

# CarpHaul

Documento di progetto



Membri del team di sviluppo:

Giacomo Cerino  
Jacopo Jop  
Luca Levita

Gruppo 6

# SOMMARIO:

Abstract.....	4
Requisiti .....	5
Vocabolario .....	7
Tabella dei Requisiti .....	8
Analisi dei Requisiti .....	11
Casi d'Uso .....	11
Scenari .....	11
Analisi del Rischio.....	19
Tabella di valutazione dei beni.....	19
Tabella Minacce/Controlli .....	19
Analisi Tecnologica della Sicurezza.....	20
Security Use Case & Misuse Case Scenari .....	22
Analisi del Problema.....	26
Analisi Documento dei Requisiti: Analisi delle Funzionalità.....	26
Analisi Documento dei Requisiti: Analisi dei Vincoli .....	32
Analisi Documento dei Requisiti: Analisi delle Interazioni.....	34
Analisi Ruoli e Responsabilità.....	36
Scomposizione del Problema .....	39
Creazione Modello del Dominio.....	40
Architettura Logica: Struttura.....	42
Diagramma dei package.....	42
Diagramma delle classi.....	42
Architettura Logica: Interazione .....	45
Architettura Logica: Comportamento .....	49
Piano di Lavoro .....	49
Piano del Collaudo.....	51
Progettazione .....	54
Progettazione Architetturale .....	54
Requisiti non funzionali.....	54

Scelta dell'architettura.....	55
Pattern Architetturali.....	55
Architettura del sistema.....	56
Diagramma dei componenti .....	56
Scelte Tecnologiche.....	57
Progettazione di Dettaglio .....	57
Struttura.....	57
Interazione .....	71
Comportamento.....	77
Progettazione della Persistenza.....	78
Diagramma E-R .....	78
Log.....	79
Formato File Log.....	79
Progettazione del Collaudo.....	79
Progettazione per il Deployment.....	85
Deployment .....	85
Artefatti .....	85
Deployment Type-Level .....	86

## **Abstract**

Web app che permette ad un organizzatore di creare eventi e di invitare i propri amici registrati. Alla creazione di ogni evento verrà fornito un codice da condividere con tutti gli invitati.

Coloro i quali inseriranno il codice nell'applicazione diventeranno partecipanti e dovranno inserire i loro dati (comprese informazioni sulla necessità di andata, ritorno o entrambi) e se dispongono di un veicolo (specificando anche i posti liberi). Sarà l'applicazione stessa a organizzare il percorso che i guidatori dovranno fare scegliendo i tragitti più corti per passare a prendere gli altri partecipanti che sono stati assegnati loro. In questo modo si potranno evitare giri a vuoto, consumo di benzina e uso eccessivo di macchine.

Al momento della partenza verrà fornito agli utenti un orario approssimativo di quando arriverà la macchina

## Requisiti

Si necessita di un applicazione che permette a un gruppo di invitati di organizzarsi automaticamente per raggiungere un evento in un determinato luogo (in tal giorno e tal ora), disponendo di un limitato numero di veicoli.

L'organizzatore deve essere in grado di creare l'evento inserendo tutte le informazioni del caso (data, ora di inizio, ora di fine, luogo) e di condividere facilmente l'invito con tutti gli invitati, per esempio attraverso un link. Inoltre l'organizzatore deve essere in grado di apportare modifiche all'evento creato, in caso di errori o cambi di programma.

Ogni invitato che aderisce deve essere in grado di specificare:

- Il suo nominativo se parteciperà
- Se necessita solo dell'andata, solo del ritorno o di entrambi
- Se dispone di una macchina (in tal caso anche i posti liberi che ha in macchina) e se la rende disponibile anche al ritorno (in tal caso anche i posti liberi che ha in macchina)
- La via e il civico in cui dovrà essere prelevato o dal quale partirà. E nel caso di necessità al ritorno, la via e il civico in cui dovrà essere lasciato.

Tutti gli invitati compreso l'organizzatore, (il quale può essere anche fruitore del servizio), devono essere registrati con un username e una password, in modo da poter stabilire se hanno già confermato la partecipazione o meno.

Gli utenti devono specificare un recapito attendibile in modo da poter essere contattati e rintracciati con facilità al momento del prelievo.

Un invitato registrato può, tramite il link di invito, visionare le conferme e la lista dei partecipanti, dopodiché può aggiungere la sua partecipazione e i suoi dati, o modificare quelli precedentemente inseriti.

È necessario conservare i dati degli utenti e i dati relativi agli eventi in un database.

L'applicazione ogni volta che un invitato aggiunge i suoi dati, ricalcola i percorsi e fornisce un programma temporaneo, completo anche di percorsi che sono però da considerare non definitivi, ma saranno disponibili in qualsiasi momento.

Al momento della partenza l'applicazione fornirà a ciascun invitato un programma, esso verrà considerato definitivo, ma di fatto sarà ancora modificabile e consisterà in:

- Se l'invitato dispone di una macchina, gli verrà indicato il percorso\* e la lista di invitati che dovrà caricare.
- Se l'invitato deve essere prelevato gli verrà comunicato il percorso del guidatore ed un orario approssimativo di arrivo.

Al momento del ritorno a ciascun invitato verrà comunicato il percorso che il guidatore dovrà seguire.

È necessario monitorare il numero di macchine disponibili in relazione agli invitati da trasportare e se necessario richiedere a chi non ha reso disponibile la macchina, di cercare di ottenerne una in modo da fornire a tutti un passaggio.

Se un invitato privo di macchina risulta particolarmente lontano o isolato gli verrà consigliato di reperire un mezzo di trasporto in modo da non allungare troppo il percorso di un guidatore, oppure di farsi trovare in un luogo più vicino.

## Precisazioni

\* come percorso si intendono una serie di indicazioni, fornite dall'applicazione esterna GoogleMaps.

## Domande da chiarire

- Un utente può inserire più richieste di passaggio in posti diversi? —> no, bisogna organizzarsi prima, comunque nel caso di deviazioni dal percorso, le tappe rimangono immutate.

- Se un invitato non vuole utilizzare l'applicazione ma per motivi organizzativi deve essere passato a prendere in che modo ci si comporta? —> Si registra, altrimenti non potrà usufruire del servizio.
- Se facciamo un login è necessario utilizzare una mail? (Come recapito e reminder o come recovery dell'account) —> si, per il recovery e per l'username. Per ora si utilizza un cellulare come recapito.

Non è previsto che qualcuno esprima preferenze su chi trasportare perché ciò renderebbe vano l'algoritmo di ottimizzazione.

# Vocabolario

Voce	Definizione	Sinonimi
<b>Evento</b>	Attività programmata creata da uno e un solo organizzatore. prevista in un certo luogo, in una certa data, in un arco di tempo definito tra un orario di inizio e un orario di fine. Ha una lista dei partecipanti. Ha un nome e una descrizione	
Dati di un evento	Dati che caratterizzano un evento come Data, ora di inizio e di fine, luogo, lista dei partecipanti, nome e descrizione	dati, informazioni essenziali
Invito	Codice generato e condivisibile dall'organizzatore utilizzabile da un invitato per unirsi alla lista dei partecipanti di un evento	
Username	Chiave primaria per riconoscere un utente, è previsto che sia un indirizzo di posta elettronica ( <b>esistente</b> )	Mail
Password	Codice alfanumerico di almeno 8 caratteri	
Nominativo	Nome identificativo di un Utente per un evento	Nickname
Credenziali	Coppia di username e password necessaria per accedere al sistema	
<b>Utente registrato</b>	Utente che possiede un username, password, perciò può accedere alle funzionalità dell'applicativo	
<b>Utente non registrato</b>	Utente che non si è ancora registrato	
Partecipante	Utente registrato che fa parte di una lista di partecipanti, è identificato da un Nominativo (può essere anche un <b>organizzatore</b> )	Partecipante registrato
Organizzatore	Utente registrato che crea, gestisce e condivide un evento tramite invito	
Invitato	Utente registrato che ha la possibilità di partecipare ad un evento soltanto sotto invito dell'organizzatore. Una volta accettato l'invito diventerà un partecipante	
Lista di partecipanti	Insieme dei partecipanti ad un evento ( invitati + organizzatore, se partecipa)	
Gruppo	Elenco di partecipanti che sono passeggeri di un veicolo, o in caso di autonomia, elenco composto da un solo partecipante	Macchinata
Veicolo	Mezzo di trasporto di una o più persone (macchina, motorino, minivan)	
Posti disponibili	Numero di posti che l'algoritmo di allocazione dei posti può assegnare. Quindi posti disponibili nella macchina escluso il guidatore	
Guidatore	Partecipante munito di veicolo	
Luogo	Coordinate geografiche o indirizzo di un utente o di un evento	destinazione, partenza, tappa
Andata	Viaggio in cui tutti i partecipanti convergono al luogo dell'evento partendo da luoghi anche diversi	
Ritorno	Viaggio in cui tutti i partecipanti partono dallo stesso luogo per raggiungere il luogo di ritorno	

<b>Voce</b>	<b>Definizione</b>	<b>Sinonimi</b>
Viaggio	Tragitto composto da partenza, destinazione e con la possibilità di avere delle tappe intermedie	Percorso, Tragitto, Itinerario
Recapito	Canale di comunicazione utilizzabile dai partecipanti per rintracciarsi a vicenda	Numero di telefono
Programma	Documento contenente il tragitto di un veicolo, le tappe del viaggio, gli orari approssimativi di arrivo alle tappe e alla destinazione, i nominativi dei partecipanti da caricare e i loro rispettivi recapiti. Comprende per chi non fa parte di un gruppo, perchè autonomo o sprovvisto temporaneamente di passaggio, le indicazioni per raggiungere il luogo dell'evento	
Abbandonare un evento	Azione di un qualsiasi partecipante che prevede la rimozione del proprio nominativo dalla lista dei partecipanti	

## Tabella dei Requisiti

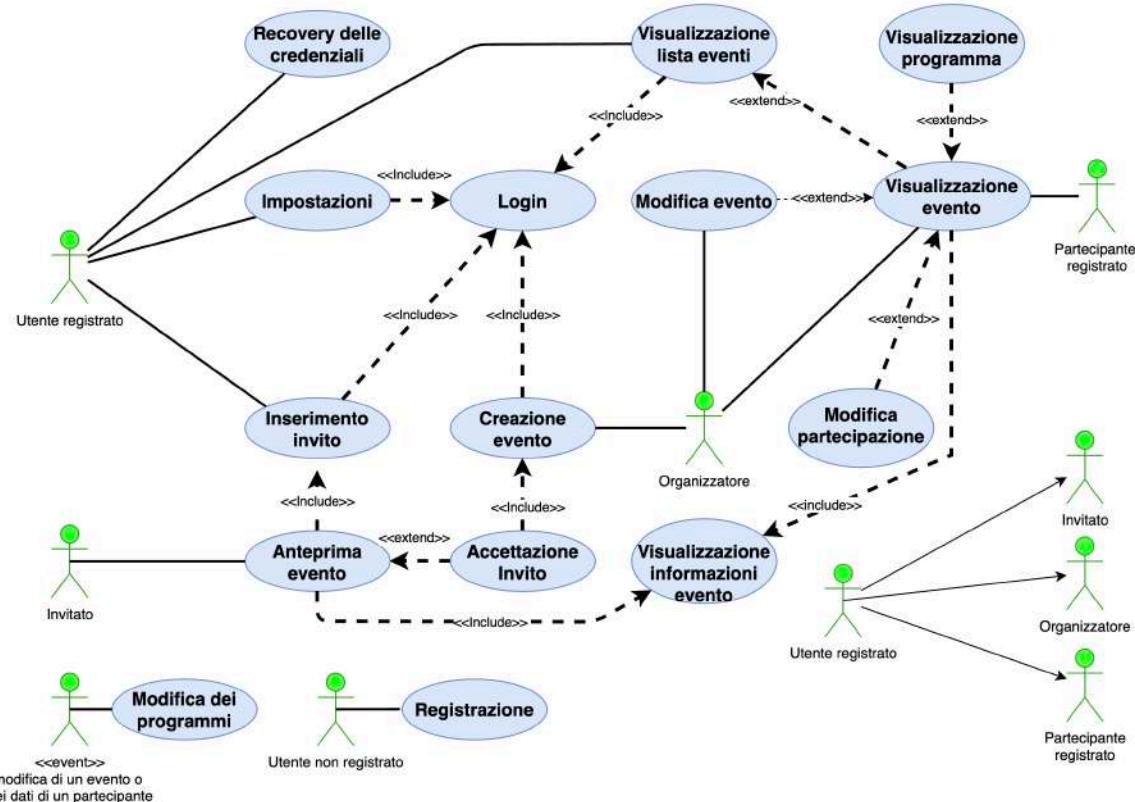
<b>Id</b>	<b>Requisito</b>	<b>Tipo</b>
1F	Registrazione di nuovi utenti	Funzionale
2F	Creazione di un evento con relative informazioni	Funzionale
3F	Modifica di un evento	Funzionale
4F	Generazione invito	Funzionale
5F	Durante la visualizzazione delle Informazioni evento verrà mostrata soltanto la lista dei nominativi dei partecipanti e i dati relativi all'evento	Funzionale
5.1F	I recapiti dei partecipanti e i loro username verranno mostrati solamente ai facenti parte dello stesso gruppo	Funzionale
6F	Conferma di partecipazione e inserimento dati	Funzionale
6.1F	Specificare disponibilità veicolo, o necessità di passaggio (inserire luogo di partenza) per l'andata	Funzionale
6.2F	Specificare disponibilità veicolo, o necessità di passaggio (inserire luogo di destinazione) per il ritorno	Funzionale
7F	Modifica dei dati inseriti da un utente rispetto ad un evento	Funzionale
8F	Generazione gruppi per ogni veicolo basati sulla vicinanza al luogo di partenza del guidatore	Funzionale
9F	Calcolo itinerario più breve attraverso un algoritmo di terzi	Funzionale
10F	Visualizzazione programma	Funzionale

