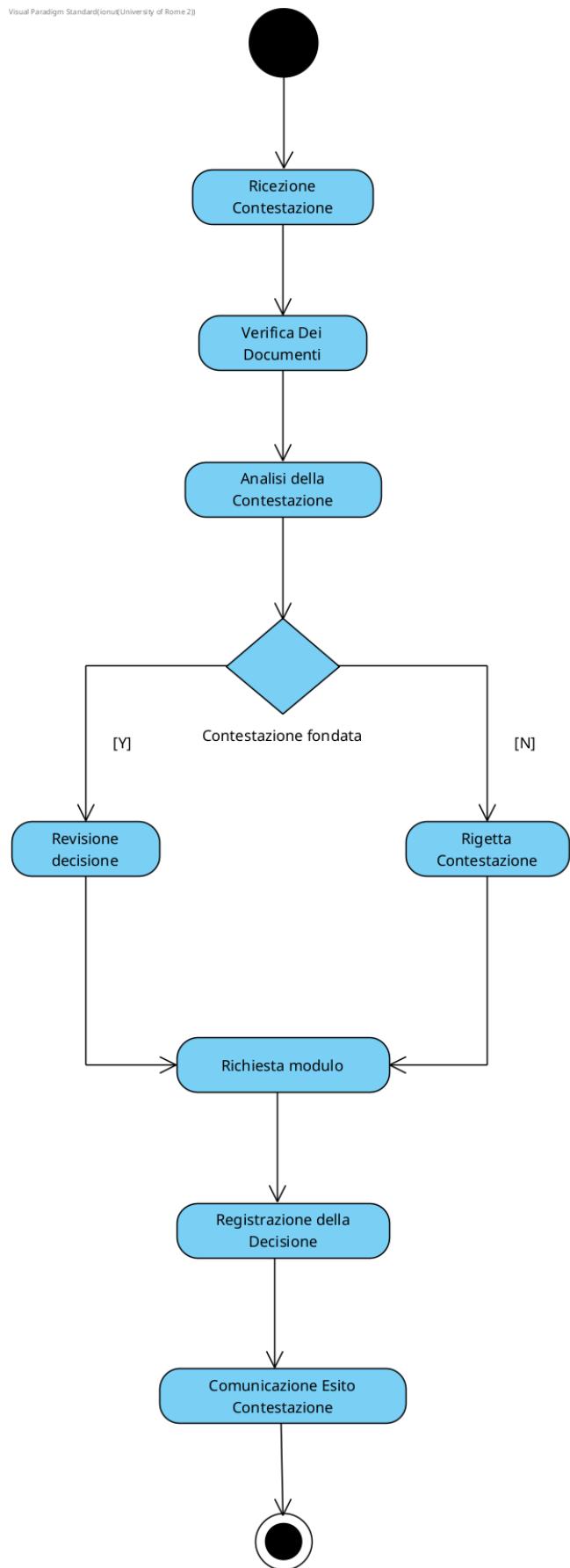


Figura 10: Activity Diagram "Riesame Contestazioni"

5.1.2 Activity Diagrams Organizzatore

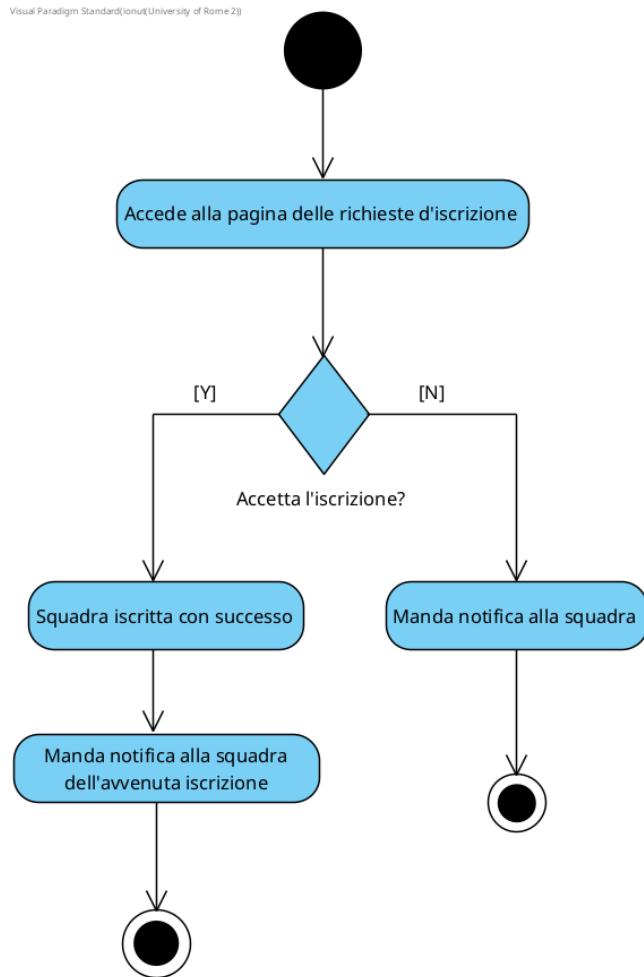


Figura 11: Activity Diagram "Accetta Iscrizione"

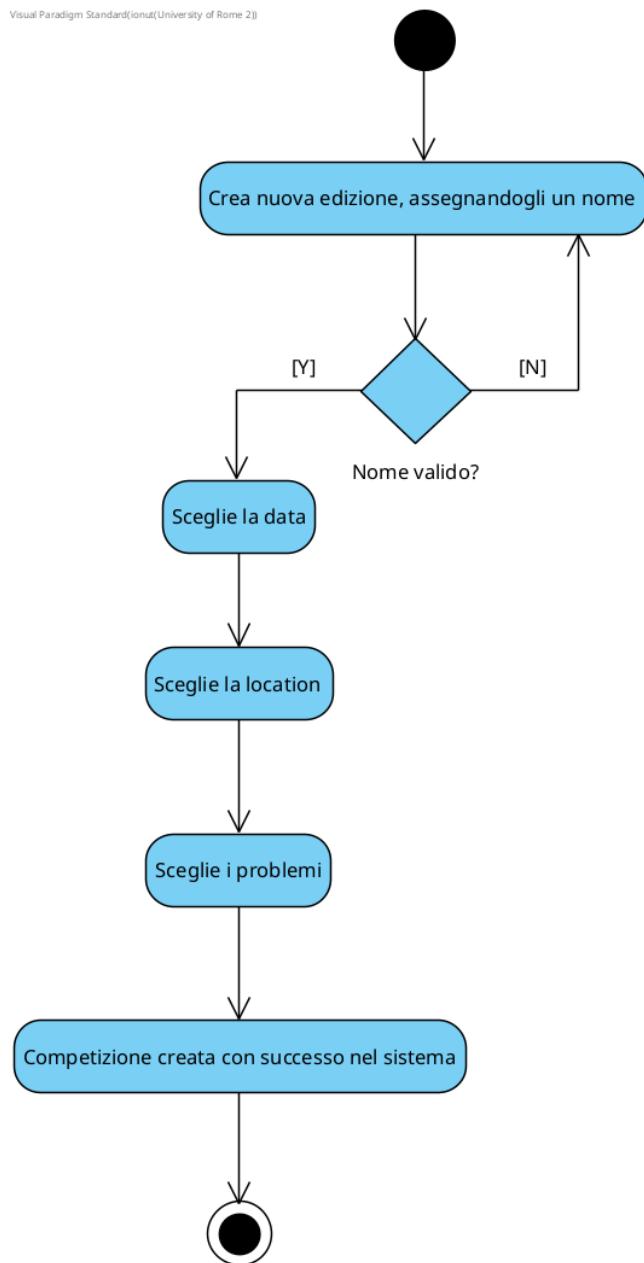


Figura 12: Activity Diagram "Crea Competizione"

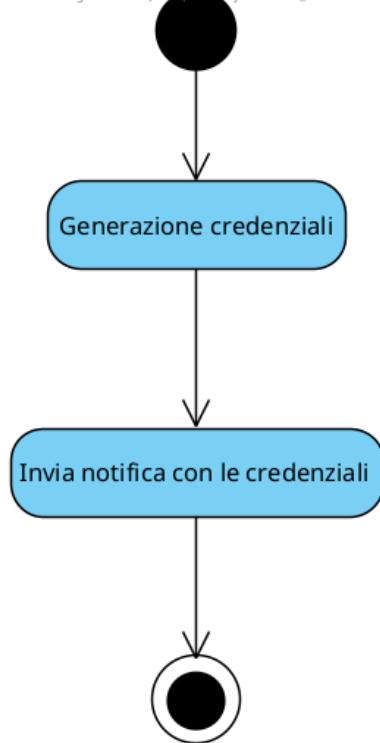


Figura 13: Activity Diagram "Crea Utente Giudice"

5.1.3 Activity Diagrams Utente Registrato

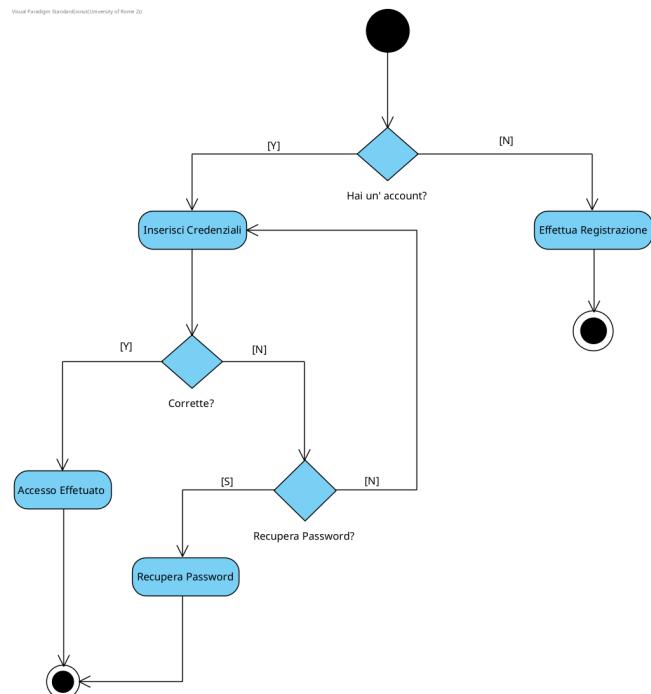


Figura 14: Activity Diagram "Login"

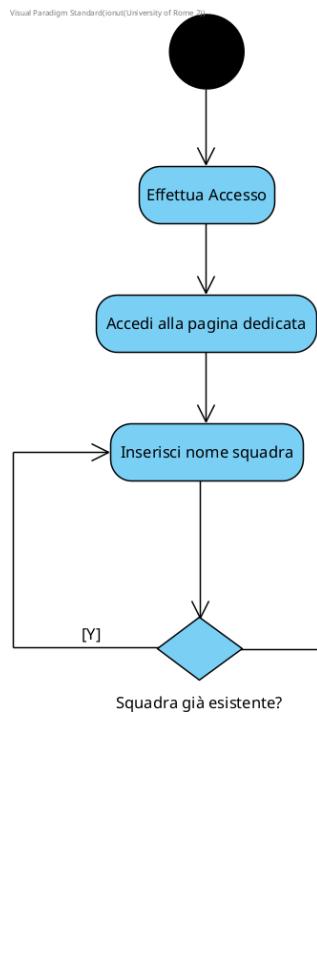


Figura 15: Activity Diagram "Crea Squadra"

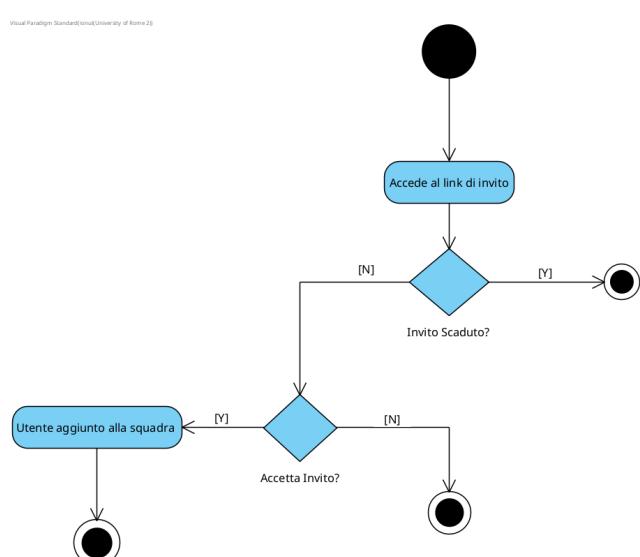


Figura 16: Activity Diagram "Unisce Squadra"

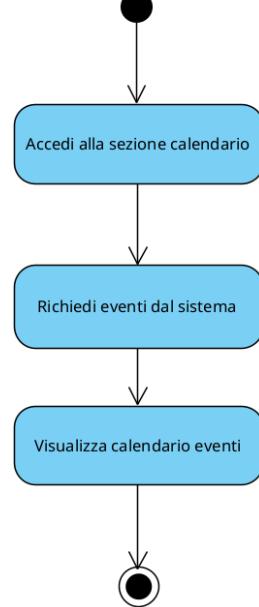


Figura 17: Activity Diagram "Consulta Eventi"