Id	Requisito	Tipo
11F	Nel caso in cui un utente sprovvisto di veicolo che necessita di un passaggio, si trovi molto distante da tutti gli altri partecipanti, è necessario che gli venga consigliato di provvedere da solo al raggiungimento dell'evento o quantomeno di trovare il modo di avvicinarsi alla zona di partenza di un guidatore. Egli verrà inserito in un gruppo dedicato, con lui solo	Funzionale
12F	Ogni invitato deve essere registrato	Funzionale
16F	È previsto nel caso in cui un utente si dimentichi la password, un servizio di recovery tramite posta elettronica	Funzionale
17F	L'organizzatore ha il potere di eliminare la <b>partecipazione</b> di un qualsiasi partecipante, compreso se stesso	Funzionale
18F	Un partecipante può decidere di abbandonare un evento a cui aveva aderito. Egli verrà eliminato dalla lista dei partecipanti	Funzionale
17.1F/ 18.1F	Se l'organizzatore elimina se stesso dalla lista dei partecipanti o (equivalentemente) abbandona l'evento, esso rimane in grado di amministrare l'evento, senza comparire nella lista dei partecipanti	Funzionale
18.2F	L'Organizzatore che ha abbandonato l'evento può decidere di riunirsi in un qualsiasi momento (Anche se il numero massimo è stato raggiunto)	Funzionale
19F	Se un invitato proverà a unirsi ad un evento senza posti liberi, verrà avvisato con un messaggio di errore	Funzionale
20F	Se un evento fallisce nella creazione, l'organizzatore verrà informato con un messaggio di errore	Funzionale
21F	Sarà possibile modificare lo username, e i dati di default	Funzionale
22F	Previsto un log che riporti l'elenco delle operazioni fatte	Funzionale
23F	Meccanismo di anti-dos	Funzionale
24F	È richiesto il login soltanto una volta, all'inizio di ogni sessione di utilizzo	Funzionale
25F	Possibilità di visualizzare tutti gli eventi a cui si è associati	Funzionale
26F	A un invitato verrà presentata un'anteprima dell'evento che ne darà solo le informazioni essenziali, senza divulgare gli username ma solo i nominativi dei partecipanti	Funzionale
	equivalente al 5 perché essendo il 5 riferito ad una situazione più generale che è "padre" di questa, ed avendo una condizione più stringente, è inutile specificare questo requisito.	
27F	Possibilità per ogni partecipante di visionare tutti i dati relativi ad un evento, i recapiti del proprio gruppo e i percorsi	Funzionale
1NF	Un evento ha uno e un solo organizzatore	Non funzionale
2NF	Non si può superare un certo numero di posti nei veicoli	Non funzionale
3NF	Semplicità di navigazione tra le diverse maschere	Non funzionale
4NF	Il sistema sarà distribuito. Sarà utilizzato front-end su un dispositivo in grado di memorizzare i dati relativi ai programmi, almeno temporaneamente in una cache	Non funzionale

<b>Id</b>	<b>Requisito</b>	<b>Tipo</b>
5NF	Sarà necessario predisporre un database che manterrà i dati degli utenti registrati in modo persistente, e che manterrà soltanto fino alla fine di ogni evento le informazioni relative ad esso	Non funzionale
6NF	Un evento DEVE avere un numero massimo di partecipanti (ad esempio di default 50 e nel caso l'organizzatore preveda che sarà un evento molto grande potrà aumentarlo fino a 200)	Non funzionale
7NF	Il database dove sono conservati i dati deve essere direttamente accessibile solo da un amministratore	Non funzionale
8NF	L'applicazione interroga il database in maniera sicura e controllata	Non funzionale
9NF	Bisognerà utilizzare Google Maps se possibile	Non funzionale

# Analisi dei Requisiti

## Casi d'Uso



## Scenari

<b>Titolo</b>	Impostazioni
<b>Descrizione</b>	Possibilità di aggiornare lo username da parte dell'utente, di inserire un indirizzo predefinito, un recapito predefinito e un nominativo predefinito
<b>Attori</b>	Utente registrato
<b>Relazioni</b>	Login
<b>Precondizioni</b>	L'utente registrato ha effettuato il login
<b>Postcondizioni</b>	L'utente ha aggiornato i suoi dati. Il log viene aggiornato
<b>Scenario principale</b>	1. L'utente ha la possibilità di effettuare cambiamenti dei dati a lui associati
<b>Scenari alternativi</b>	1. L'aggiornamento non va a buon fine es. nuovo username non valido, nuova mail non esistente. In questo caso le informazioni non vengono modificate.
<b>Requisiti non funzionali</b>	Semplicità di navigazione tra le diverse maschere, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Visualizzazione lista eventi
<b>Descrizione</b>	Visualizzazione di tutti gli eventi organizzati dall'utente o a cui l'utente partecipa
<b>Attori</b>	Utente registrato
<b>Relazioni</b>	Login, Visualizzazione evento
<b>Precondizioni</b>	L'utente registrato ha effettuato il login
<b>Postcondizioni</b>	
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. L'utente visiona la lista degli eventi a cui partecipa</li> <li>2. L'utente sceglie l'evento da visualizzare singolarmente</li> </ol>
<b>Scenari alternativi</b>	
<b>Requisiti non funzionali</b>	Semplicità di navigazione tra le diverse maschere, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Registrazione
<b>Descrizione</b>	Un utente non registrato inserisce la mail e sceglie una password. Così facendo diventa un utente registrato.
<b>Attori</b>	Utente non registrato
<b>Relazioni</b>	
<b>Precondizioni</b>	La mail non deve essere già utilizzata
<b>Postcondizioni</b>	L'utente si è registrato ed è in possesso delle credenziali d'accesso. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Un utente si vuole registrare perchè non ha un account</li> </ol>
<b>Scenari alternativi</b>	<ol style="list-style-type: none"> <li>1. Un utente prova a registrarsi con una mail già in uso. Viene mostrato un messaggio di errore.</li> </ol>
<b>Requisiti non funzionali</b>	Attenersi alla normativa gdpr riguardo ai dati come la mail
<b>Punti aperti</b>	

<b>Titolo</b>	Login
<b>Descrizione</b>	Un utente esegue l'accesso all'inizio della sessione e rimane autenticato fino all'uscita volontaria dall'applicativo
<b>Attori</b>	Utente registrato
<b>Relazioni</b>	Creazione evento, Impostazioni, Visualizzazione lista eventi, Inserimento invito
<b>Precondizioni</b>	Un utente vuole eseguire l'accesso
<b>Postcondizioni</b>	Il log viene aggiornato
<b>Scenario principale</b>	L'utente registrato inserisce le sue credenziali e il sistema dopo le opportune verifiche gli presenta la pagina principale
<b>Scenari alternativi</b>	L'utente inserisce delle credenziali non valide: il sistema allora ripropone il modulo di login con un messaggio di errore.
<b>Requisiti non funzionali</b>	Protezione dei dati
<b>Punti aperti</b>	

<b>Titolo</b>	Recovery delle credenziali
<b>Descrizione</b>	Possibilità di reimpostare la password di accesso tramite la mail (cioè l'username)
<b>Attori</b>	Utente registrato
<b>Relazioni</b>	
<b>Precondizioni</b>	L'utente registrato ha dimenticato la password e non riesce ad accedere
<b>Postcondizioni</b>	L'utente riceverà per email un link che lo reindirizzerà a una pagina dove gli verrà permesso di cambiare la sua password. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. L'Utente inserisce il suo username e richiede l'invio della mail</li> <li>2. L'Utente riceve per mail un link per reimpostare la sua password</li> <li>3. L'utente reimposta la password.</li> </ol>
<b>Scenari alternativi</b>	<ol style="list-style-type: none"> <li>1. L'Utente richiede la mail ma poi non cambia password. La password rimane la stessa</li> <li>2. L'utente non si ricorda l'username. Si fa un altro account</li> </ol>
<b>Requisiti non funzionali</b>	Protezione dei dati
<b>Punti aperti</b>	

<b>Titolo</b>	Creazione evento
<b>Descrizione</b>	Funzionalità per inserire e creare le informazioni relative ad un evento nuovo
<b>Attori</b>	Utente registrato, Organizzatore
<b>Relazioni</b>	Login
<b>Precondizioni</b>	L'utente registrato ha effettuato il login
<b>Postcondizioni</b>	L'utente (agendo da organizzatore) ha inserito i dati dell'evento e può riscattare il codice d'invito. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. L'Organizzatore visiona una maschera in cui è possibile inserire i dati di un nuovo evento</li> <li>2. L'Organizzatore inserisce luogo, data di inizio e fine, ora di inizio e di fine, nome, descrizione, numero massimo di partecipanti</li> <li>3. L'Organizzatore viene reindirizzato momentaneamente ad accettazione invito in modo da fargli inserire i suoi dati personali</li> <li>4. L'Organizzatore viene riportato a Creazione evento</li> <li>5. Viene generato il primo gruppo contenente l'organizzatore e il suo rispettivo programma.</li> <li>6. Viene fornito all'Organizzatore il codice d'invito da condividere</li> <li>7. L'Organizzatore viene reindirizzato alla pagina principale.</li> </ol>
<b>Scenari alternativi</b>	<ol style="list-style-type: none"> <li>1. La creazione dell'evento viene interrotta, l'evento non viene creato</li> <li>2. Finché alcuni dati non sono validi, la creazione non può essere ultimata, l'Organizzatore riceverà messaggi di errore che gli comunicheranno quali dati è opportuno modificare</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di inserimento dati, chiarezza nel comunicare gli errori.
<b>Punti aperti</b>	

<b>Titolo</b>	Inserimento invito
<b>Descrizione</b>	Funzionalità per inserire il codice di invito condiviso da un Organizzatore
<b>Attori</b>	Utente registrato, Invitato
<b>Relazioni</b>	Login
<b>Precondizioni</b>	L'utente registrato ha effettuato il login
<b>Postcondizioni</b>	L'utente ha inserito il codice d'invito, Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Il codice d'invito è valido</li> <li>2. L'Utente viene reindirizzato all'anteprima dell'evento</li> </ol>
<b>Scenari alternativi</b>	<p>Scenario A</p> <ol style="list-style-type: none"> <li>1. Il codice d'invito non è valido</li> <li>2. L'utente viene reindirizzato alla pagina principale</li> </ol> <p>Scenario B</p> <ol style="list-style-type: none"> <li>1. L'utente è già un partecipante all'evento</li> <li>2. Viene visualizzato un messaggio di errore e l'Utente viene reindirizzato alla pagina principale</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di inserimento dati, chiarezza nel comunicare gli errori.
<b>Punti aperti</b>	

<b>Titolo</b>	Visualizzazione evento
<b>Descrizione</b>	Visualizzazione di un singolo evento a cui l'utente registrato partecipa
<b>Attori</b>	Partecipante Registrato, Organizzatore
<b>Relazioni</b>	Visualizzazione lista eventi, Modifica evento, Modifica partecipazione, Visualizzazione programma, Visualizzazione informazioni evento
<b>Precondizioni</b>	Il Partecipante Registrato ha scelto di visualizzare un evento specifico
<b>Postcondizioni</b>	
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>Il Partecipante Registrato visiona le informazioni relative all'evento</li> <li>Il Partecipante Registrato può scegliere se visualizzare il programma, se modificare la partecipazione, oppure se è organizzatore di modificare l'evento</li> </ol>
<b>Scenari alternativi</b>	<p>Scenario A</p> <ol style="list-style-type: none"> <li>L'evento viene cancellato dall'organizzatore. Il Partecipante registrato riceve un messaggio che glielo comunica e viene reindirizzato alla pagina principale.</li> </ol> <p>Scenario B</p> <ol style="list-style-type: none"> <li>Il Partecipante Registrato viene rimosso. Esso riceve un messaggio che glielo comunica e viene reindirizzato alla pagina principale.</li> </ol> <p>Scenario C</p> <ol style="list-style-type: none"> <li>Il Partecipante Registrato viene rimosso mentre sta visualizzando l'evento. Fino a quando non compie un'azione sarà possibile per lui vedere le informazioni dell'evento.</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di navigazione tra le diverse maschere, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Anteprima evento
<b>Descrizione</b>	Vengono visualizzate le informazioni basilari di un evento
<b>Attori</b>	Invitato
<b>Relazioni</b>	Inserimento invito, visualizzazione informazioni evento, Accettazione invito
<b>Precondizioni</b>	L'invitato ha inserito un codice di invito che ha una corrispondenza tra gli eventi esistenti
<b>Postcondizioni</b>	
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>Vengono visualizzate le informazioni fornite da Visualizzazione informazioni evento</li> <li>L'invitato può decidere se procedere all'accettazione (o all'annullamento dell'accettazione) dell'invito</li> </ol>
<b>Scenari alternativi</b>	
<b>Requisiti non funzionali</b>	Semplicità rappresentazione dei dati.
<b>Punti aperti</b>	