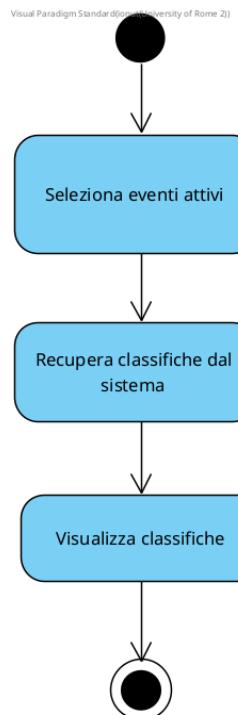


Figura 18: Activity Diagram "Visualizza Eventi Attivi"

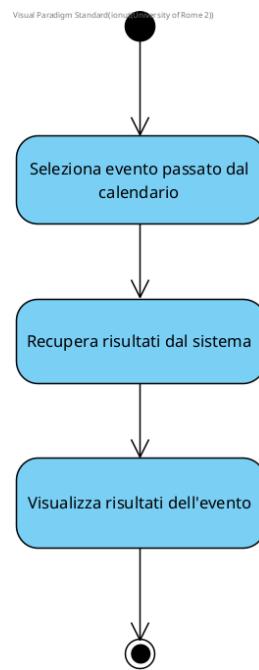


Figura 19: Activity Diagram "Visualizza Eventi Passati"

5.1.4 Activity Diagrams Utente Non Registrato

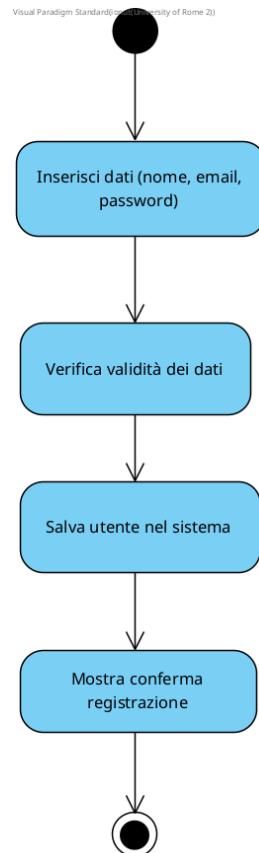


Figura 20: Activity Diagram "Resistrazione Utente"

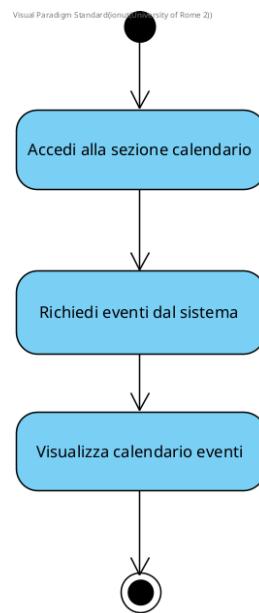


Figura 21: Activity Diagram "Consulta Eventi"

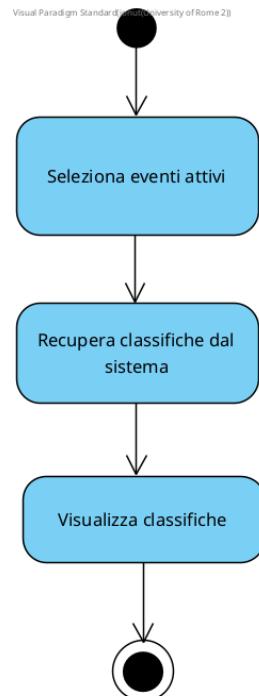


Figura 22: Activity Diagram "Visualizza Eventi Attivi"



Figura 23: Activity Diagram "Visualizza Eventi Passati"

5.1.5 Activity Diagrams Team

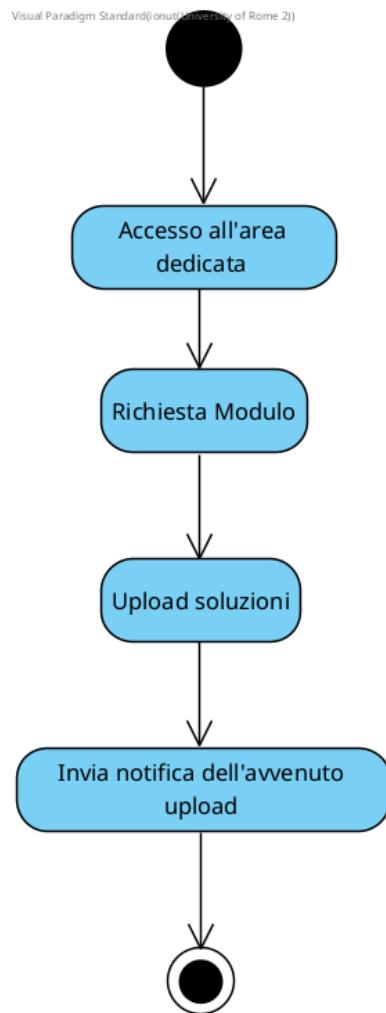


Figura 24: Activity Diagram "Fornisce Soluzioni"

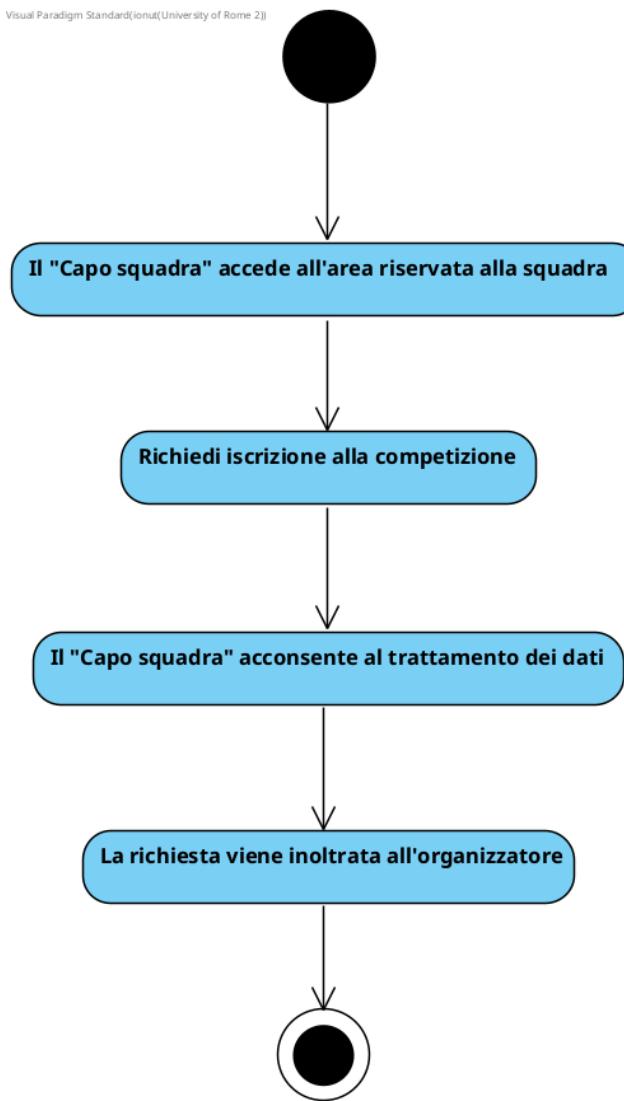


Figura 25: Activity Diagram "Iscrizione alla Competizione"

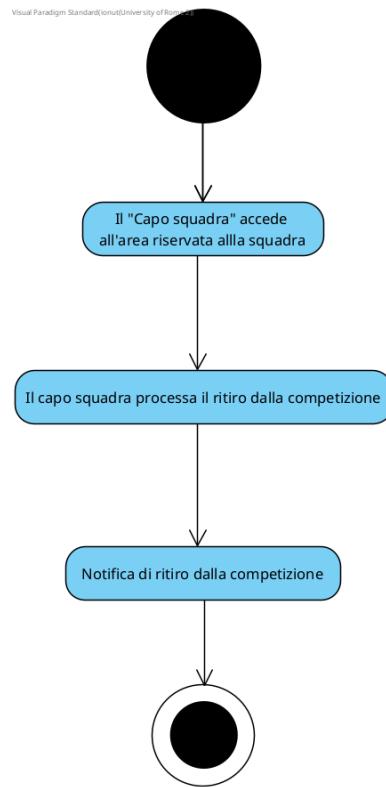


Figura 26: Activity Diagram "Ritiro dalla Competizione"

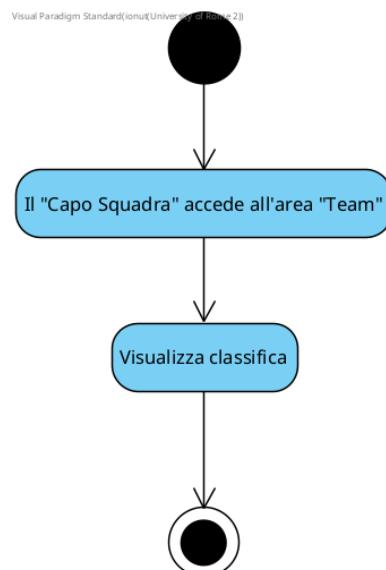


Figura 27: Activity Diagram "Visualizza Classifica"

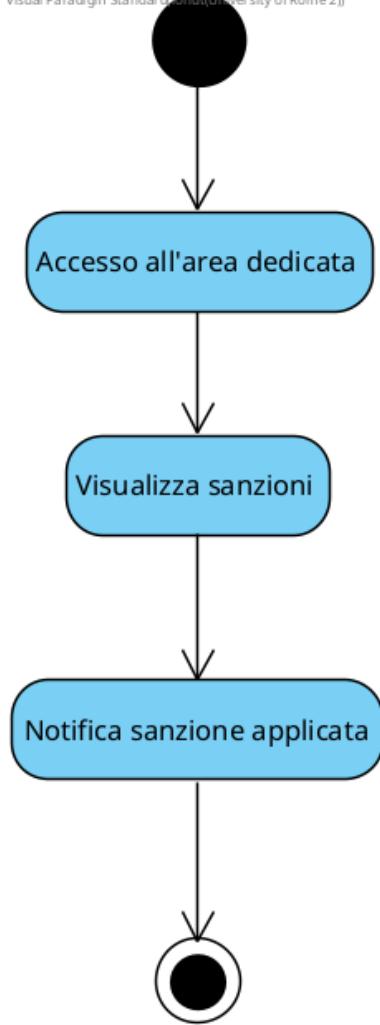


Figura 28: Activity Diagram "Visualizza Sanzioni"



Figura 29: Activity Diagram "Presenta Ricorso"

5.2 Sequence Diagrams

5.2.1 Sequence Diagrams Giudice

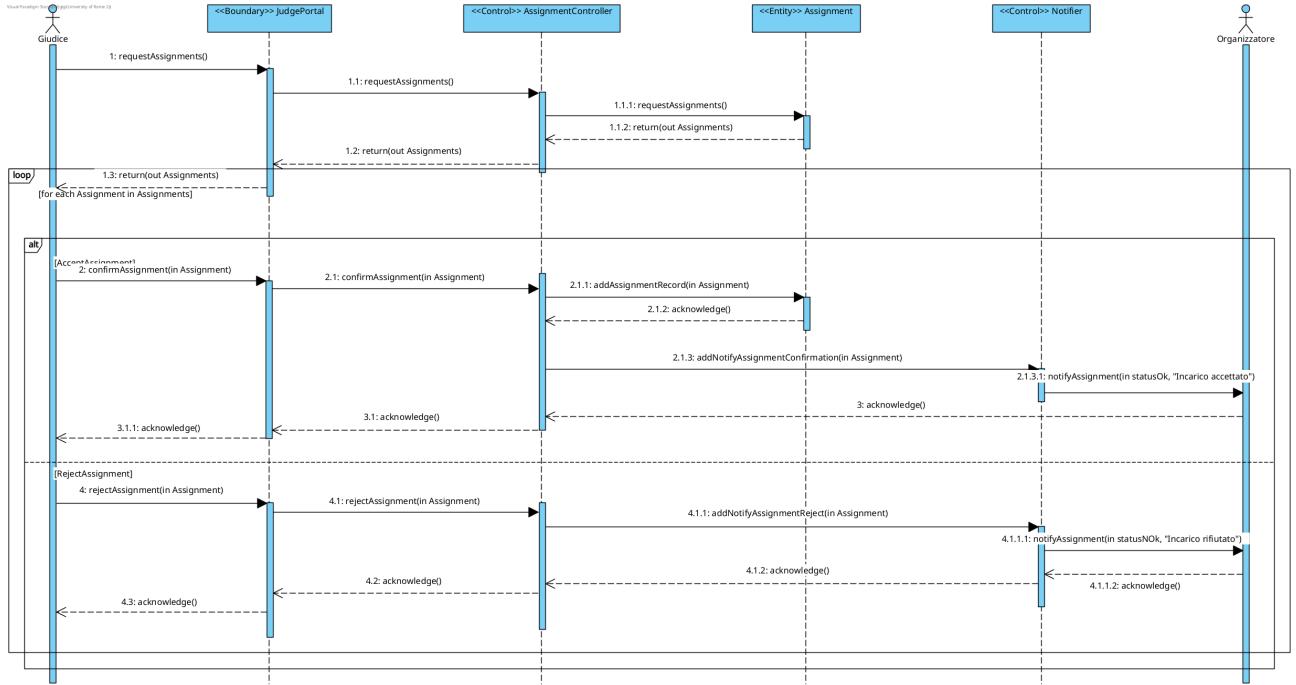


Figura 30: Sequence Diagram "Conferma Incarico"