<b>Titolo</b>	Modifica evento
<b>Descrizione</b>	L'Organizzatore può modificare un evento creato da lui
<b>Attori</b>	Organizzatore
<b>Relazioni</b>	Visualizzazione evento
<b>Precondizioni</b>	Il Partecipante Registrato deve essere l'organizzatore dell'evento
<b>Postcondizioni</b>	I dati dell'evento possono essere stati modificati. Ciò causa la modifica dei programmi. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. L'Organizzatore modifica i dati dell'evento</li> <li>2. L'Organizzatore viene reindirizzato alla visualizzazione evento</li> </ol>
<b>Scenari alternativi</b>	<p>Scenario A</p> <ol style="list-style-type: none"> <li>1. L'Organizzatore elimina l'evento</li> <li>2. L'Organizzatore viene reindirizzato alla Pagina principale</li> </ol> <p>Scenario B</p> <ol style="list-style-type: none"> <li>1. L'Organizzatore annulla il processo di modifica</li> <li>2. L'Organizzatore viene reindirizzato alla visualizzazione evento</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di navigazione tra le diverse maschere, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Modifica partecipazione
<b>Descrizione</b>	Il Partecipante registrato ha la possibilità di modificare i dati da lui precedentemente inseriti, compresa la partecipazione
<b>Attori</b>	Partecipante registrato
<b>Relazioni</b>	Visualizzazione evento
<b>Precondizioni</b>	Il Partecipante registrato ha visualizzato l'evento
<b>Postcondizioni</b>	Ciò causa la modifica dei programmi. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Il Partecipante registrato ha la possibilità di modificare i dati personali inseriti durante l'accettazione dell'invito o la creazione, compresa la partecipazione o meno.</li> <li>2. Il Partecipante viene reindirizzato a Visualizzazione evento o alla pagina principale</li> </ol>
<b>Scenari alternativi</b>	<p>Scenario A</p> <ol style="list-style-type: none"> <li>1. Nel caso il Partecipante sia un organizzatore che abbandona l'evento, esso manterrà il titolo di organizzatore e la gestione dell'evento ma scomparirà dalla lista dei partecipanti</li> </ol> <p>Scenario B</p> <ol style="list-style-type: none"> <li>1. Nel caso in cui un Organizzatore non sia partecipante, ha la possibilità di essere reinserito tra i partecipanti venendo indirizzato a Accettazione invito.</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di inserimento dati, chiarezza e leggibilità, il nominativo deve rimanere univoco all'interno dell'evento
<b>Punti aperti</b>	

<b>Titolo</b>	Visualizza programma
<b>Descrizione</b>	Viene mostrato il programma attuale specifico per il gruppo del partecipante registrato che ha eseguito l'azione
<b>Attori</b>	Partecipante registrato
<b>Relazioni</b>	Visualizzazione evento
<b>Precondizioni</b>	Il partecipante registrato deve aver eseguito la richiesta di visualizzare il programma
<b>Postcondizioni</b>	Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Il Partecipante ha scelto di visualizzare il programma</li> <li>2. Il sistema recupera il programma relativo al gruppo di cui fa parte il Partecipante</li> <li>3. Il programma viene visualizzato</li> </ol>
<b>Scenari alternativi</b>	<p>scenario A</p> <ol style="list-style-type: none"> <li>1. Non ci sono abbastanza posti per trasportare il partecipante in questione, gli viene presentato un messaggio di avviso riguardante la sua condizione di Partecipante unico nel gruppo e la destinazione in una mappa</li> </ol> <p>Scenario B</p> <ol style="list-style-type: none"> <li>1. Il Partecipante è autonomo, gli viene presentata la destinazione in una mappa. (Fa comunque parte di un gruppo di un solo membro)</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di inserimento dati, chiarezza e leggibilità, il nominativo deve rimanere univoco all'interno dell'evento
<b>Punti aperti</b>	

<b>Titolo</b>	Visualizzazione informazioni evento
<b>Descrizione</b>	Vengono impaginate e mostrate le informazioni relative all'evento come i dati e la lista attuale dei partecipanti
<b>Attori</b>	Invitato, Partecipante registrato
<b>Relazioni</b>	Visualizzazione evento, Anteprima evento
<b>Precondizioni</b>	L'utente registrato è collegato all'evento come invitato o come partecipante
<b>Postcondizioni</b>	
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Vengono visualizzate le informazioni dell'evento.</li> </ol>
<b>Scenari alternativi</b>	<ol style="list-style-type: none"> <li>1. Se l'evento è stato cancellato viene visualizzato un messaggio di errore e l'attore viene reindirizzato alla pagina principale</li> </ol>
<b>Requisiti non funzionali</b>	Semplicità di navigazione tra le diverse maschere, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Accettazione Invito
<b>Descrizione</b>	L'invitato decide se accettare o meno l'invito
<b>Attori</b>	Invitato
<b>Relazioni</b>	Anteprima evento
<b>Precondizioni</b>	L'utente registrato ha visualizzato le informazioni dell'evento e ha deciso di procedere
<b>Postcondizioni</b>	Ciò causa la modifica dei programmi. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. L'invitato decide di accettare l'invito</li> <li>2. Gli viene presentata una maschera per l'inserimento dei dati in cui inserire:             <ol style="list-style-type: none"> <li>2.1. Il nominativo (univoco all'interno dell'evento.)</li> <li>2.2. La partecipazione alla carpool o l'autonomia                     <ol style="list-style-type: none"> <li>2.2.1 all'andata</li> <li>2.2.2 al ritorno</li> <li>2.2.3 recapito</li> </ol> </li> <li>2.3. In ognuno dei casi carpool bisognerà inserire                     <ol style="list-style-type: none"> <li>2.3.1 indirizzo di partenza o ritorno</li> <li>2.3.2 disponibilità veicolo e posti disponibili sul veicolo</li> </ol> </li> </ol> </li> <li>3. L'Invitato viene reindirizzato alla pagina principale</li> </ol>
<b>Scenari alternativi</b>	<p>Scenario A</p> <ol style="list-style-type: none"> <li>1. L'invitato decide di annullare l'operazione</li> <li>2. L'invitato viene reindirizzato alla pagina principale</li> </ol> <p>Scenario B</p> <p>2 -1 (scenario principale) Se il nominativo inserito è già presente, verrà richiesto all'Invitato di sceglierne un altro.</p>
<b>Requisiti non funzionali</b>	Semplicità di inserimento dati, chiarezza e leggibilità
<b>Punti aperti</b>	

<b>Titolo</b>	Modifica dei programmi
<b>Descrizione</b>	Vengono modificati o generati nuovi programmi
<b>Attori</b>	Evento: modifica di un evento o dei dati di un partecipante
<b>Relazioni</b>	
<b>Precondizioni</b>	L'evento esiste
<b>Postcondizioni</b>	I Programmi sono aggiornati. Il log viene aggiornato
<b>Scenario principale</b>	<ol style="list-style-type: none"> <li>1. Dopo una qualsiasi modifica ai dati di un evento, si ri-esegue l'algoritmo per dividere i partecipanti in gruppi in base al numero di posti liberi in ogni veicolo e il calcolo della distanza tra il guidatore e i passeggeri.</li> <li>2. Vengono caricati i programmi aggiornati e resi disponibili per i partecipanti</li> </ol>
<b>Scenari alternativi</b>	
<b>Requisiti non funzionali</b>	Velocità di calcolo e ottimizzazione dell'algoritmo
<b>Punti aperti</b>	

# Analisi del Rischio

## Tabella di valutazione dei beni

Bene	Valore	Esposizione
Sistema Informativo	Alto. Supporto a tutta la gestione dell'applicazione	Media, perdita dati degli eventi, con inclusi dati degli utenti. Perdita di immagine e rare conseguenze legali
Informazioni relative agli eventi	Alto. Tutta l'applicazione si basa su questi dati	Media. Diffusione di dati degli eventi e rare conseguenze legali
Informazioni relative agli utenti	Alto. Se si perdono i dati degli utenti o se vengono rubati gli utenti non possono più eseguire l'accesso	Media. Diffusione di dati degli utenti e rare conseguenze legali

## Tabella Minacce/Controlli

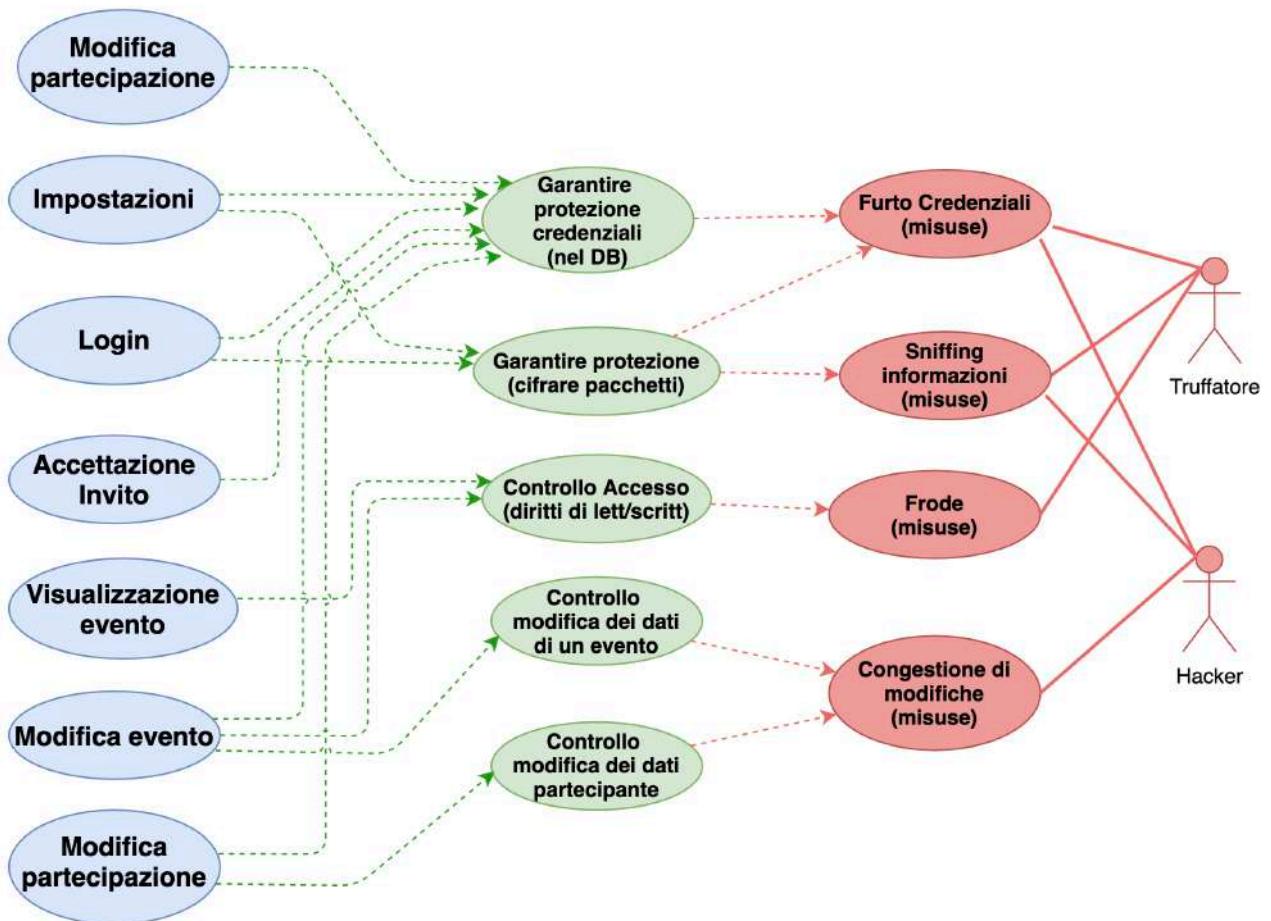
Minaccia	Probabilità	Controllo	Fattibilità
Furto credenziali amministratore del sistema informativo	Media	Accesso consentito solo agli sviluppatori	Basso costo, basta non divulgare le password e non informare nessuno della loro esistenza
		Log di sistema	Basso costo implementativo
Furto credenziali di un qualsiasi utente dal database che potrebbe agire da organizzatore	Media	I dati sono conservati in modo sicuro nel database	Medio costo implementativo
Divulgazione involontaria da parte di un utente delle sue credenziali	Alta	Possibilità di essere connessi soltanto da un dispositivo alla volta	Basso costo implementativo
Intercettazione comunicazioni	Alta. Il sistema è distribuito e avvengono tantissime interazioni tra i diversi enti che comunicano	Utilizzo di protocolli adatti allo scambio sicuro di informazioni	Basso costo implementativo
		Log delle operazioni	Basso costo implementativo
DoS	Bassa	Progettazione studiata in modo da controllare il numero di richieste in arrivo in un lasso di tempo. Eventualmente tempo di cooldown.	Medio costo implementativo