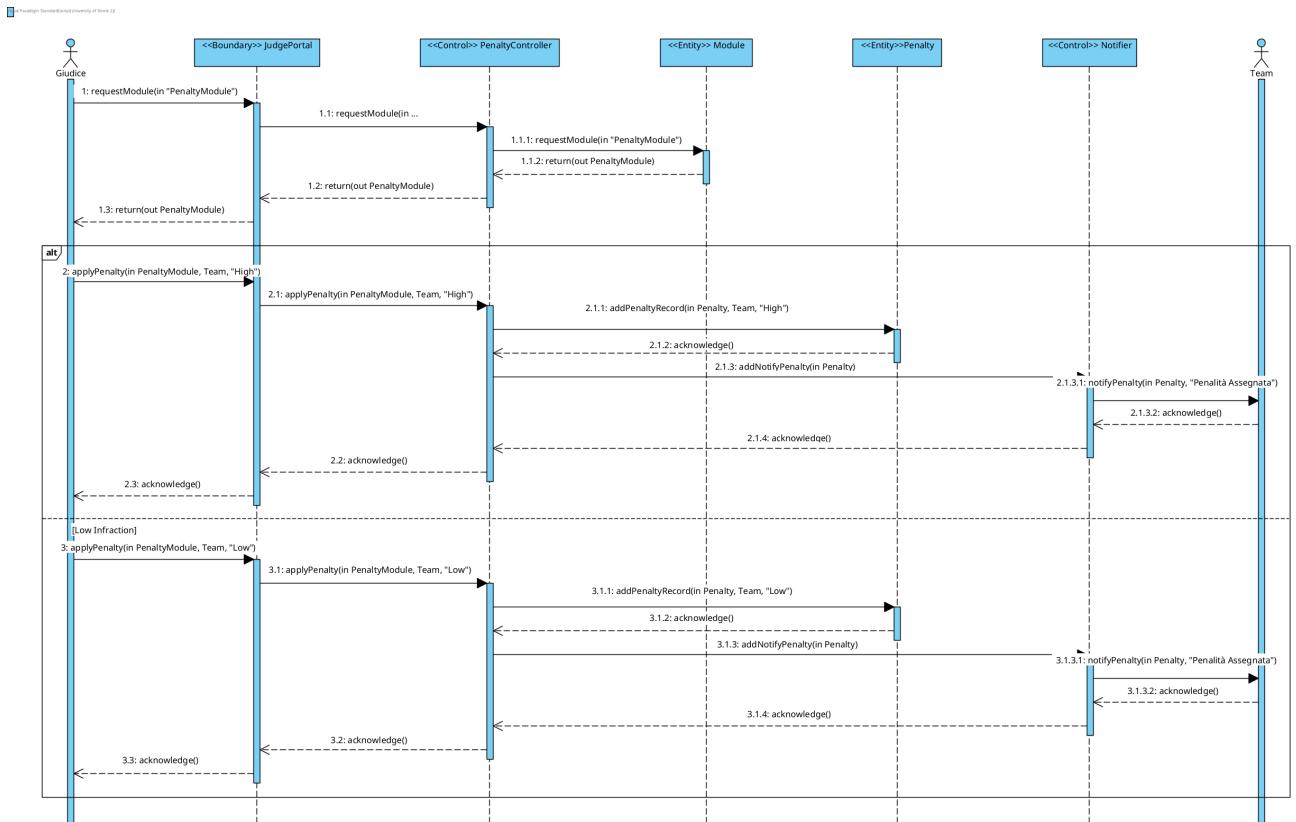


Figura 31: Sequence Diagram "Assegnazione Penalità"

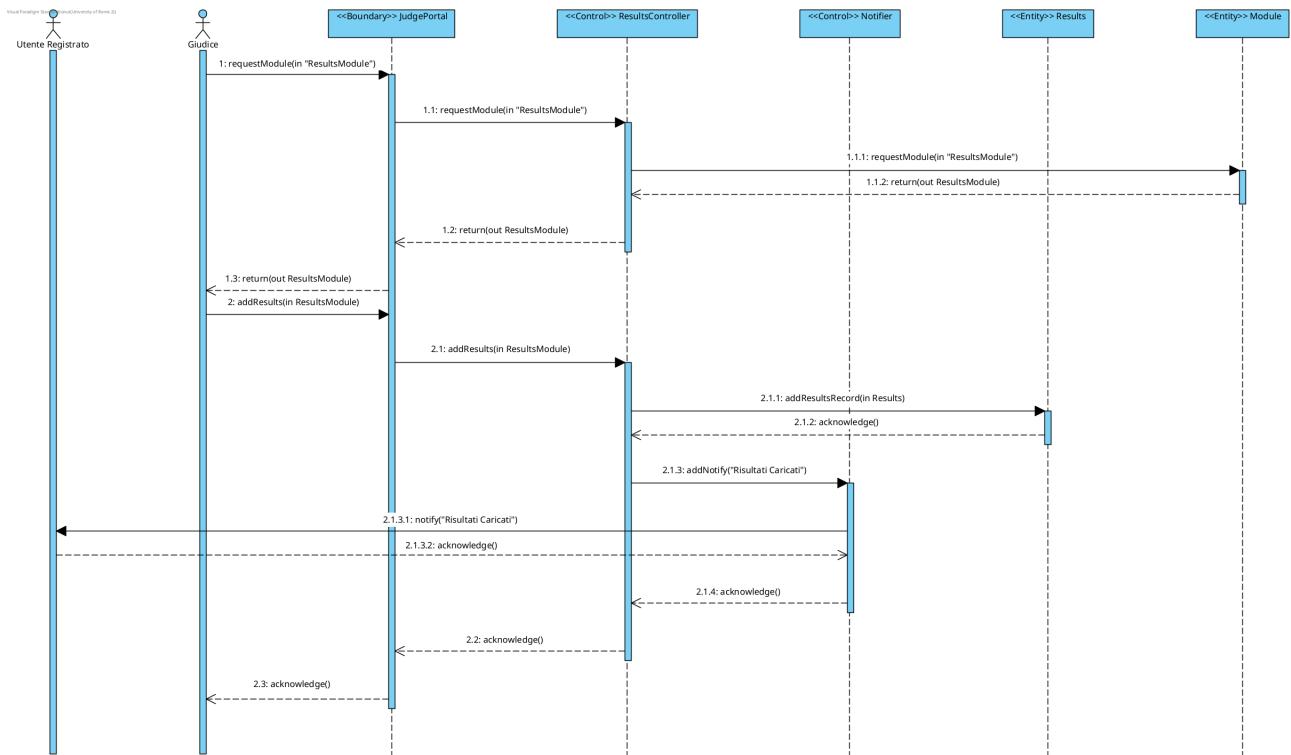


Figura 32: Sequence Diagram "Pubblicazione Risultati"

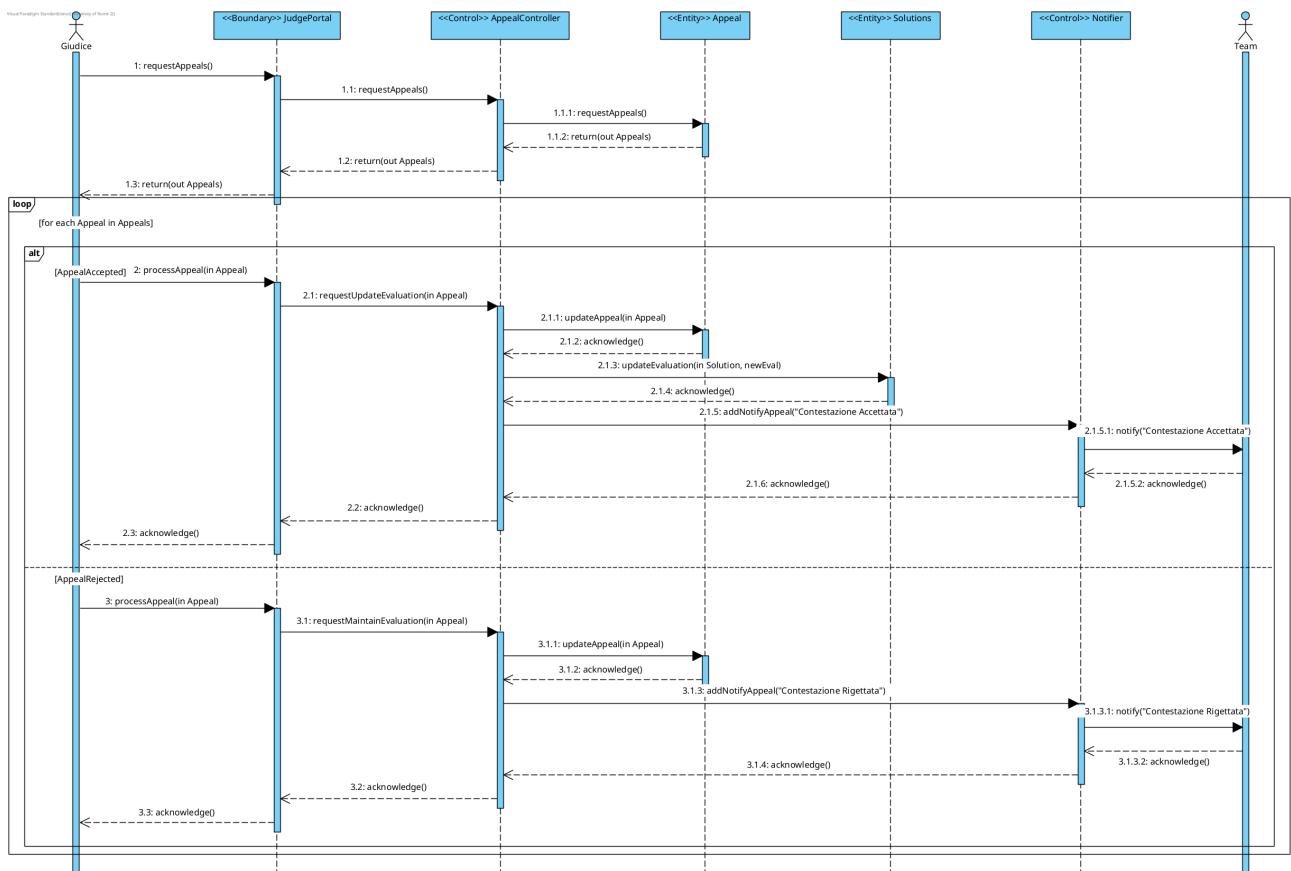


Figura 33: Sequence Diagram "Riesame Contestazione"

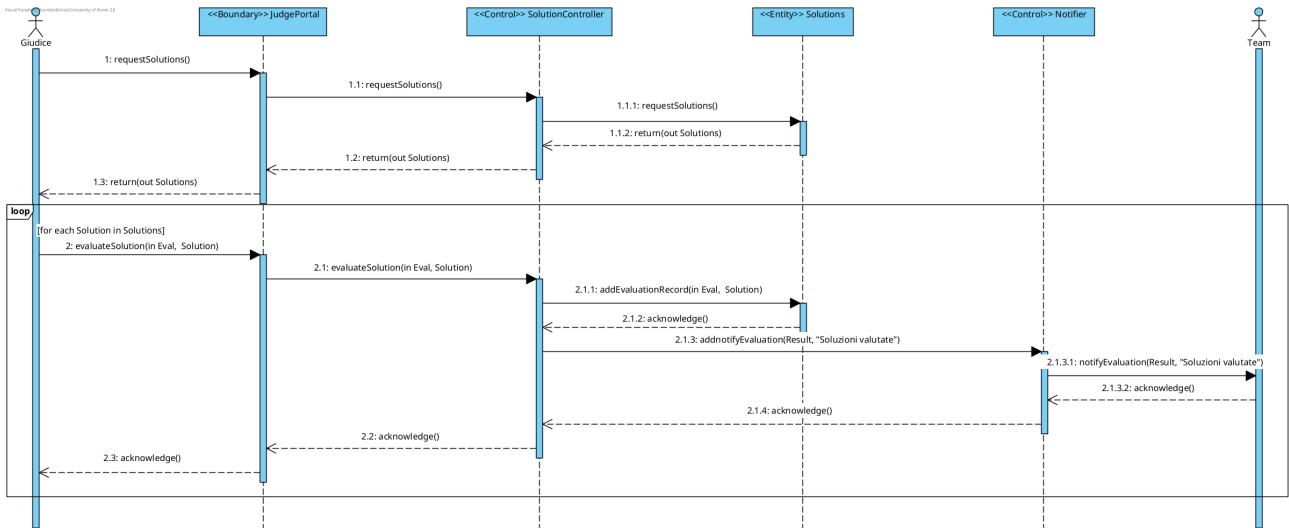


Figura 34: Sequence Diagram "Valutazione Soluzione"