Minaccia	Probabilità	Controllo	Fattibilità
Overbooking di modifiche ai dati di un partecipante e incapacità di calcolare i programmi in maniera responsive	Media	Mentre i programmi vengono rielaborati non possono avvenire modifiche	Medio costo implementativo
		Possibilità di inserire un limite di modifiche per ora che previene lo spreco di risorse	

## Analisi Tecnologica della Sicurezza

Tecnologia	Vulnerabilità
Autenticazione username/password	<ul style="list-style-type: none"> <li>- Password banali</li> <li>- Utente rivela volontariamente password</li> <li>- Utente rivela la password a seguito di un attacco di ingegneria sociale</li> </ul>
Cifratura comunicazioni	<p>Le vulnerabilità dipendono dal tipo di cifratura.</p> <p>Simmetrica</p> <ul style="list-style-type: none"> <li>- Tempo di vita della chiave. Più informazioni vengono cifrate con la stessa chiave, più materiale è offerto per l'analisi del testo di un attaccante</li> <li>- Lunghezza della chiave</li> <li>- Memorizzazione della chiave</li> </ul> <p>Asimmetrica</p> <ul style="list-style-type: none"> <li>- Memorizzazione chiave privata</li> </ul>
Architettura Client/Server	<ul style="list-style-type: none"> <li>- Dos</li> <li>- Man in the middle</li> <li>- Sniffing delle comunicazioni</li> </ul>
Ricerca delle strade	Crash dell'applicazione esterna usata per calcolare i percorsi
Recovery email	- il server che gestisce le mail di recovery ha problemi

# Security Use Case e Misuse Case



## Security Use Case & Misuse Case Scenari

<b>Titolo</b>	Garantire protezione (cifrare pacchetti)	
<b>Descrizione</b>	I pacchetti inviati devono essere cifrati	
<b>Misuse case</b>	Sniffing Informazioni, furto credenziali	
<b>Relazioni</b>		
<b>Precondizioni</b>	L'attaccante è conosce il modo di intercettare i pacchetti	
<b>Postcondizioni</b>		
<b>Scenario principale</b>	<b>Sistema</b>	<b>Attaccante</b>
	Il sistema si occupa di proteggere i messaggi che fluiscono tra le diverse parti e memorizza nel log i messaggi	
		Intercettazione di un messaggio nel sistema
		Non riesce a rimuovere il meccanismo di protezione, così genera un nuovo messaggio e lo prova a mandare al destinatario
<b>Scenari di attacco avvenuto con successo</b>	Il sistema si accorge che il messaggio ricevuto non è pertinente e segnala un tentativo di frode	
	<b>Sistema</b>	<b>Attaccante</b>
	Il sistema si occupa di proteggere i messaggi che fluiscono tra le diverse parti e memorizza nel log i messaggi	
		Intercettazione di un messaggio nel sistema
		Rimozione del meccanismo di protezione. Modifica del messaggio, ri-applicazione della protezione e invio del messaggio al destinatario
	Il sistema elabora il messaggio e agisce di conseguenza, scrive il messaggio nel log	
	Un'analisi successiva del log riporterà questo dato. Se rileva una discrepanza tra pacchetto inviato e ricevuto segnala un tentativo di frode e cerca di riportare il sistema ad uno stato precedente	

<b>Titolo</b>	Garantire protezione credenziali (nel DB)	
<b>Descrizione</b>	Le credenziali devono essere protette	
<b>Misuse case</b>	Furto credenziali	
<b>Relazioni</b>		
<b>Precondizioni</b>	L'attaccante è riuscito ad entrare nel database	
<b>Postcondizioni</b>	Le credenziali sono trapelate ma le password sono comunque protette da cifratura	
<b>Scenario principale</b>	<b>Sistema</b>	<b>Attaccante</b>
		L'attaccante riesce ad entrare nel database e rubare username e password degli utenti
	Tutte le password sono protette da una cifratura hash, è altamente improbabile scoprire le password	
<b>Scenari di attacco avvenuto con successo</b>	<b>Sistema</b>	<b>Attaccante</b>
		L'attaccante riesce ad entrare nel database e rubare username e password degli utenti
		L'attaccante riesce a decifrare l'hash e scoprire la password tramite bruteforcing

<b>Titolo</b>	Controllo modifica dei dati partecipante	
<b>Descrizione</b>	È necessario che i dati di un partecipante non possano essere modificati in maniera eccessivamente frequente, per non sovraccaricare di richieste il server che fornisce il servizio di navigazione.	
<b>Misuse case</b>	Congestione di modifiche	
<b>Relazioni</b>		
<b>Precondizioni</b>	Un partecipante tenta ripetutamente di modificare i suoi dati di partecipazione	
<b>Postcondizioni</b>	La possibilità di modificare i dati di partecipazione viene bloccata per un certo periodo di tempo.	
<b>Scenario principale</b>	<b>Sistema</b>	<b>Attaccante</b>
		Modifica la partecipazione ripetutamente e per molte volte al secondo
	Il sistema si accorge dell'anomalia perché durante il tempo di cooldown riceve molte altre richieste	
	Viene negata la possibilità di ulteriori modifiche per un certo periodo di tempo	
<b>Scenari di attacco avvenuto con successo</b>	<b>Sistema</b>	<b>Attaccante</b>
		Modifica la partecipazione ripetutamente e per molte volte al secondo
	Il sistema non si accorge dell'anomalia perché l'attaccante evidentemente ha aggirato sistemi di sicurezza più avanzati. Si entra in un altro caso d'uso	

<b>Titolo</b>	Controllo Accesso (diritti lettura/scrittura)	
<b>Descrizione</b>	Gli accessi al sistema devono controllati nel senso che soltanto i partecipanti registrati ad un evento possono visualizzarlo e soltanto l'organizzatore può modificare un evento da lui creato	
<b>Misuse case</b>	Frode	
<b>Relazioni</b>		
<b>Precondizioni</b>	Il truffatore è in grado di visualizzare un evento a cui non partecipa o è in grado di modificare i dati di un evento di cui non è organizzatore	
<b>Postcondizioni</b>	Il sistema si accorge del tentativo e lancia un avvertimento di uso incorretto	
<b>Scenario principale</b>	<b>Sistema</b>	<b>Attaccante</b>
		L'attaccante conosce l'esistenza di un evento e prova a visualizzarlo anche se non è nella lista dei partecipanti.
	Il sistema opera un controllo sull'username dell'utente cercandolo nella lista dei partecipanti associato ad un nominativo. Se non lo trova comunica l'errore.	
<b>Scenari di attacco avvenuto con successo</b>	<b>Sistema</b>	<b>Attaccante</b>
		<b>1</b>
		L'attaccante riesce aggiungersi alla lista partecipanti facendo bruteforce degli inviti
	Il sistema opera un controllo sull'username ed esso va a buon fine	
		Naviga tra le maschere del sistema e cerca di carpire più informazioni possibili. Infine abbandona l'evento.
	Il log registra ogni volta che un utente si aggiunge con un invito e ogni volta che la lista dei partecipanti viene aggiornata, così come quando viene abbandonato un evento. Questo permetterà di intuire la situazione accorgendosi di un'aggiunta di partecipazione e un abbandono di un evento in un breve lasso di tempo. Oppure l'organizzatore stesso ha la possibilità di eliminare la partecipazione di un utente sospetto	
		<b>2</b>
		L'attaccante riesce a violare il database e a cambiare se stesso in organizzatore. Prova poi a modificare l'evento
	Il sistema opera un controllo sull'username ed esso va a buon fine	
		Naviga tra le maschere per la modifica dell'evento e lo riesce a modificare.
	Il log registra tutti i cambiamenti. Quando un amministratore si accorge che l'utente che ha modificato l'evento non è colui che lo ha originariamente creato, provvederà a sistemare.	

<b>Titolo</b>	Controllo modifica dei dati di un evento	
<b>Descrizione</b>	È necessario che l'evento venga modificato solo dall'organizzatore e che i nuovi dati siano accettabili	
<b>Misuse case</b>	Congestione di modifiche	
<b>Relazioni</b>		
<b>Precondizioni</b>	Un organizzatore tenta ripetutamente di modificare un evento	
<b>Postcondizioni</b>	La possibilità di modificare l'evento viene bloccata per un certo periodo di tempo.	
<b>Scenario principale</b>	<b>Sistema</b>	<b>Attaccante</b>
		Modifica l'evento ripetutamente e per molte volte al secondo
	Il sistema si accorge dell'anomalia perché durante il tempo di cooldown riceve molte altre richieste	
	Viene negata la possibilità di ulteriori modifiche per un certo periodo di tempo	
<b>Scenari di attacco avvenuto con successo</b>	<b>Sistema</b>	<b>Attaccante</b>
		Modifica l'evento ripetutamente e per molte volte al secondo
	Il sistema non si accorge dell'anomalia perché l'attaccante evidentemente ha aggirato sistemi di sicurezza più avanzati. Si entra in un altro caso d'uso	

# Analisi del Problema

## Analisi Documento dei Requisiti: Analisi delle Funzionalità

**Tabella Funzionalità**

Funzionalità	Tipo	Grado di complessità	Requisiti collegati
Registrazione	Memorizzazione dati, Interazione esterno	Semplice	1F
Recovery_delle_credenziali	Interazione esterno, Gestione dati	Complessa	16F
Creazione_Evento	Memorizzazione dati, Gestione dati	Semplice	2F, 4F, 20F
Login	Interazione esterno, Gestione dati	Semplice	24F
Impostazioni	Gestione dati, Memorizzazione dati	Semplice	21F
Visualizzazione_lista_eventi	Gestione dati	Semplice	25F
Inserimento_invito	Gestione dati	Semplice	12F
Anteprima_evento	Gestione dati	Semplice	26F, 5F
Accettazione_invito	Memorizzazione dati	Semplice	6F, 6.1F, 6.2F, 18.2F, 19F
Modifica_evento	Memorizzazione dati, Gestione dati	Semplice	3F, 17F, 17.1F/18.1F, 18.2F
Visualizzazione_evento	Gestione dati	Semplice	27F
Visualizzazione_programma	Gestione dati	Semplice	5.1F, 10F, 11F
Modifica_partecipazione	Memorizzazione dati, Gestione dati	Semplice	7F, 18F
Modifica_dei_programmi	Interazione esterno, Manipolazione dati	Complesso	8F, 9F, 11F
Visualizzazione_informazioni_evento	Gestione dati	Semplice	5F
Scrittura_del_log	Memorizzazione dati	Semplice	22F
Anti-dos	Manipolazione dati	Complesso	23F

**NB:** interazione con l'esterno in questa tabella è intesa come un'interazione di un qualcosa di esterno all'applicazione con l'applicazione. Es: utente che esegue il login o che si registra, oppure un'interazione con una tecnologia esterna. Una volta autenticato, l'utente è considerato interno all'applicazione.

## Registrazione: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Username (mail)	Semplice	Protezione media	Input	Deve essere una mail valida
Password	Semplice	Protezione molto alta	Input	Tra 8 e 16 caratteri

## Recovery delle credenziali: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Username (mail)	Semplice	Protezione media	Input	Deve essere una mail valida
Nuova password	Semplice	Protezione molto alta	Input	Tra 8 e 16 caratteri

## Creazione\_Evento: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Nome Evento	Semplice	Protezione media	Input	Massimo 100 caratteri
Descrizione	Semplice	Protezione media	Input	Massimo 2000 caratteri
Username organizzatore	Semplice	Protezione media	Input	Inserito automaticamente
Inserimento dati organizzatore	Composto	Protezione alta	Input	
Luogo	Composto	Protezione media	Input	
Data inizio	Semplice	Protezione media	Input	Deve essere futura rispetto alla data di creazione
Ora inizio	Semplice	Protezione media	Input	
Data fine	Semplice	Protezione media	Input	Deve essere successiva alla data di inizio
Ora fine	Semplice	Protezione media	Input	In caso di "data inizio" e "data fine" coincidenti, deve essere maggiore di "Ora inizio"
Massimo partecipanti	Semplice	Protezione media	Input	Numero di default modificabile.
Identificativo evento	Semplice	Protezione media	Output	
Invito	Semplice	Protezione alta	Output	Generato in maniera univoca

## Login: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Username (mail)	Semplice	Protezione media	Input	
Password	Semplice	Protezione molto alta	Input	Tra 8 e 16 caratteri

## Impostazioni: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Indirizzo default	Semplice	Protezione alta	Input	
Recapito default	Semplice	Protezione alta	Input	
Nominativo default	Semplice	Protezione bassa	Input	Massimo 64 caratteri
Nuovo username	Semplice	Protezione media	Input	Deve essere una mail valida

## Visualizzazione\_liste\_eventi: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Nomi eventi	Semplice	Protezione media	Output	

## Inserimento\_invito: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Invito	Semplice	Protezione media	Input	

## Anteprima\_evento: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Dati evento	Composto	Protezione media	Output	

## Visualizzazione\_evento: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Dati evento	Composto	Protezione media	Output	

## Accettazione\_invito: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Nominativo	Semplice	Protezione bassa	Input	Massimo 64 caratteri
Recapito	Semplice	Protezione Alta	Input	Adesione carpool
Partecipazione carpool <b>andata</b>	Semplice	Protezione bassa	Input	Booleano
Luogo di partenza	Composto	Protezione Alta	Input	Adesione carpool andata
Disponibilità veicolo andata	Semplice	Protezione bassa	Input	Adesione carpool andata Booleano
Posti disponibili andata	Semplice	Protezione bassa	Input	Disponibilità veicolo andata
Partecipazione carpool <b>ritorno</b>	Semplice	Protezione bassa	Input	Booleano
Luogo di ritorno	Composto	Protezione Alta	Input	Adesione carpool ritorno
Disponibilità veicolo ritorno	Semplice	Protezione bassa	Input	Adesione carpool ritorno Booleano
Posti disponibili ritorno	Semplice	Protezione bassa	Input	Disponibilità veicolo ritorno

## Visualizzazione\_programma: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Destinazione	Composto	Protezione bassa	Output	
Nominativi gruppo	Composto	Protezione media	Output	Partecipazione carpool
Recapiti gruppo	Composto	Protezione molto alta	Output	Partecipazione carpool
Percorso previsto	Composto	Protezione molto alta	Output	Partecipazione carpool
Orari previsti	Composto	Protezione bassa	Output	Partecipazione carpool

## Modifica\_evento: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Nome Evento	Semplice	Protezione media	Input	Massimo 100 caratteri
Descrizione	Semplice	Protezione media	Input	Massimo 2000 caratteri
Eliminazione partecipante	Composto	Protezione media	Input	
Luogo	Composto	Protezione media	Input	
Data inizio	Semplice	Protezione media	Input	Deve essere futura rispetto alla data di creazione
Ora inizio	Semplice	Protezione media	Input	Deve essere futura rispetto all'ora di creazione
Data fine	Semplice	Protezione media	Input	Deve essere successiva alla data di inizio
Ora fine	Semplice	Protezione media	Input	In caso di "data inizio" e "data fine" coincidenti, deve essere maggiore di "Ora inizio"
Massimo partecipanti	Semplice	Protezione media	Input	Numero default, modificabile

## Modifica\_dei\_programmi: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Programmi aggiornati	Complesso	Protezione media	Output	

## Visualizzazione\_informazioni\_evento: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Dati essenziali dell'evento	Complesso	Protezione bassa	Output	

## Modifica\_partecipazione: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Nominativo	Semplice	Protezione bassa	Input	Massimo 64 caratteri
Recapito	Semplice	Protezione Alta	Input	Adesione carpool
Partecipazione carpool <b>andata</b>	Semplice	Protezione bassa	Input	Booleano
Luogo di partenza	Composto	Protezione Alta	Input	Adesione carpool andata
Disponibilità veicolo andata	Semplice	Protezione bassa	Input	Adesione carpool andata Booleano
Posti disponibili andata	Semplice	Protezione bassa	Input	Disponibilità veicolo andata
Partecipazione carpool <b>ritorno</b>	Semplice	Protezione bassa	Input	Booleano
Luogo di ritorno	Composto	Protezione Alta	Input	Adesione carpool ritorno
Disponibilità veicolo ritorno	Semplice	Protezione bassa	Input	Adesione carpool ritorno Booleano
Posti disponibili ritorno	Semplice	Protezione bassa	Input	Disponibilità veicolo ritorno

## Scrittura\_del\_log: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / output	Vincoli
Data	Semplice	Protezione media	Input	
Ora	Semplice	Protezione media	Input	
Messaggio	Composto	Protezione molto alta	Input	
File di Log aggiornato	Semplice	Protezione media	Output	

# Analisi Documento dei Requisiti: Analisi dei Vincoli

**Tabella Vincoli**

Requisito	Categorie	Impatto	Funzionalità
Un evento ha uno e un solo organizzatore.	Protezione dei dati	Semplicità di progetto	Creazione_evento, Modifica_evento
Non si può superare un certo numero di posti nei veicoli.	Scalabilità, Performance	Pone un tetto al numero di membri di un gruppo. Migliora i tempi di risposta e la quantità di calcoli da eseguire	Accettazione_invito, Modifica_partecipazione, Modifica_programmi
Semplicità di navigazione tra le diverse maschere	Usabilità	Cercare di migliorare	Tutte le funzionalità
Il sistema sarà distribuito. Per il front-end sarà utilizzato su un dispositivo in grado di memorizzare i dati relativi ai programmi, almeno temporaneamente in una cache	Tempo di risposta, Performance	Cercare di migliorare	Visualizzazione_programma, Visualizzazione_informazioni_evento
Sarà necessario predisporre un database che manterrà i dati degli utenti registrati in modo persistente, e che manterrà soltanto fino alla fine di ogni evento le informazioni relative ad esso	Protezione dei dati, Sicurezza	È fondamentale per la realizzazione	Tutte le funzionalità
Un evento DEVE avere un numero massimo di partecipanti (ad esempio di default 50 e nel caso l'organizzatore prevede che sarà un evento molto grande potrà aumentarlo fino a 200)	Performance, Scalabilità	Pone un tetto al numero di partecipanti a un evento. Migliora i tempi di risposta e la quantità di calcoli da eseguire	Accettazione invito, Modifica Programmi, Modifica Partecipazione,
Il database dove sono conservati i dati deve essere direttamente accessibile solo da un amministratore.	Protezione dei dati, Sicurezza	Aumenta la sicurezza e la privacy dei dati	
L'applicazione interroga il database in maniera sicura e controllata	Protezione dei dati, Sicurezza	Aumenta la sicurezza e la privacy dei dati	Tutte le funzionalità
Bisognerà utilizzare Google Maps se possibile	Tempo di risposta, Performance, Scalabilità, Usabilità	È fondamentale per il calcolo dei gruppi e dei percorsi	Modifica dei programmi
Controllo Accesso	Sicurezza	Peggiorano il tempo di risposta, migliorano la privacy dei dati e la sicurezza	Modifica partecipazione, Modifica evento, Visualizzazione evento

Requisito	Categorie	Impatto	Funzionalità
Garantire protezione credenziali (nel DB)	Sicurezza, protezione dei dati	Peggiorano il tempo di risposta, migliorano la privacy dei dati e la sicurezza	
Garantire protezione (cifrare pacchetti)	Sicurezza	Peggiorano il tempo di risposta, migliorano la privacy dei dati e la sicurezza	Tutte le funzionalità che richiedono una comunicazione con il server
Controllo modifica dei dati di un evento	Sicurezza	Peggiorano il tempo di risposta, migliorano la privacy dei dati e la sicurezza	Modifica evento
Controllo modifica dei dati partecipante	Sicurezza	Peggiorano il tempo di risposta, migliorano la privacy dei dati e la sicurezza	Modifica partecipazione

# Analisi Documento dei Requisiti: Analisi delle Interazioni

## Tabella Maschere

Maschere	Informazioni	Funzionalità
View Registrazione	Mail (username), password	Registrazione
View Recovery	Nuova password, Mail	Recovery_credenziali
View Login	Username, Password	Login
Home Utente Registrato	Lista di eventi	Visualizzazione Lista Eventi
Home Evento Singolo	Organizzatore, data inizio, ora inizio, data fine, ora fine, Luogo, nome e descrizione, numero massimo partecipanti	Visualizzazione_Evento, Visualizzazione_Informazioni_evento
View Lista Partecipanti	Lista dei partecipanti	Visualizzazione_Informazioni_evento
View Programma	Nominativi dei passeggeri, orari di partenza, itinerario	Visualizzazione_Programma, Visualizzazione_Informazioni_evento
View Modifica Partecipazione	Vecchi dati (Nominativo, partecipazione carpool andata, partecipazione carpool ritorno, recapito, Luogo di partenza, Luogo di ritorno, disponibilità veicolo, posti disponibili veicolo) Nuovi dati (Nominativo, partecipazione carpool andata, partecipazione carpool ritorno, recapito, Luogo di partenza, Luogo di ritorno, disponibilità veicolo, posti disponibili veicolo)	Modifica_patecipazione
View Creazione Evento	data inizio, ora inizio, data fine, ora fine, Luogo, nome e descrizione, numero massimo partecipanti	Creazione_evento
View Inserimento Dati Partecipazione	Nominativo, partecipazione carpool andata, partecipazione carpool ritorno, recapito, Luogo di partenza, Luogo di ritorno, disponibilità veicolo, posti disponibili veicolo	Accettazione_Invito
View Modifica Evento	Vecchi dati ( data inizio, ora inizio, data fine, ora fine, Luogo, numero massimo partecipanti, nome e descrizione) Nuovi dati (data inizio, ora inizio, data fine, ora fine, Luogo, numero massimo partecipanti, nome e descrizione)	Modifica_evento
View Inserimento Invito	Invito	Inserimento_Invito
View Anteprima Evento	Organizzatore, data inizio, ora inizio, data fine, ora fine, indirizzo, nome e descrizione, numero massimo partecipanti, lista dei nominativi dei partecipanti	Anteprima_Evento
Home Impostazioni	Cambio username, modifica dati di default	Impostazioni
View Cambio Username o Dati Default	Dati vecchi (username, Luogo default, recapito default, nominativo default) Dati Nuovi (username, Luogo default, recapito default, nominativo default)	Impostazioni

**Tabella Sistemi esterni**

Sistema	Descrizione	Protocollo di Interazione	Livello di Sicurezza
API Places API Distance Matrix (Google Maps)	Sistema che si occupa del calcolo dei tempi di percorrenza tra i vari indirizzi	Si chiede a Google Maps attraverso a delle API che convertono gli indirizzi in delle variabili che il server può capire. con un'altra API si ottiene la distanza tra due indirizzi.	Molto Alto
API Routes API Directions (Google Maps)	Sistema che si occupa del calcolo degli itinerari una volta forniti gli indirizzi e le tappe	Si chiede a Google Maps attraverso una API di restituire le informazioni relative ad un itinerario composto da partenza, tappe intermedie e destinazione	Molto Alto
Database	Sistema che si occupa dell'archiviazione dei dati degli eventi e degli utenti	Si interrogherà il server per ottenere i dati adatti ad ogni maschera	Alto, ci sono dati personali. Nel caso si scelga un gestore esterno la sicurezza sarà affidata completamente a lui

# Analisi Ruoli e Responsabilità

## Tabella Ruoli

Ruolo	Responsabilità	Maschere	Riservatezza	Numerosità
Utente non Registrato	Nessuna	Registrazione, Login, Recovery	Molto alta	Indefinita
Utente Registrato	È responsabile delle sue credenziali	Home Utente Registrato, Home Impostazioni, View Cambio Username o Dati Default	Media	Indefinita
Organizzatore	È responsabile delle informazioni relative all'evento.	View Creazione Evento, View Modifica Evento, View Inserimento Dati Partecipazione	Media	Indefinita
Partecipante	È responsabile delle sue informazioni specifiche relative ad ogni evento.	Home Evento Singolo, View Lista Partecipanti, View Programma, View Modifica Partecipazione, View Inserimento Dati Partecipazione	Media	Definita per ogni evento
Invitato	Non ha responsabilità aggiuntive rispetto ad un utente Registrato	View Inserimento Invito, View Anteprima Evento	Media	Indefinita

## Utente non Registrato : Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Username	Scrittura
Password	Scrittura

## Utente Registrato : Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Username	Lettura/Scrittura
Password	Scrittura
Indirizzo default	Lettura/Scrittura
Recapito default	Lettura/Scrittura
Nominativo default	Lettura/Scrittura
Lista di eventi	Lettura

## Organizzatore : Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Luogo	Lettura/Scrittura
Data inizio	Lettura/Scrittura
Ora inizio	Lettura/Scrittura
Data fine	Lettura/Scrittura
Ora fine	Lettura/Scrittura
Nome Evento	Lettura/Scrittura
Descrizione evento	Lettura/Scrittura
Numero massimo partecipanti	Lettura/Scrittura
Lista dei partecipanti	Lettura/Scrittura
Programma	Lettura
Dati dei partecipanti relativi all'evento	Lettura
Nominativo	Lettura/Scrittura
Invito	Lettura
Recapito	Lettura/Scrittura
Partecipazione carpool andata	Lettura/Scrittura
Luogo di partenza	Lettura/Scrittura
Disponibilità veicolo andata	Lettura/Scrittura
Posti disponibili sul veicolo andata	Lettura/Scrittura
Partecipazione carpool ritorno	Lettura/Scrittura
Luogo di ritorno	Lettura/Scrittura
Disponibilità veicolo ritorno	Lettura/Scrittura
Posti disponibili sul veicolo ritorno	Lettura/Scrittura