5.2.2 Sequence Diagrams Organizzatore

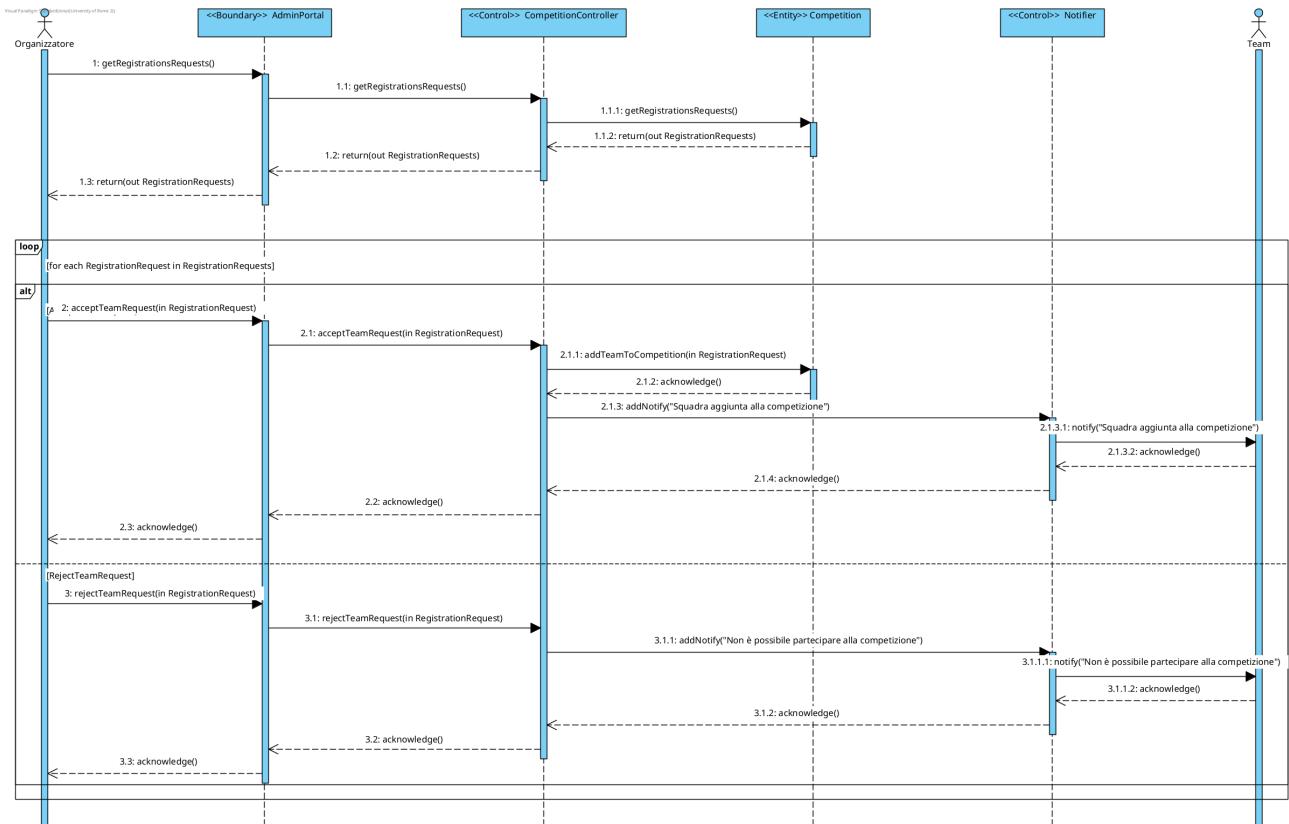


Figura 35: Sequence Diagram "Accetta Iscrizione"

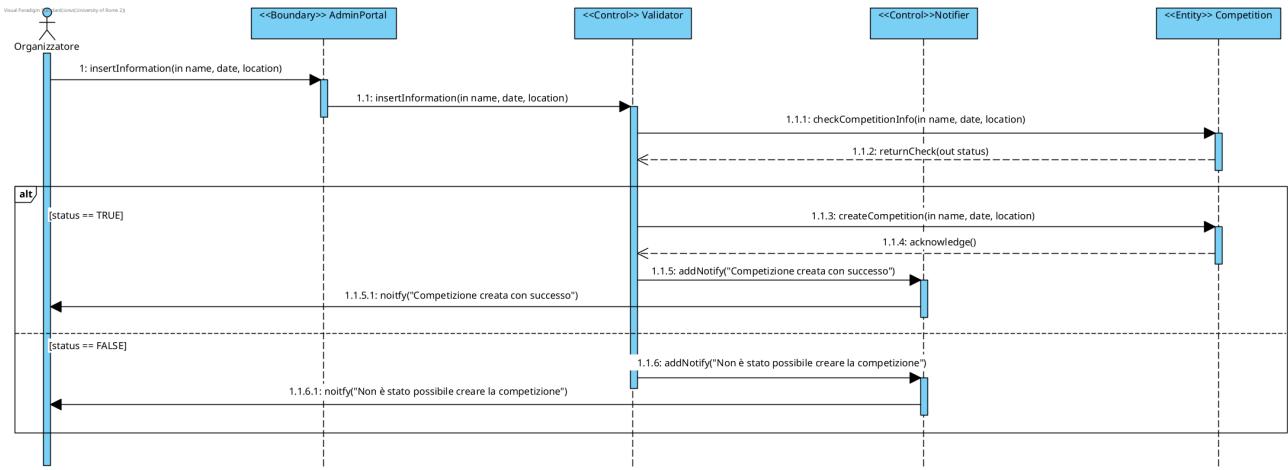


Figura 36: Sequence Diagram "Crea Evento"

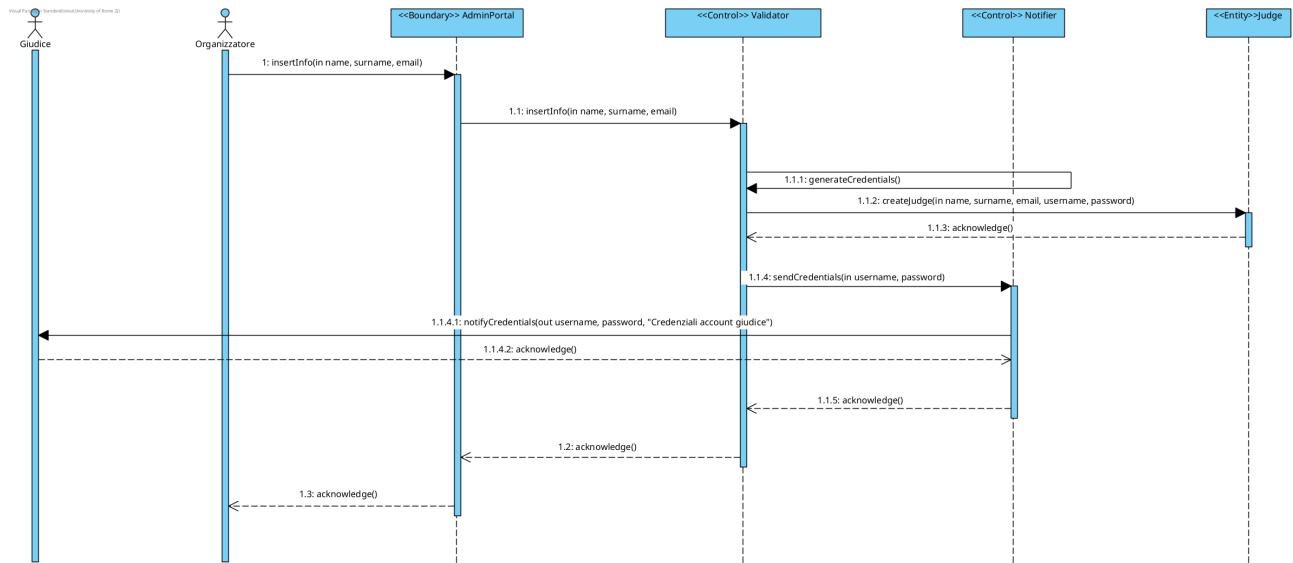


Figura 37: Sequence Diagram "Crea Utente Speciale: Giudice"

5.2.3 Sequence Diagrams Utente Registrato

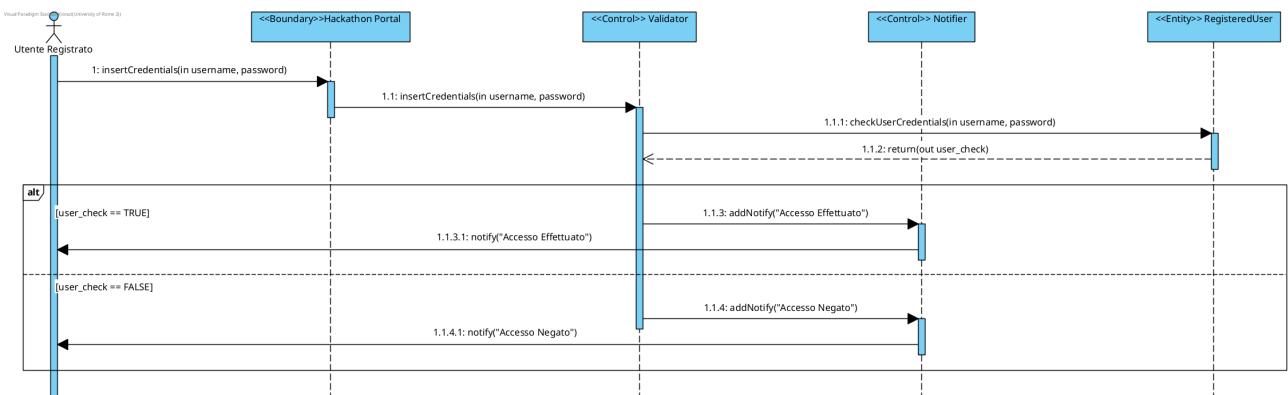


Figura 38: Sequence Diagram "Effettua Accesso"

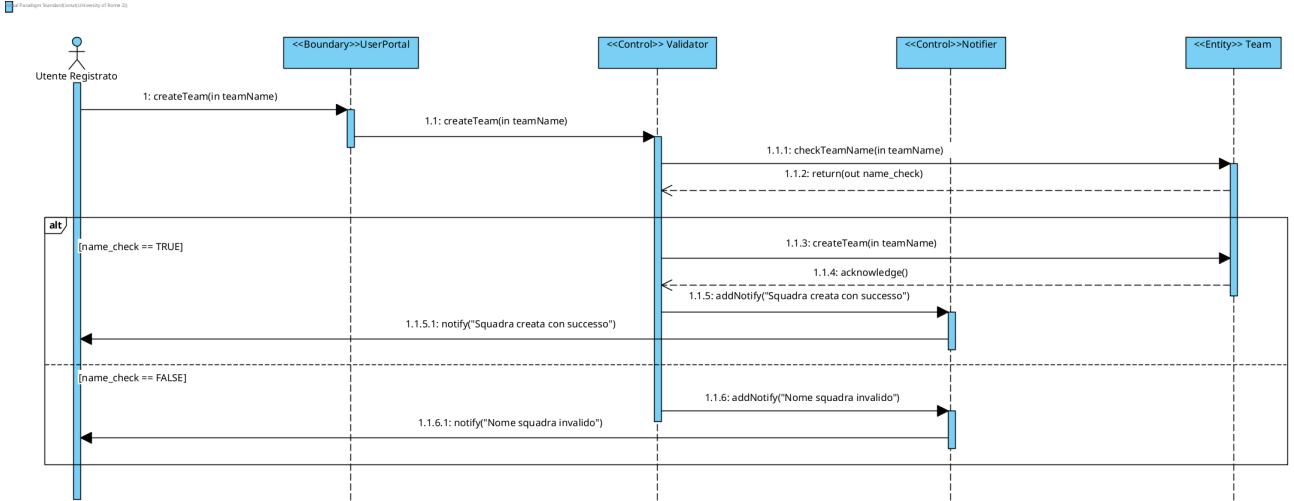


Figura 39: Sequence Diagram "Crea Squadra"

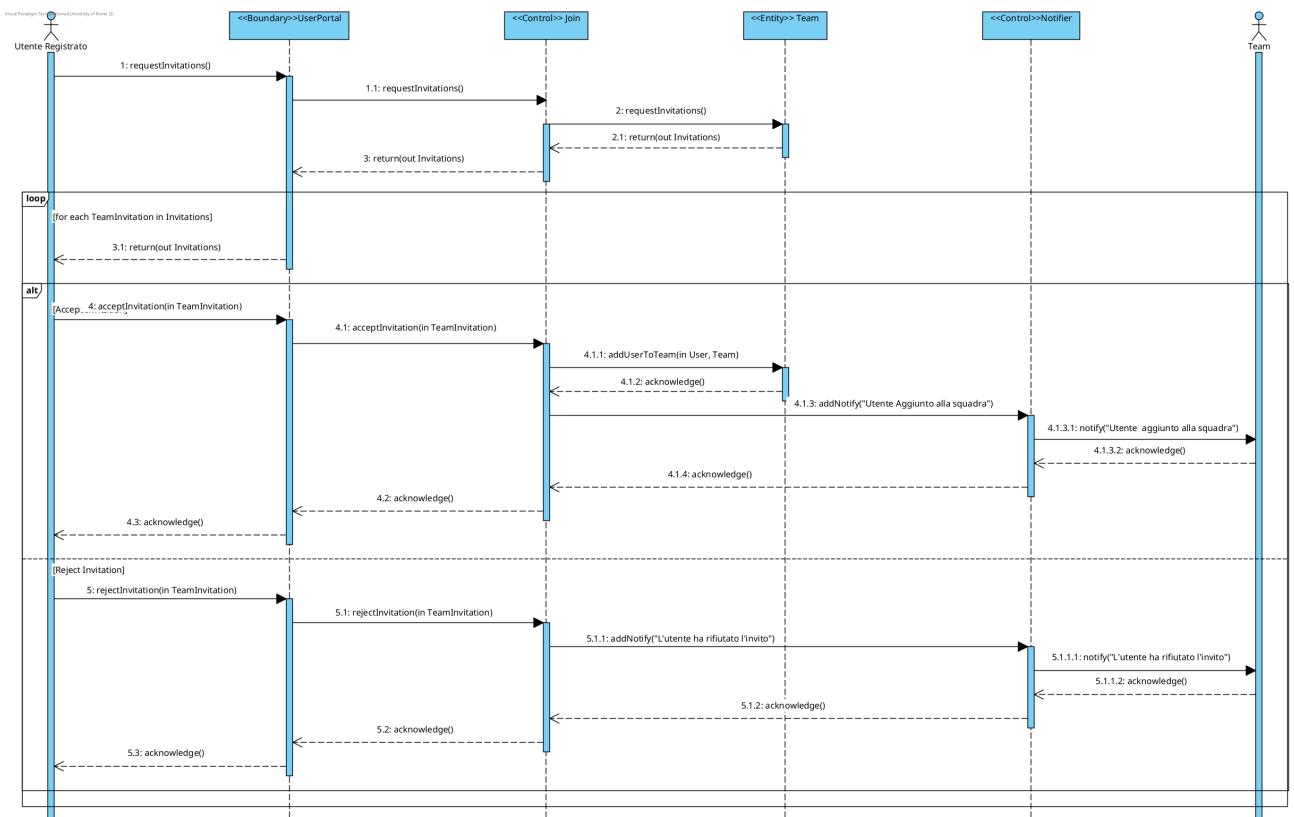


Figura 40: Sequence Diagram "Unisce a Squadra"

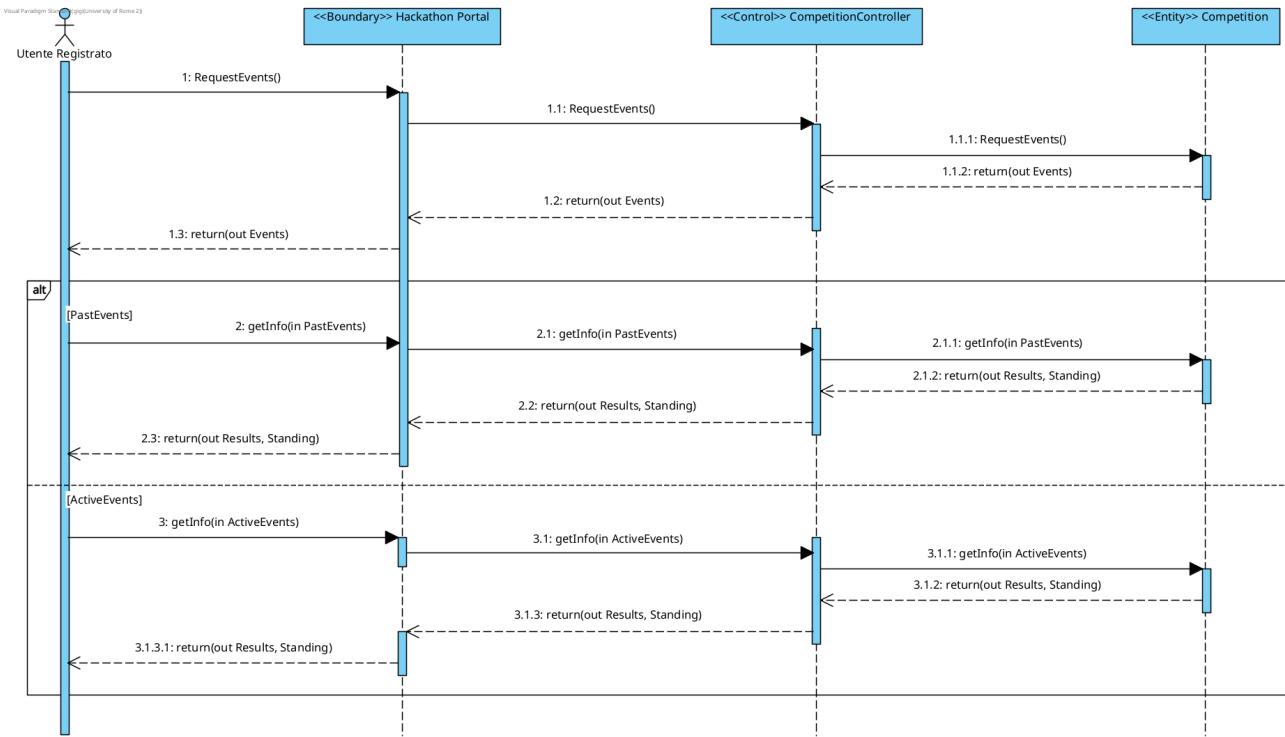


Figura 41: Sequence Diagram "Consulta Eventi"

5.2.4 Sequence Diagrams Utente Non Registrato

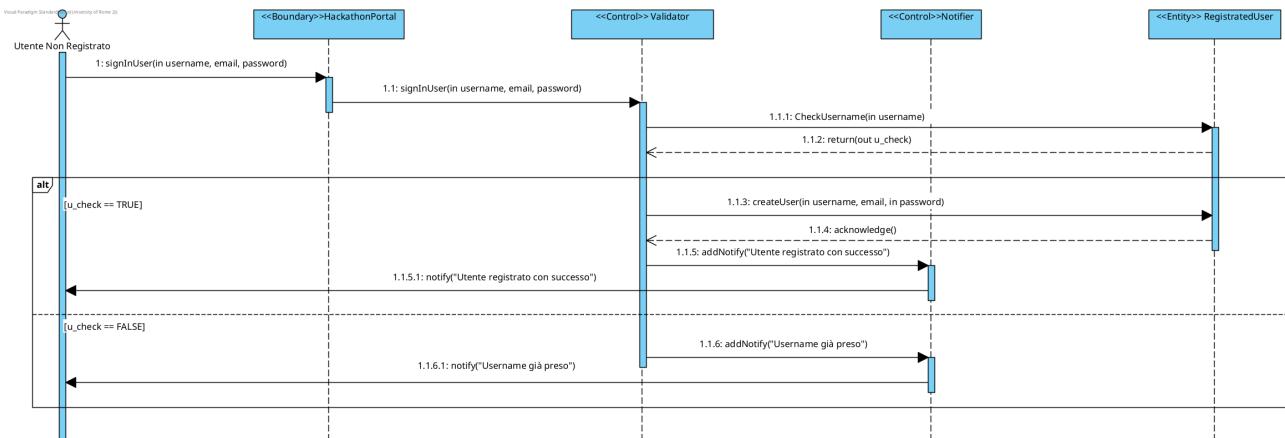


Figura 42: Sequence Diagram "Effettua Registrazione"

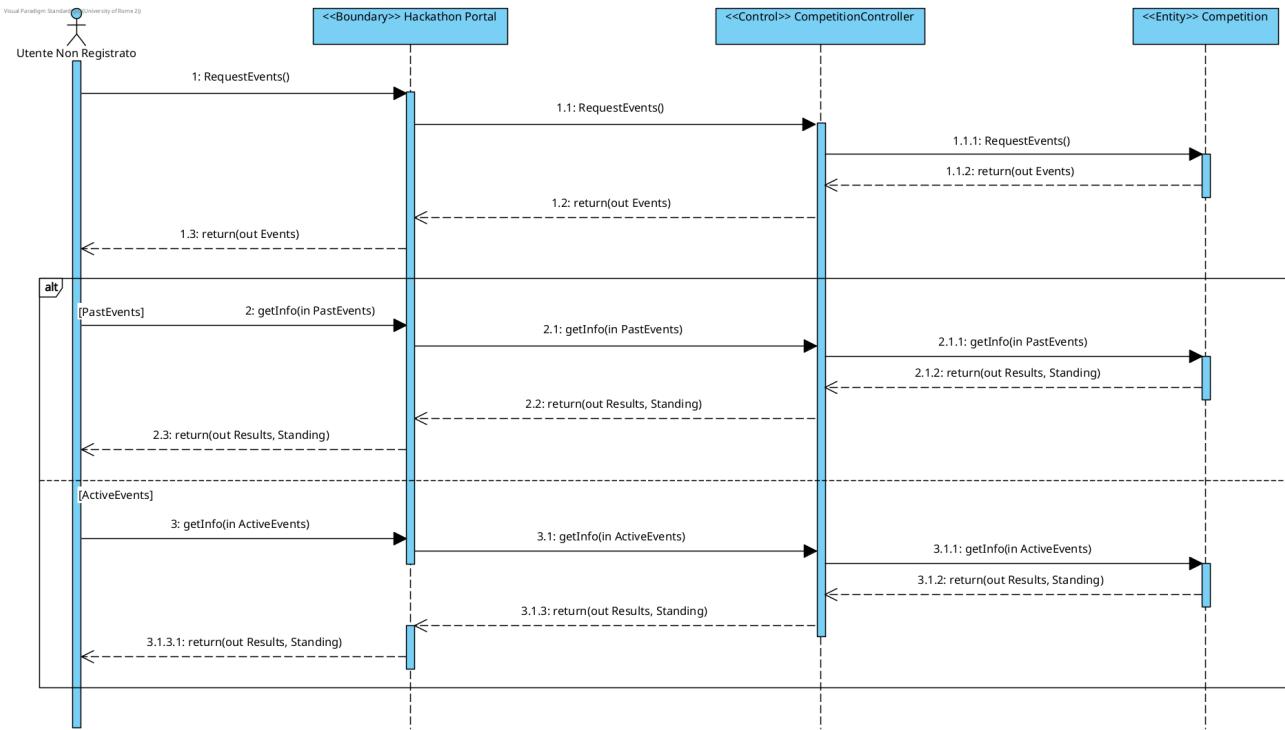


Figura 43: Sequence Diagram "Consulta Eventi"

5.2.5 Sequence Diagrams Team

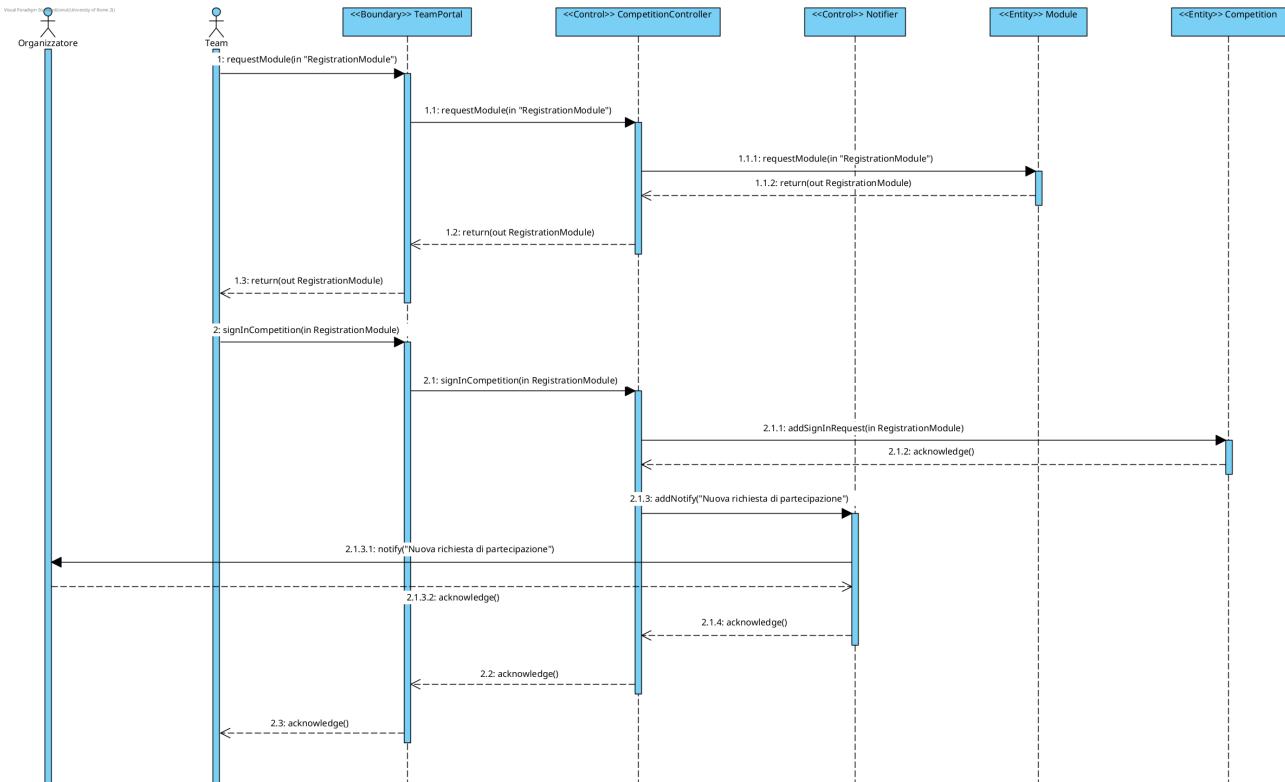


Figura 44: Sequence Diagram "Iscrizione Competizione"