## Partecipante : Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Luogo	Lettura
Data inizio	Lettura
Ora inizio	Lettura
Data fine	Lettura
Ora fine	Lettura
Nome Evento	Lettura
Descrizione evento	Lettura
Numero massimo partecipanti	Lettura
Lista dei partecipanti	Lettura
Programma	Lettura
Nominativo	Lettura/Scrittura
Recapito	Lettura/Scrittura
Partecipazione carpool andata	Lettura/Scrittura
Luogo di partenza	Lettura/Scrittura
Disponibilità veicolo andata	Lettura/Scrittura
Posti disponibili sul veicolo andata	Lettura/Scrittura
Partecipazione carpool ritorno	Lettura/Scrittura
Luogo di ritorno	Lettura/Scrittura
Disponibilità veicolo ritorno	Lettura/Scrittura
Posti disponibili sul veicolo ritorno	Lettura/Scrittura

## Invitato : Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Invito	Scrittura
Luogo	Lettura
Data inizio	Lettura
Ora inizio	Lettura
Data fine	Lettura
Ora fine	Lettura
Nome Evento	Lettura
Descrizione evento	Lettura
Numero massimo partecipanti	Lettura
Luogo	Lettura
Lista dei nominativi dei partecipanti	Lettura

## Scomposizione del Problema

### Tabella Scomposizione Funzionalità

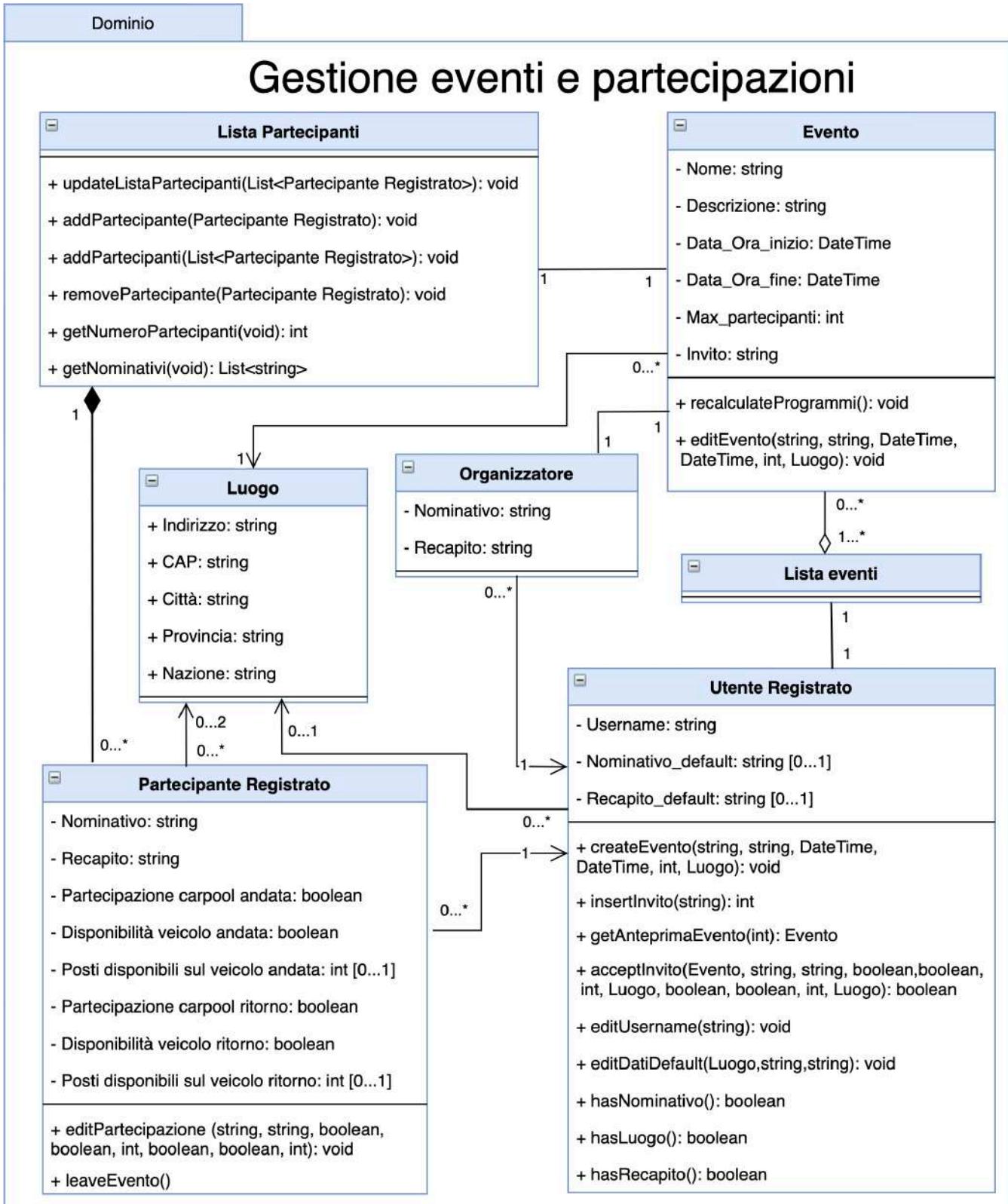
Funzionalità	Scomposizione
Impostazioni	Cambio Username Modifica dati di default

### Impostazioni: Tabella Sotto-Funzionalità

Sotto-Funzionalità	Sotto-Funzionalità	Legame	Informazioni
Cambio Username	Registrazione	Bisogna autenticarsi prima di poter cambiare il proprio username.	Username (mail)
Modifica dati di default	Registrazione	Bisogna autenticarsi prima di poter cambiare i propri dati.	Luogo default, Recapito default, NominativoDefault,

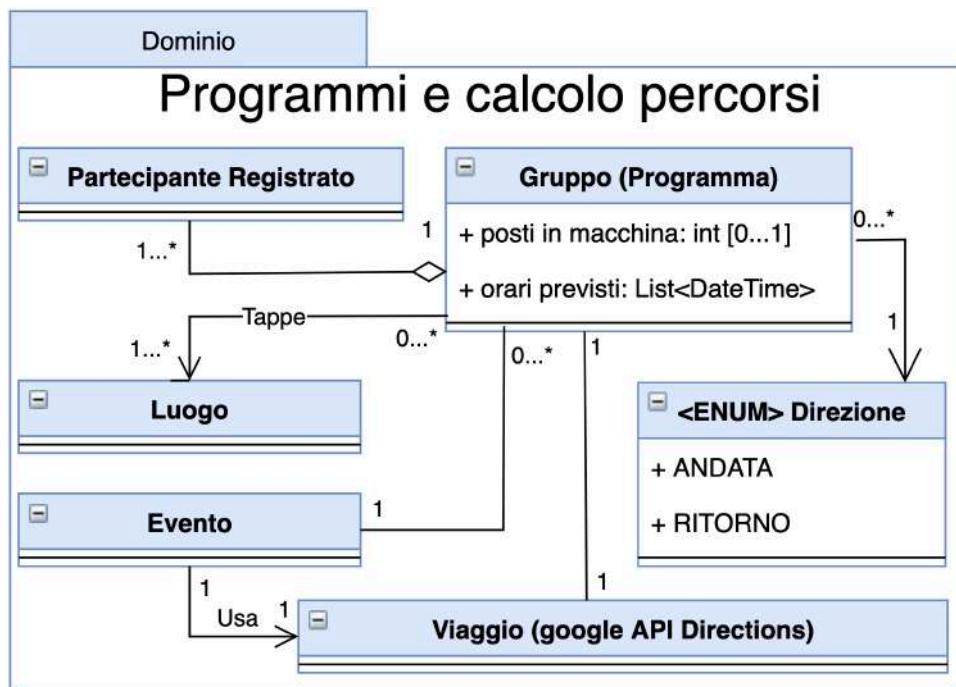
## Creazione Modello del Dominio

Il seguente diagramma delle classi rappresenta la porzione di dominio relativa a Eventi, Utenti registrati e Partecipazioni.

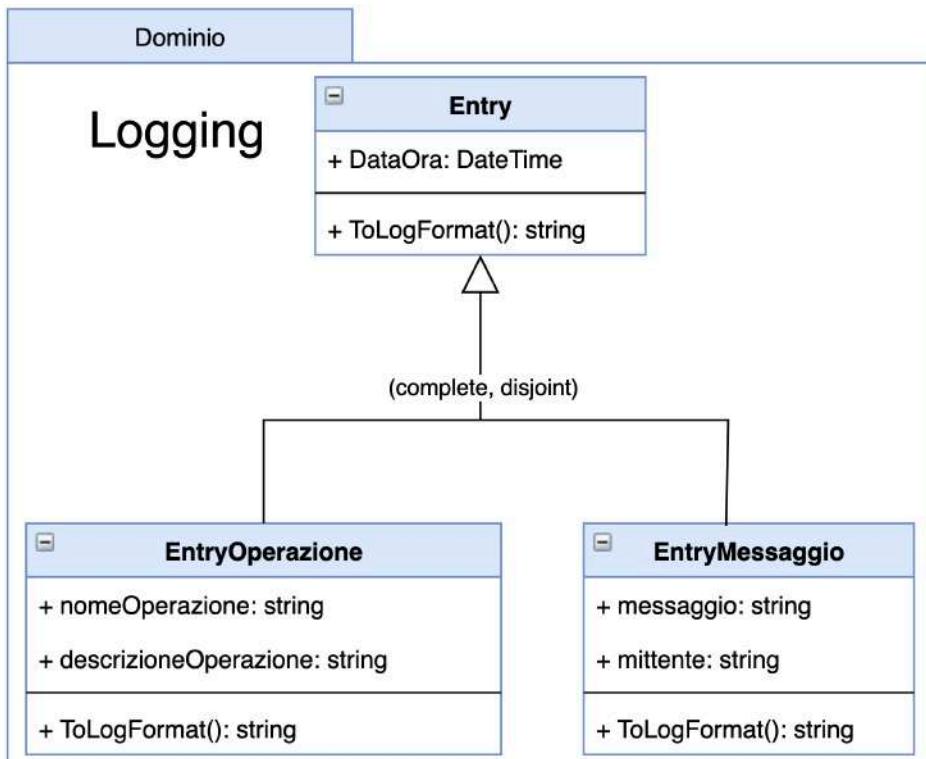


La funzione `recalculateProgrammi()` ha il compito di eseguire l'algoritmo del calcolo dei programmi al fine di generare gruppi sempre ottimizzati anche a seguito di eventuali cambiamenti.

Il seguente diagramma delle classi rappresenta la porzione di dominio che gestisce i gruppi.



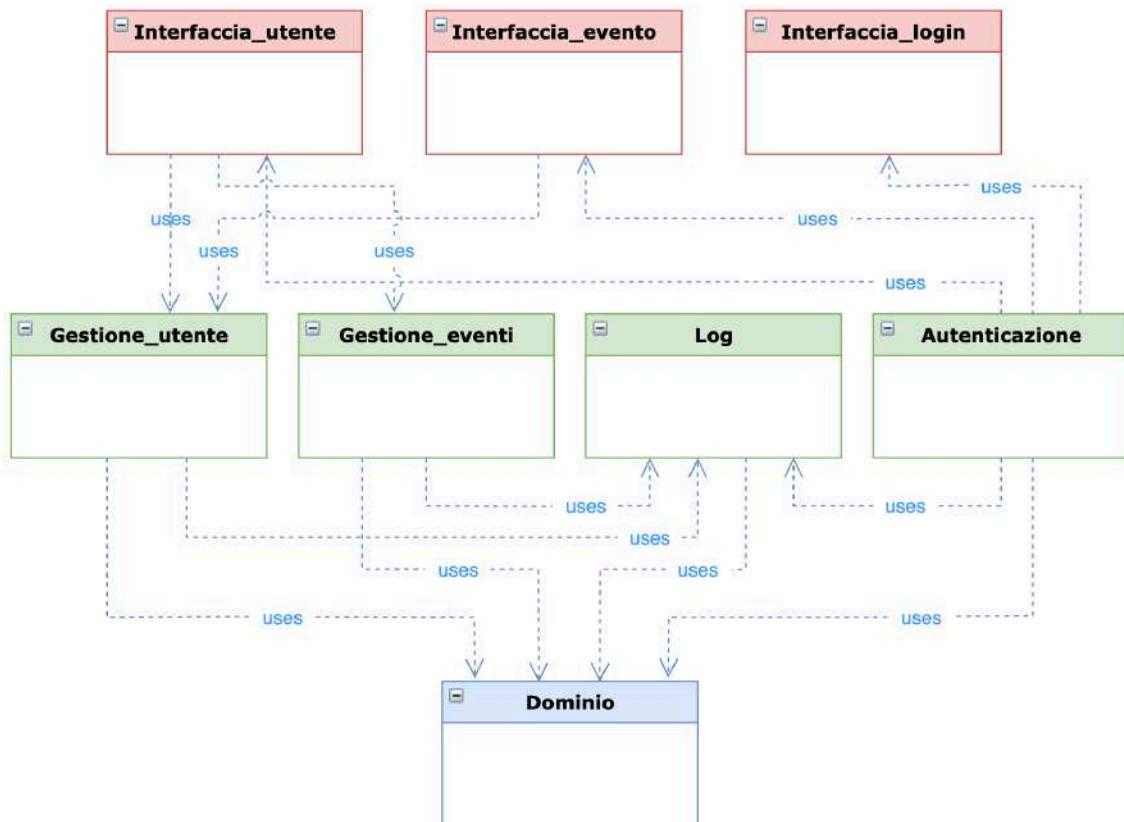
Il seguente diagramma delle classi rappresenta la porzione di dominio relativa alla gestione dei Log.