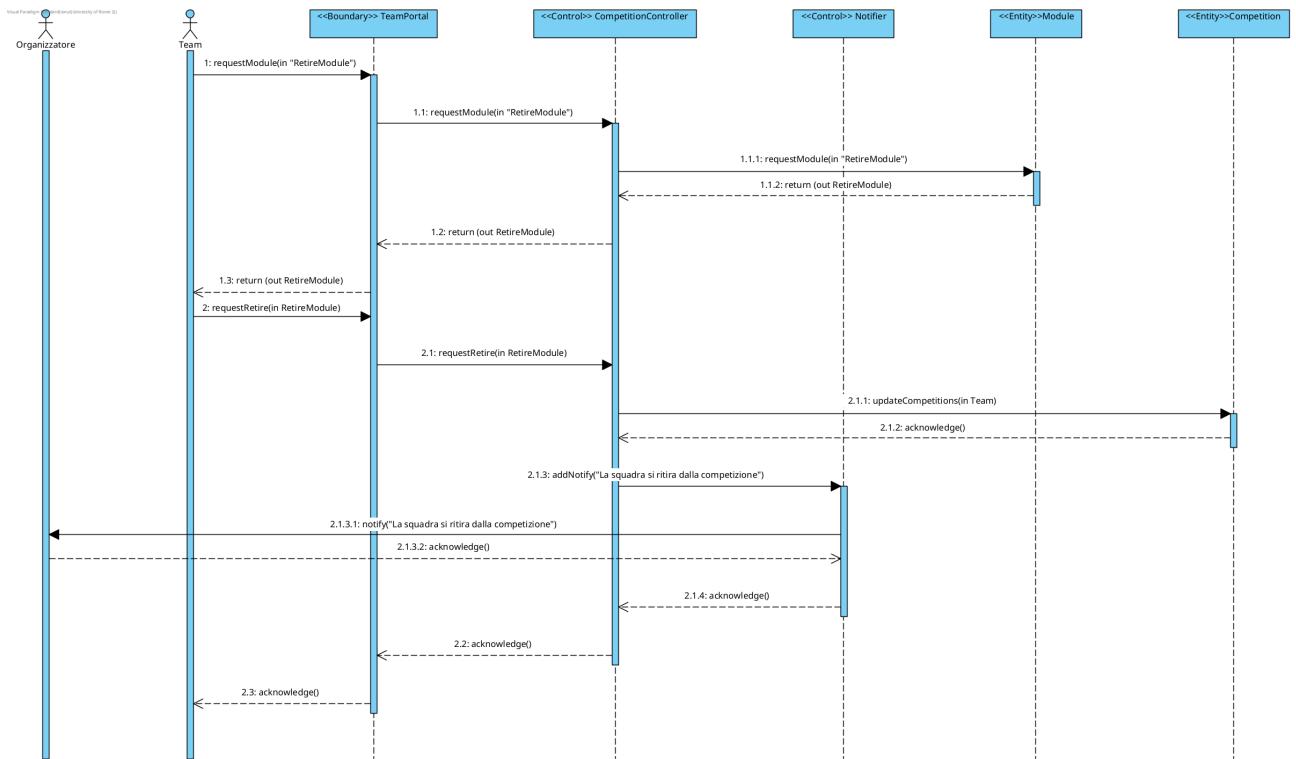


Figura 45: Sequence Diagram "Ritiro Competizione"

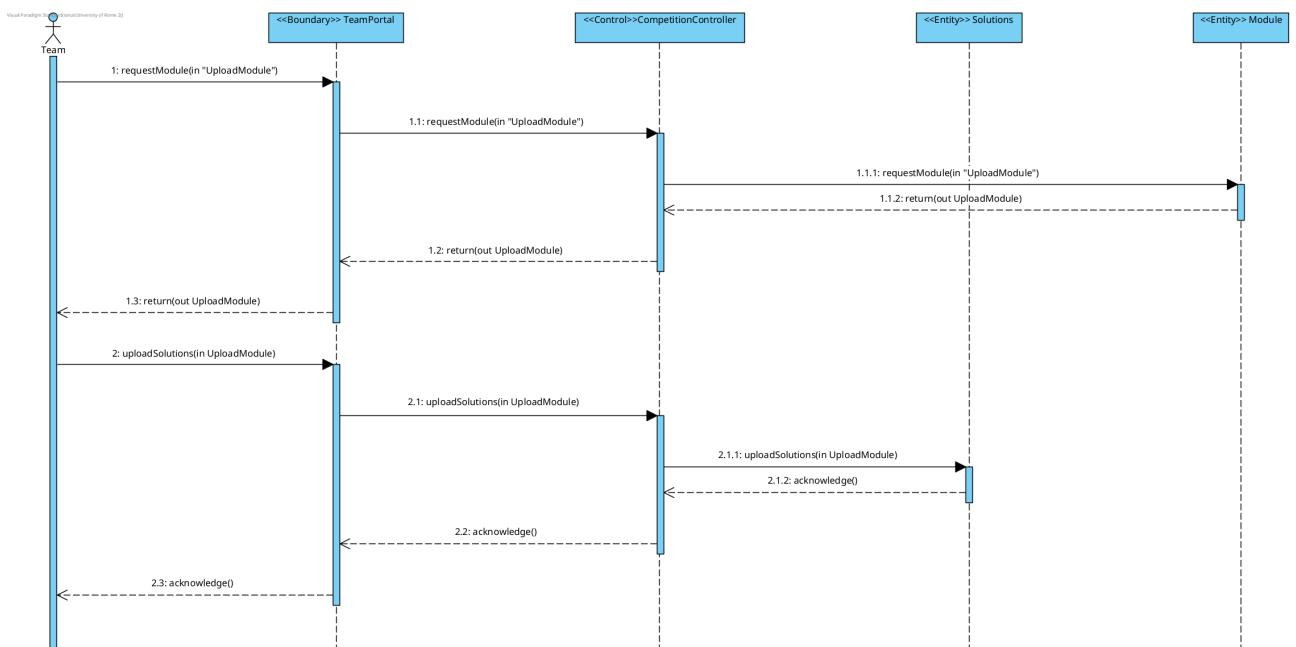


Figura 46: Sequence Diagram "Fornisce Soluzioni"

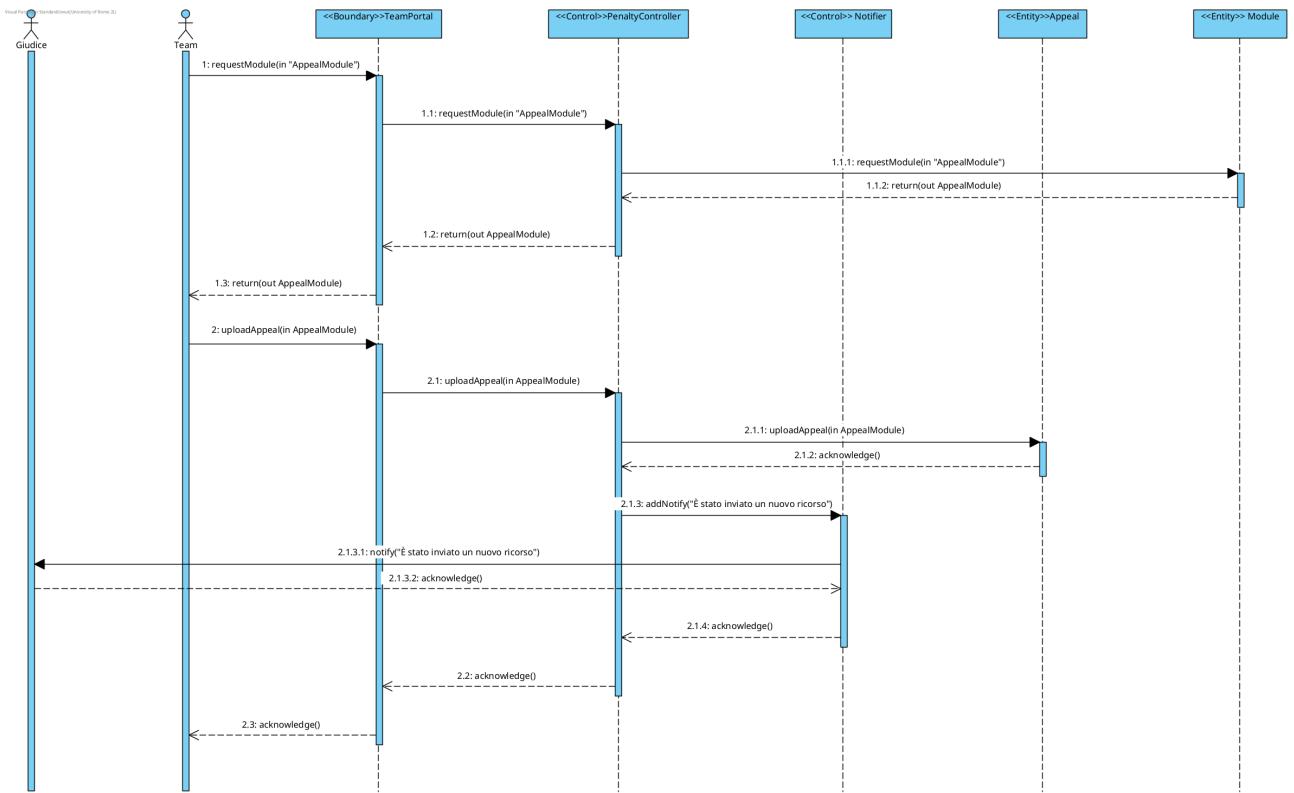


Figura 47: Sequence Diagram "Presenta Ricorso"

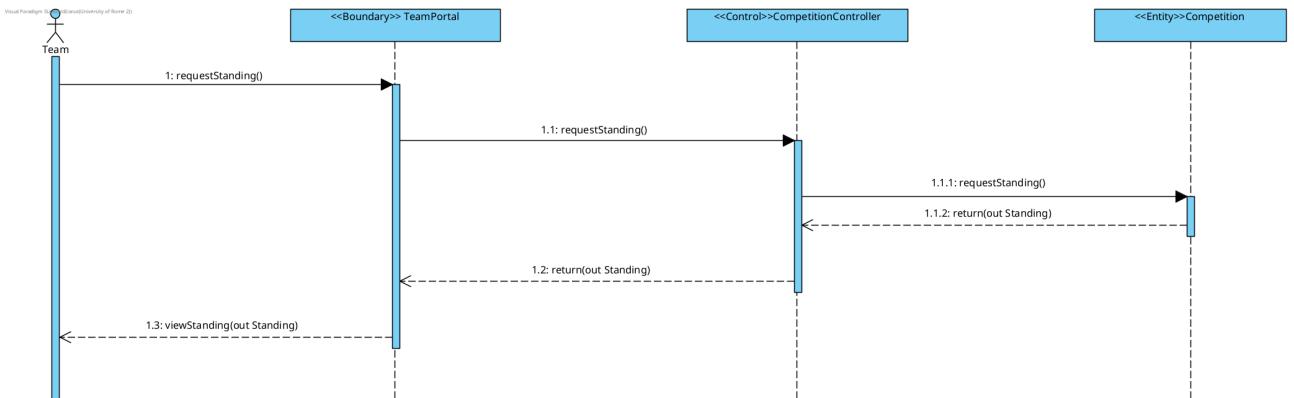


Figura 48: Sequence Diagram "Visualizza Classifica"

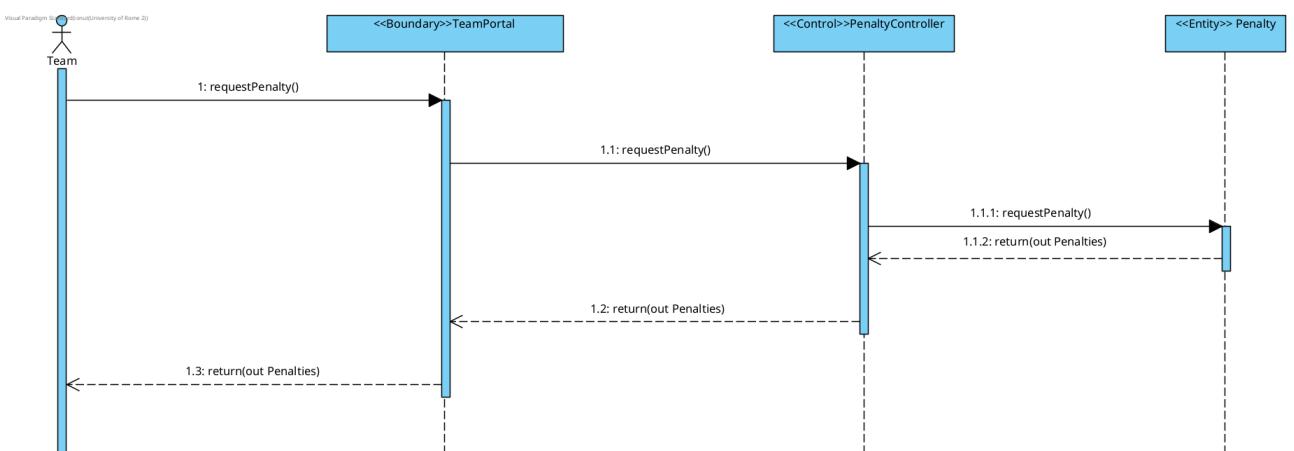
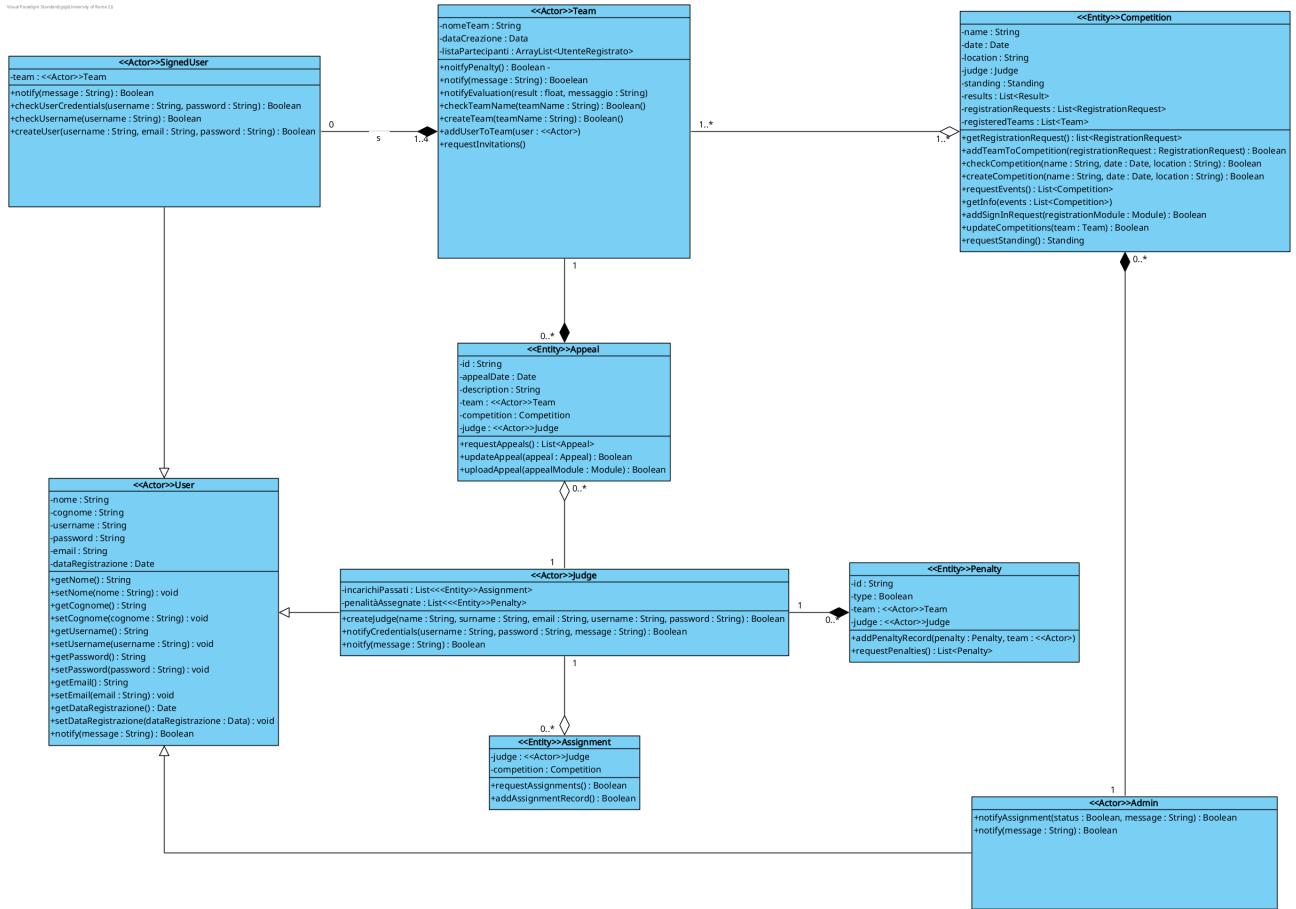


Figura 49: Sequence Diagram "Visualizza Sanzioni"

5.3 Class Diagrams

5.3.1 Class Diagram Unrefined



5.3.2 Class Diagram Refined



6 Design Patterns

6.1 Observer

Immaginiamo un sistema software per la gestione di un Hackathon universitario, in cui utenti registrati, Giudici e Organizzatori possono ricevere notifiche su eventi rilevanti, come:

- L'inizio o la fine di una competizione;
- Aggiornamenti in tempo reale della classifica;
- L'assegnazione di un punteggio da parte di un giudice;
- L'annuncio dei vincitori delle varie competizioni;
- La pubblicazione di aggiornamenti da parte dell'organizzatore.

Per evitare che gli utenti debbano controllare manualmente e con frequenza l'app o il portale, è stato progettato un meccanismo di iscrizione alle notifiche, sfruttando il Design Pattern Observer.

In questo contesto:

- Il **soggetto (Subject)** è rappresentato da un evento dell'Hackathon, ad esempio una "sfida aperta", "votazione terminata", o "risultati pubblicati";
- Gli **osservatori (Observers)** sono gli utenti registrati, i teams o il Giudice che hanno espresso interesse o sono protagonisti di quell'evento;
- Il **Notifier** è la componente centrale del sistema, già presente nell'architettura, utilizzata in più Sequence Diagram per propagare aggiornamenti tra i vari attori del sistema.

Grazie al pattern Observer, ogni entità coinvolta può manifestare o revocare l'interesse per specifici eventi, rendendo il sistema più reattivo e interattivo, e riducendo la necessità di polling manuale da parte dell'utente.

Esempi di Applicazione concreta del pattern:

- Un Team può iscriversi per ricevere notifiche sull'inizio della valutazione del proprio progetto.
- Un Giudice può essere notificato quando una nuova soluzione è pronta per la valutazione.
- Gli Utenti registrati ricevono aggiornamenti sulla base dei loro interessi.
- Gli Utenti non registrati, invece, possono seguire nel portale gli eventi attivi e ricevere notifiche sulle classifiche.

Il sistema, tramite la classe Notifier, permette a qualsiasi componente interessata (Observer) di registrarsi a uno o più eventi di interesse e ricevere aggiornamenti automaticamente al loro verificarsi. Questo approccio mantiene basso il carico sul sistema, migliorando la scalabilità e la responsività dell'interfaccia. Analizziamo l'ultimo esempio di applicazione concreta del pattern; Immaginiamo adesso un sistema in cui gli utenti non registrati vogliono richiedere notifiche inerenti agli eventi attivi della attuale competizione. Questo sistema potrebbe essere realizzato tramite un meccanismo di iscrizione, in cui ad ogni evento attivo è associato un insieme di utenti non registrati interessati allo svolgimento.

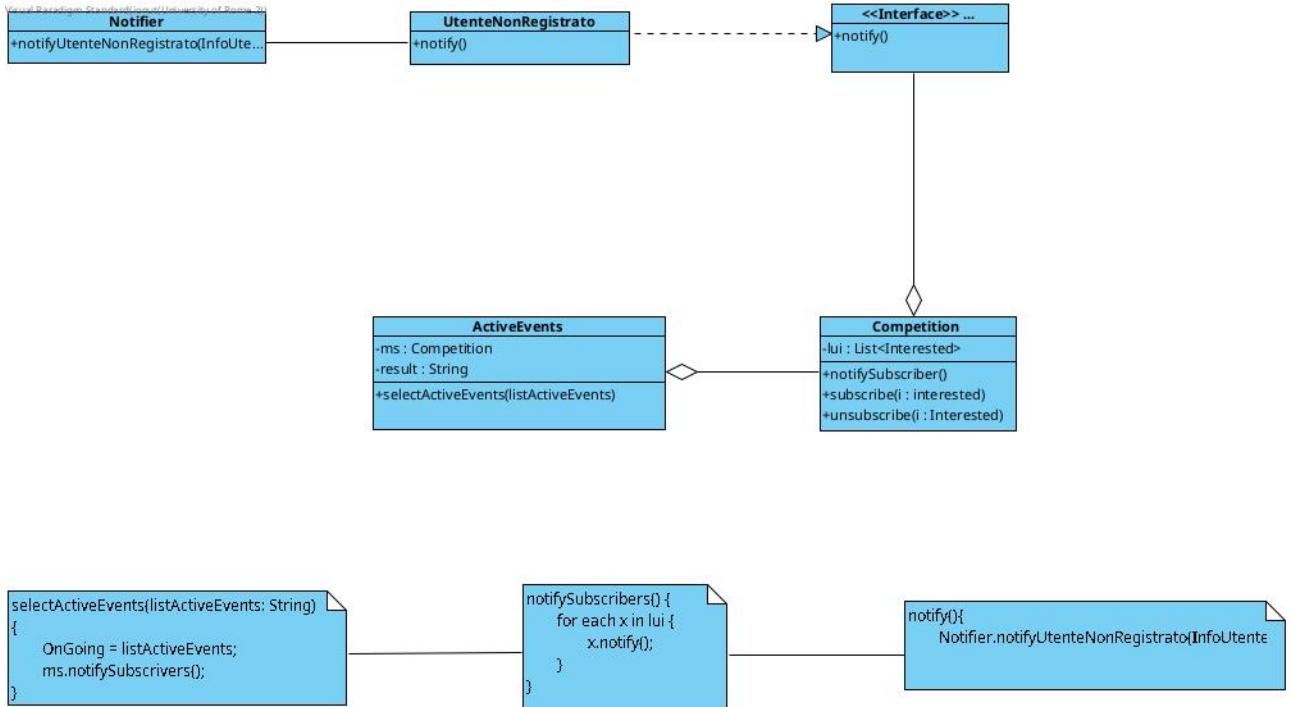


Figura 50: Design Pattern Observer

6.2 Decorator

Il diagramma UML illustra un sistema di notifiche realizzato secondo il **Design Pattern Decorator**, pensato per inviare messaggi attraverso molteplici canali (come WhatsApp, Telegram ed email) in modo flessibile e modulare. Grazie a questo approccio, non è necessario definire metodi separati per ogni combinazione possibile di canali: il sistema *decora* dinamicamente le notifiche in base alle preferenze configurate dall'utente.

Nel cuore dello schema troviamo l'interfaccia **Notifier**, che stabilisce un metodo standard **notify()**. Questa interfaccia è il punto di riferimento per il **Client**, che interagisce esclusivamente con essa per inviare notifiche. La classe **ConcreteNotifier** implementa l'interfaccia, fornendo la funzionalità di notifica di base. Successivamente, la classe **BaseNotifier** estende **ConcreteNotifier** e serve da fondamento per i decoratori specifici.

Le classi **EmailNotifier**, **TelegramNotifier** e **WhatsAppNotifier** derivano da **BaseNotifier** e aggiungono comportamenti specifici per l'invio tramite i rispettivi canali. Questi *decoratori* permettono al sistema di combinare dinamicamente più funzionalità, in modo da poter inviare notifiche su uno o più canali contemporaneamente, senza dover riscrivere o duplicare il comportamento di base.

Il diagramma evidenzia inoltre le relazioni tra i componenti: le frecce tratteggiate partono da **ConcreteNotifier** (e, indirettamente, da **BaseNotifier**) verso l'interfaccia **Notifier** per rappresentare le realizzazioni (ossia l'implementazione dell'interfaccia), mentre le relazioni di ereditarietà sono usate per mostrare come i vari decoratori amplino le funzionalità del sistema.

Questo schema permette al sistema di essere altamente scalabile e facilmente estendibile: se in futuro si volesse aggiungere un nuovo canale di notifica, basterebbe creare un nuovo decoratore che eredita da **BaseNotifier**, mantenendo inalterata la struttura centrale e sfruttando il meccanismo dinamico del pattern **Decorator**.