La funzione **ToLogFormat()** provvede a generare una stringa esplicativa salvabile sul file di Log.

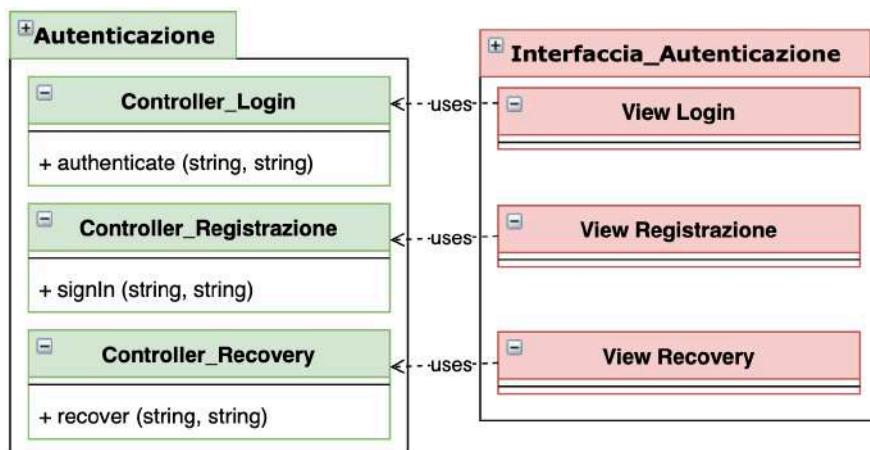
# Architettura Logica: Struttura

## Diagramma dei package



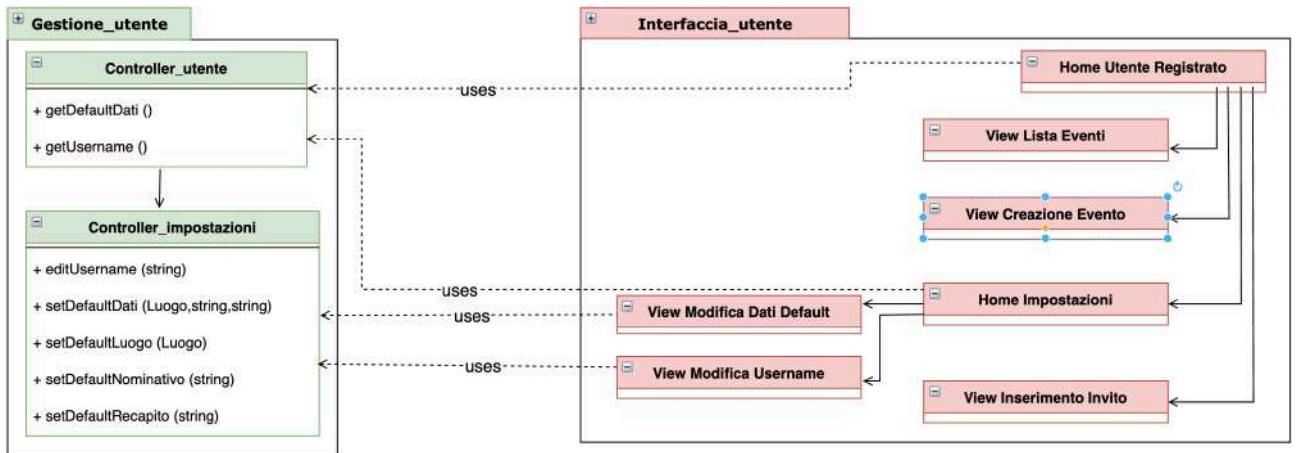
## Diagramma delle classi

### Diagramma delle classi: Autenticazione



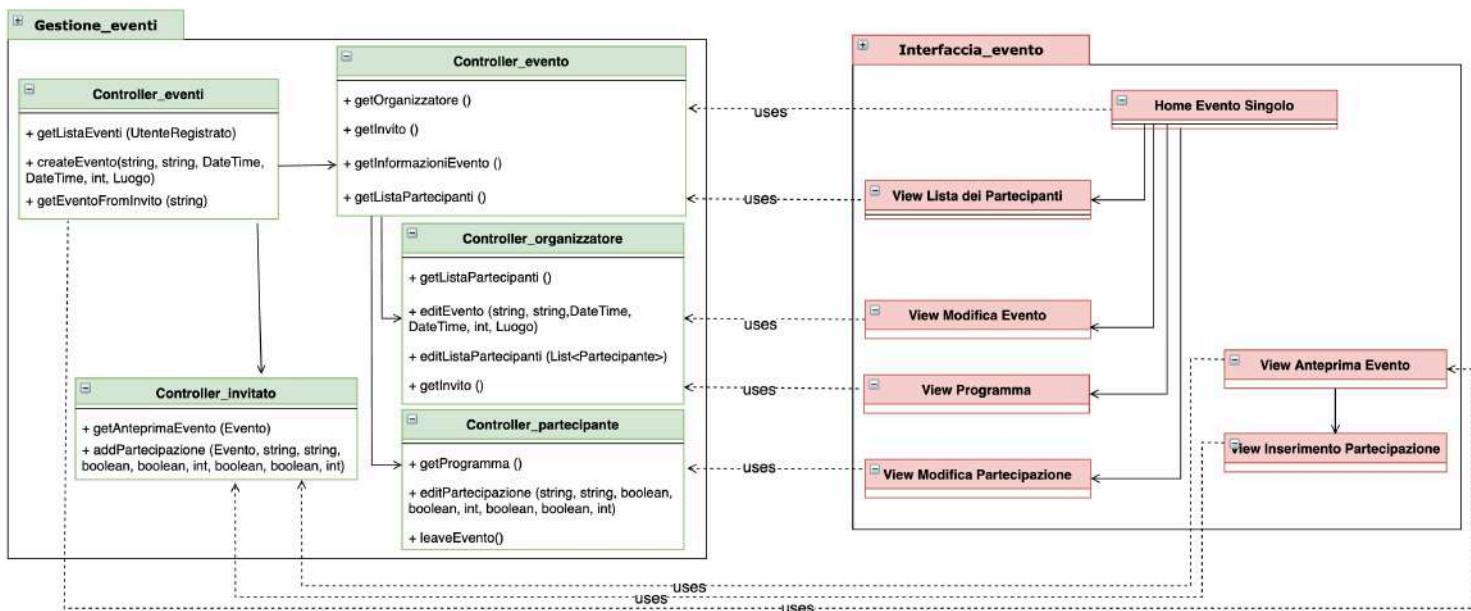
Le classi **Controller\_Login**, **Controller\_Registrazione**, **Controller\_Recovery** si occuperanno di tutto ciò che concerne la parte di accesso ai servizi offerti dall'applicazione.

## Diagramma delle classi: Gestione Utente



Controller\_utente permetterà di accedere ai dati dell'utente, mentre Controller\_impostazioni permetterà di modificarli.

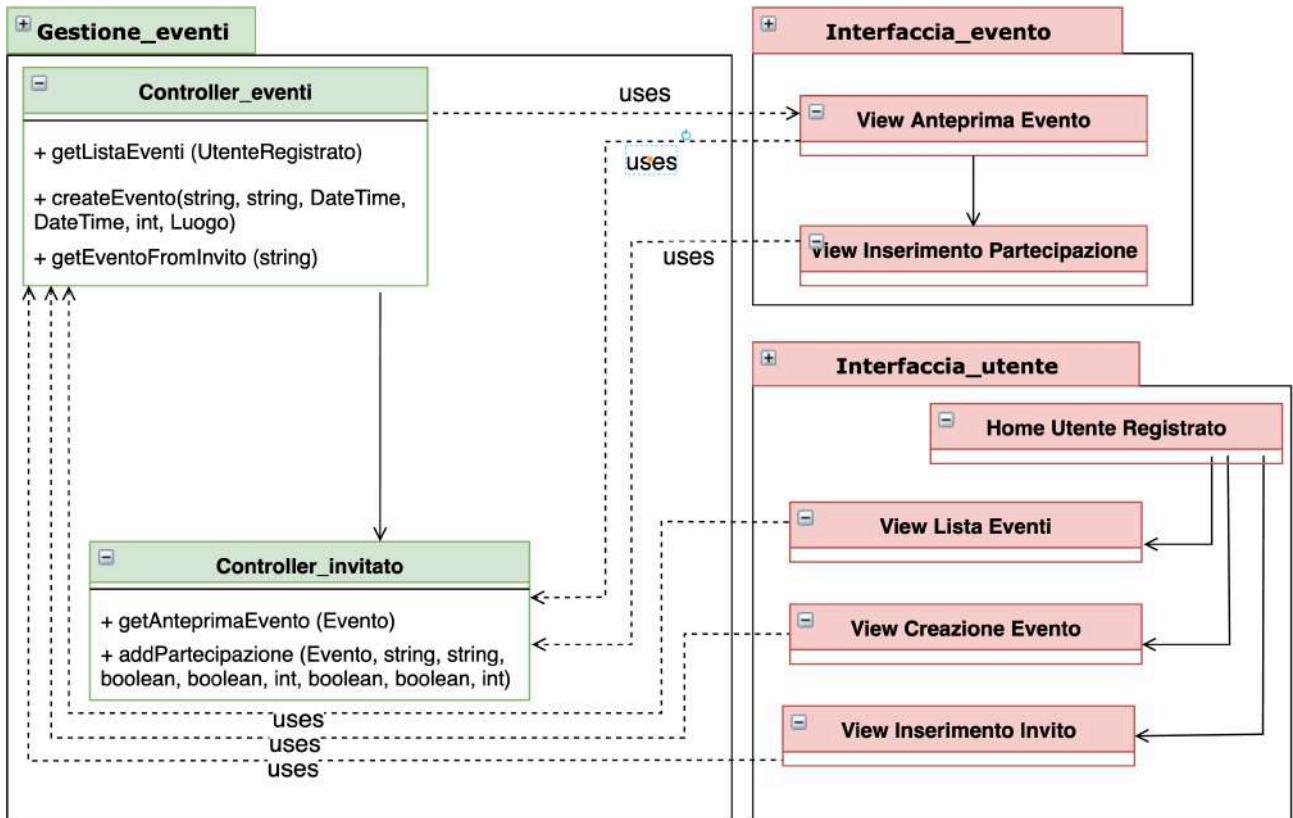
## Diagramma delle classi: Gestione Evento



Tutto il package dei controller serve a gestire tutto ciò che concerne un evento. Inoltre per ogni attore è stato previsto un controller dedicato. In questo diagramma si possono vedere tutte le possibili interazioni di Partecipanti e Organizzatori con l'evento.