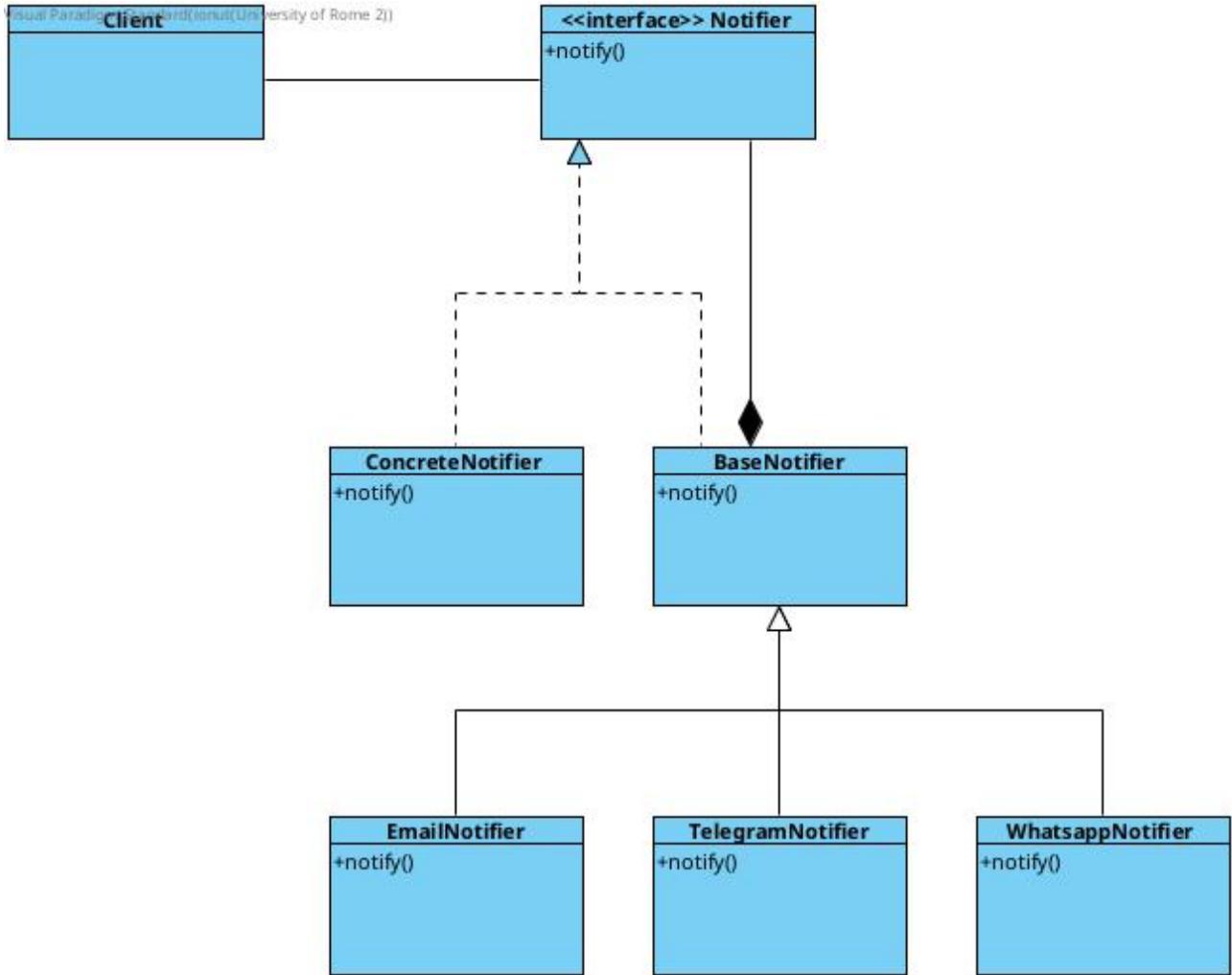


Figura 51: Design Pattern Decorator

6.3 Strategy

Lo *Strategy Pattern* è un design pattern comportamentale basato su oggetti che permette di definire una famiglia di algoritmi, incapsularli e renderli intercambiabili indipendentemente dal client che li utilizza.

Nel contesto della piattaforma **HackathonPortal**, utilizzata per gestire tutte le attività legate alle edizioni degli hackathon universitari, è fondamentale poter gestire *criteri di valutazione diversi o modalità alternative di calcolo del punteggio*. Alcuni esempi includono:

- Valutazione basata sulla **velocità di esecuzione dell'algoritmo** (efficienza).
- Valutazione basata sulla **qualità della soluzione** (leggibilità, struttura del codice).
- Adozione di **bonus o penalità**, ad esempio per soluzioni consegnate in ritardo ma di alta qualità.

Attraverso lo *Strategy Pattern*, possiamo implementare ciascuno di questi criteri come una **strategia separata**, tutte conformi alla stessa interfaccia, ma con logiche di calcolo differenti.

Questo approccio consente agli utenti della piattaforma di **visualizzare la classifica** secondo il criterio di valutazione più adatto alle proprie esigenze, rendendo l'esperienza più personalizzabile.

Inoltre, l'adozione del pattern Strategy semplifica l'estensione del sistema: **aggiungere un nuovo criterio di valutazione** richiederà solo l'implementazione di una nuova strategia, senza modificare il codice esistente. Infine, consente anche una **rapida selezione e sostituzione dell'algoritmo di punteggio**, rendendo l'applicazione più flessibile e manutenibile.

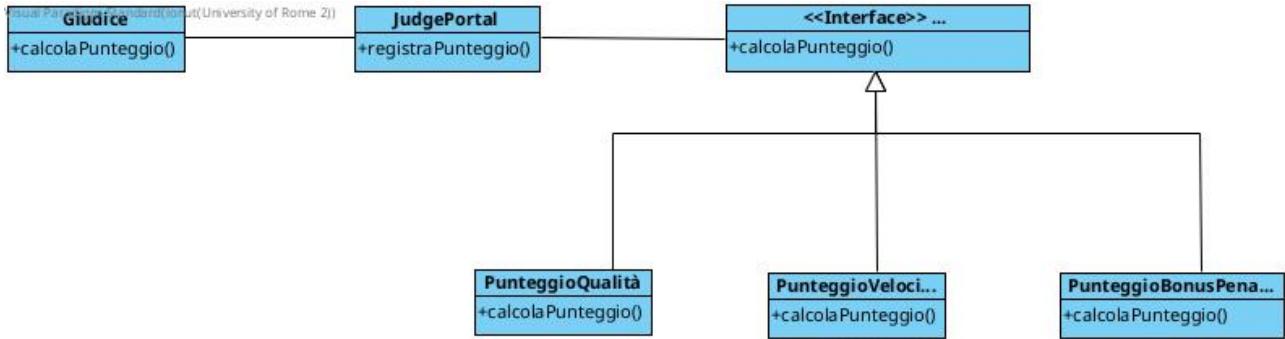


Figura 52: Design Pattern Strategy

6.4 Factory

Nel contesto della progettazione orientata agli oggetti, il design pattern **Factory** rappresenta una soluzione elegante per gestire la creazione di oggetti in modo flessibile e centralizzato. Questo pattern permette di definire un'interfaccia comune per la costruzione di oggetti, demandando alle sottoclassi o a componenti specifici la responsabilità di decidere, a **tempo di esecuzione**, quale classe concreta istanziare.

Nel nostro progetto, la gestione degli utenti riveste un ruolo centrale, in quanto il sistema prevede diverse tipologie di utenza: **utente registrato, giudice e admin** (o **organizzatore**). Ognuna di queste classi presenta comportamenti, responsabilità e privilegi differenti all'interno della piattaforma. L'utilizzo del pattern Factory ci consente di **centralizzare la logica di creazione degli utenti**, garantendo una separazione delle responsabilità tra chi utilizza gli oggetti e chi li costruisce, e facilitando al contempo la manutenzione e l'estensibilità del sistema. Ad esempio, in futuro potrebbe essere semplice aggiungere un nuovo tipo di utente senza dover modificare il codice client che interagisce con la factory.

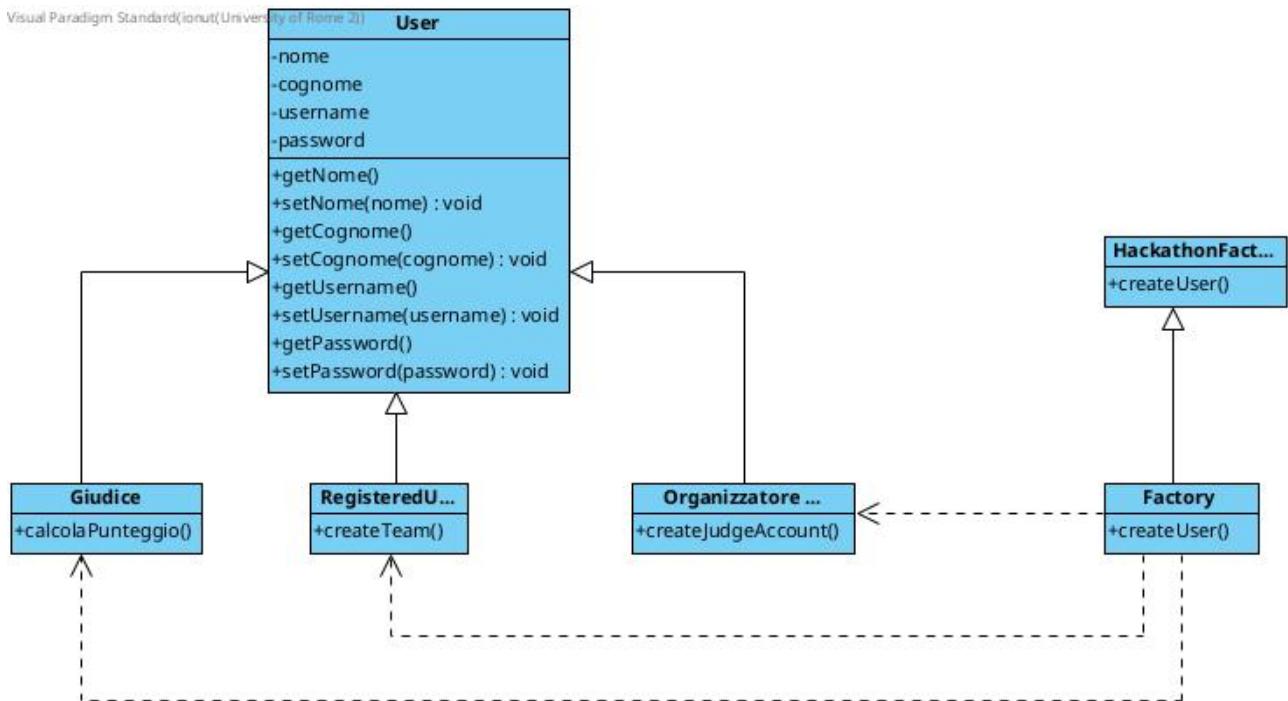


Figura 53: Design Pattern Factory

6.5 Singleton

Dopo aver creato i Sequence Diagrams e il Class Diagram Raffinato, è emerso che la classe "Notifier" veniva frequentemente referenziata. Per questo motivo, si è deciso di applicare il Design Pattern Singleton, così da garantire un punto di accesso globale alla classe mantenendo, al contempo, una singola istanza per l'intero funzionamento del software. Questo approccio è stato scelto per una ragione semplice ma fondamentale: in certi casi, è importante che una determinata

classe abbia una sola istanza in tutta l'applicazione. Pensiamo ad esempio a un gestore notifiche, come nel nostro caso: sarebbe complicato (e potenzialmente pericoloso) avere più oggetti che cercano di fare lo stesso lavoro, magari in modo incoerente. Il Singleton risolve questo problema, permettendo un accesso centralizzato e sicuro all'unica istanza disponibile, semplificando così lo sviluppo e riducendo il rischio di errori.

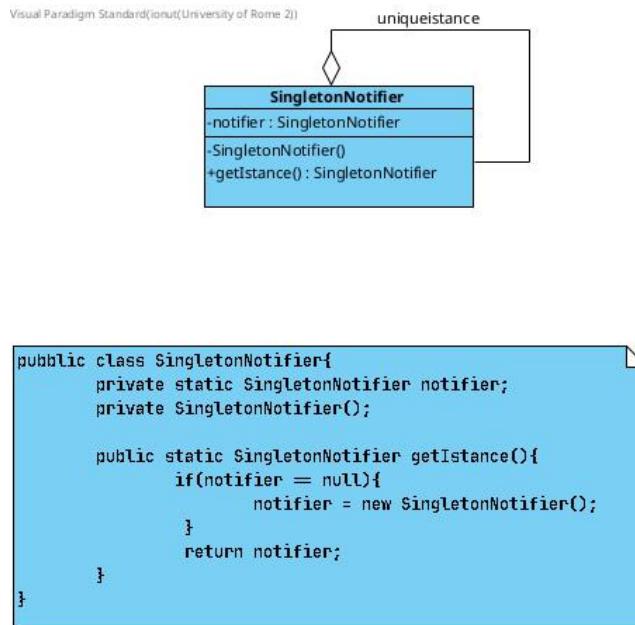


Figura 54: Design Pattern Singleton