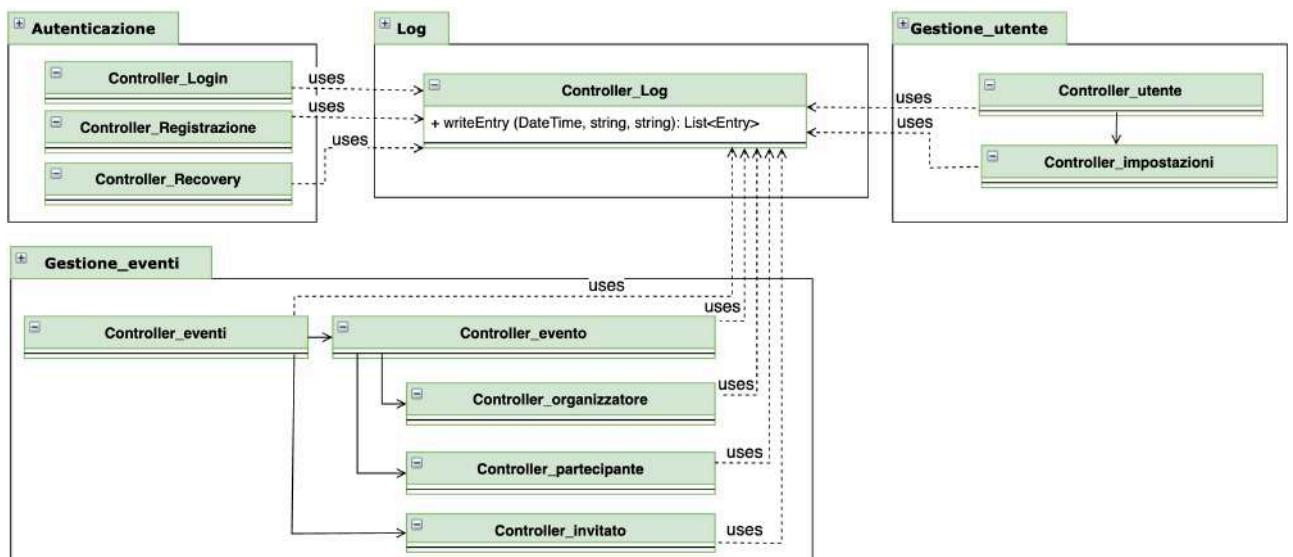
Inoltre è anche presente una coppia di view apposita per gli invitati che si stanno iscrivendo. Essa non sarà accessibile direttamente in nessun caso poiché sarà il Controller\_Eventi a mostrarla solo in seguito all'inserimento di un invito valido. (Vedi in seguito)

## Diagramma delle classi: Interazione Evento - Utente Registrato



Come accennato in precedenza è qui riportato come il **Controller\_eventi** è utilizzato per azioni di Utente Registrato come la visualizzazione della lista degli eventi e la creazione di un evento. In veste di Invitato invece, all'inserimento di un invito, il **Controller\_eventi** restituirà le **View Anteprima Evento** e di Aggiunta Partecipazione. (Notare la freccia invertita da **Controller\_eventi** a **View Anteprima Evento**)

## Diagramma delle classi: Interazione con i Log



In questa parte di diagramma delle classi si evince come tutti i controller facciano uso del **Controller\_Log** per la scrittura delle entry pertinenti ai loro rispettivi compiti.

## Architettura Logica: Interazione

Diagramma di sequenza: Login eseguito con successo

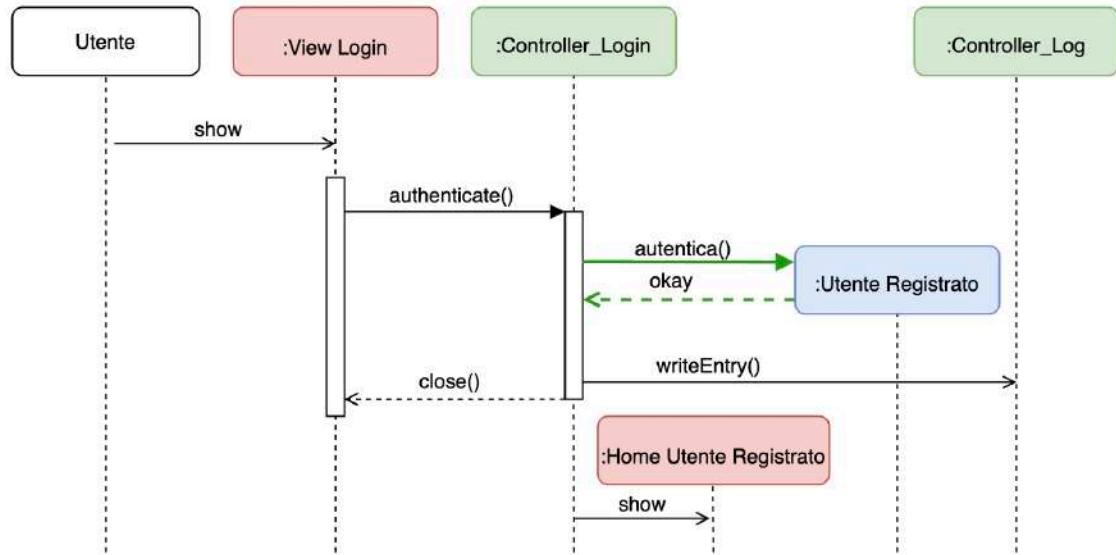


Diagramma di sequenza: Visualizzazione lista eventi

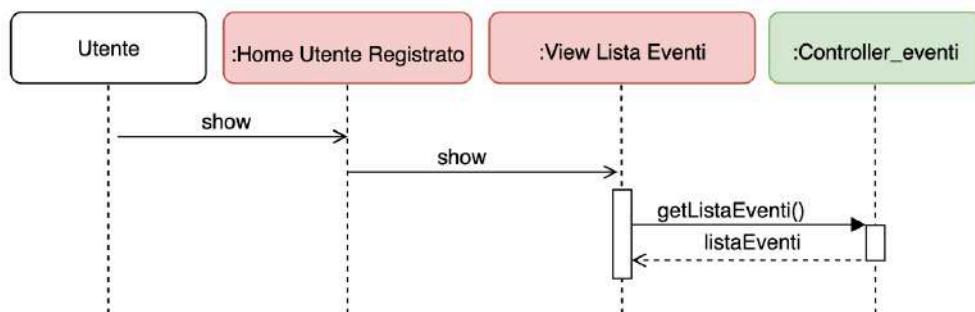
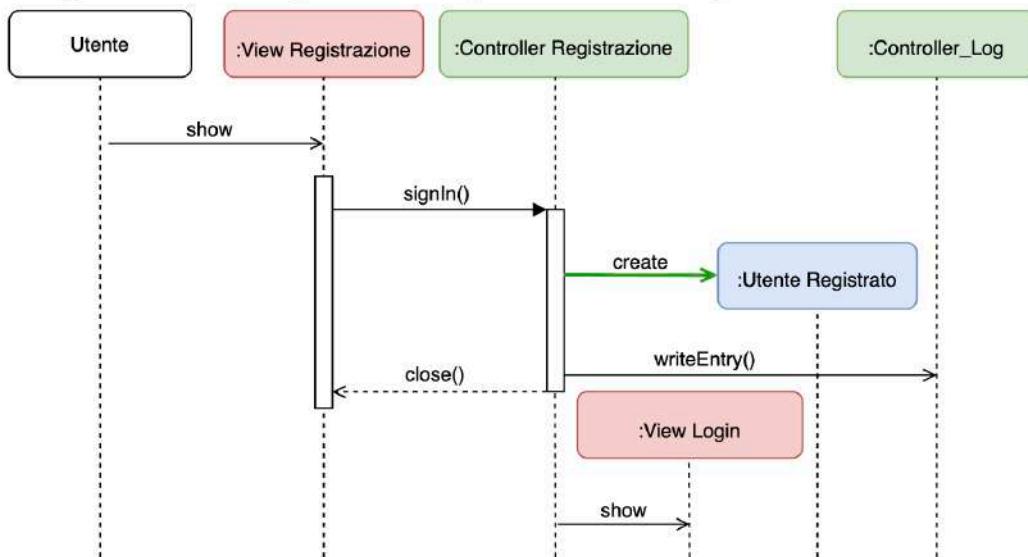
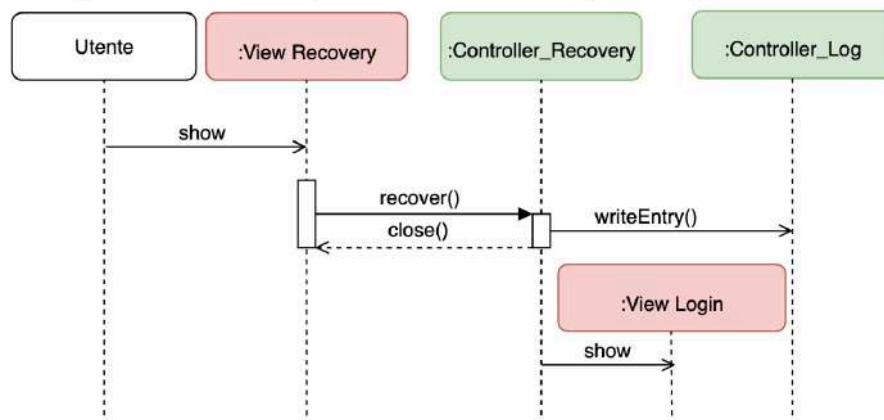


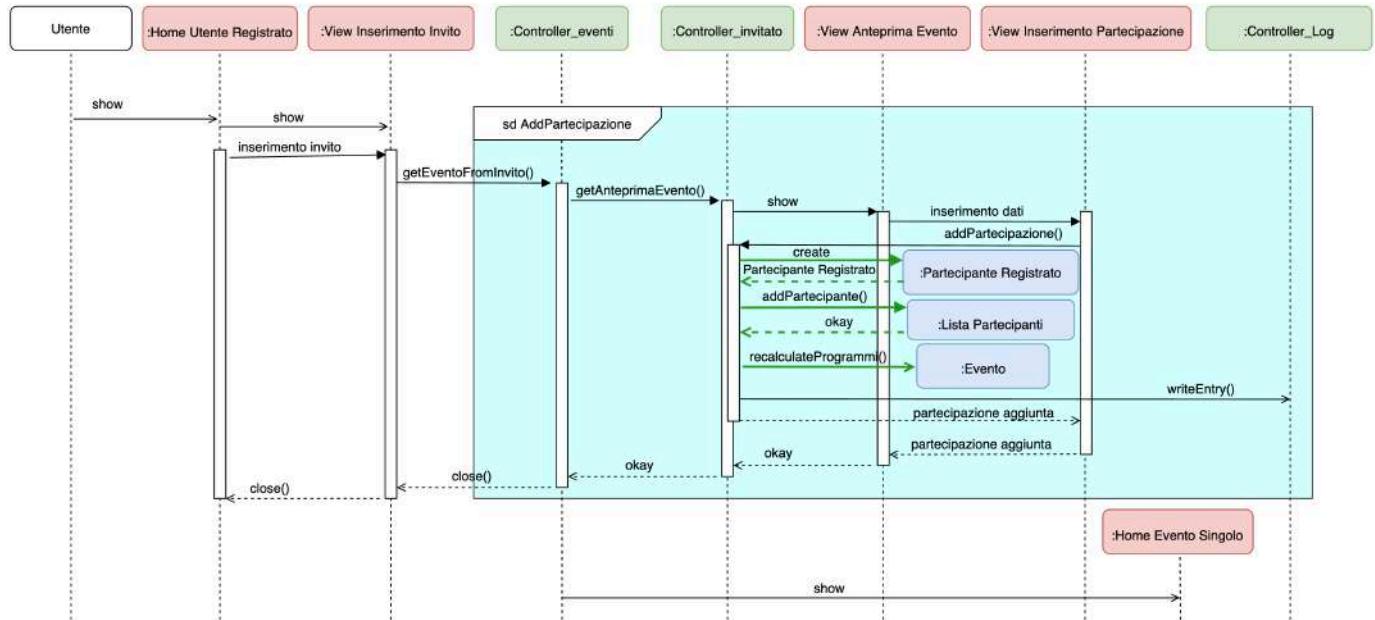
Diagramma di sequenza: Registrazione eseguito con successo



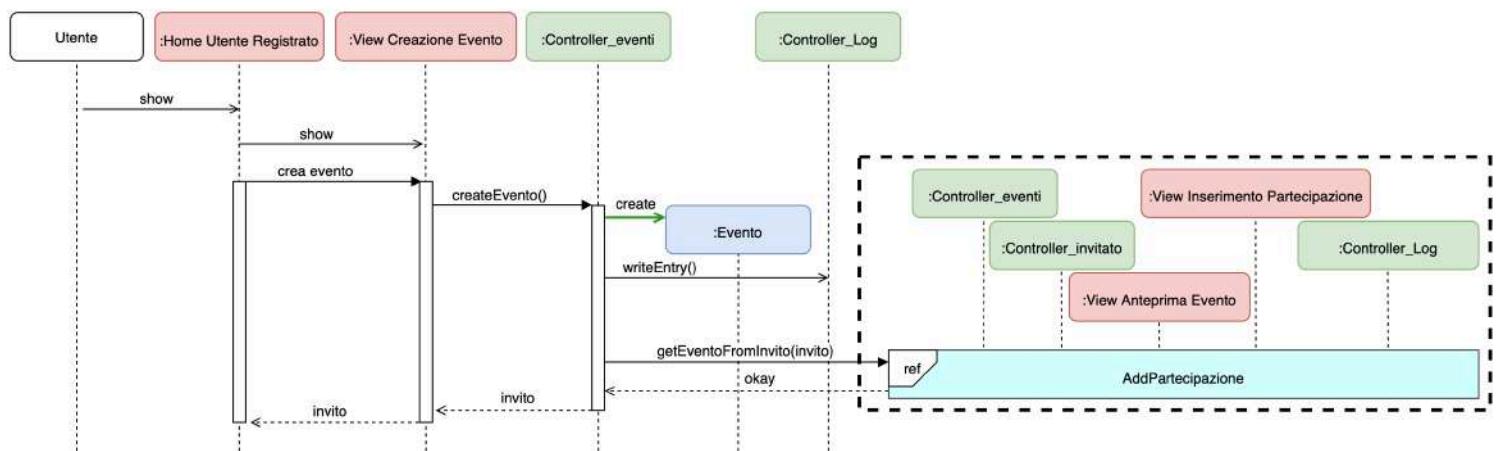
## Diagramma di sequenza: Recovery della password



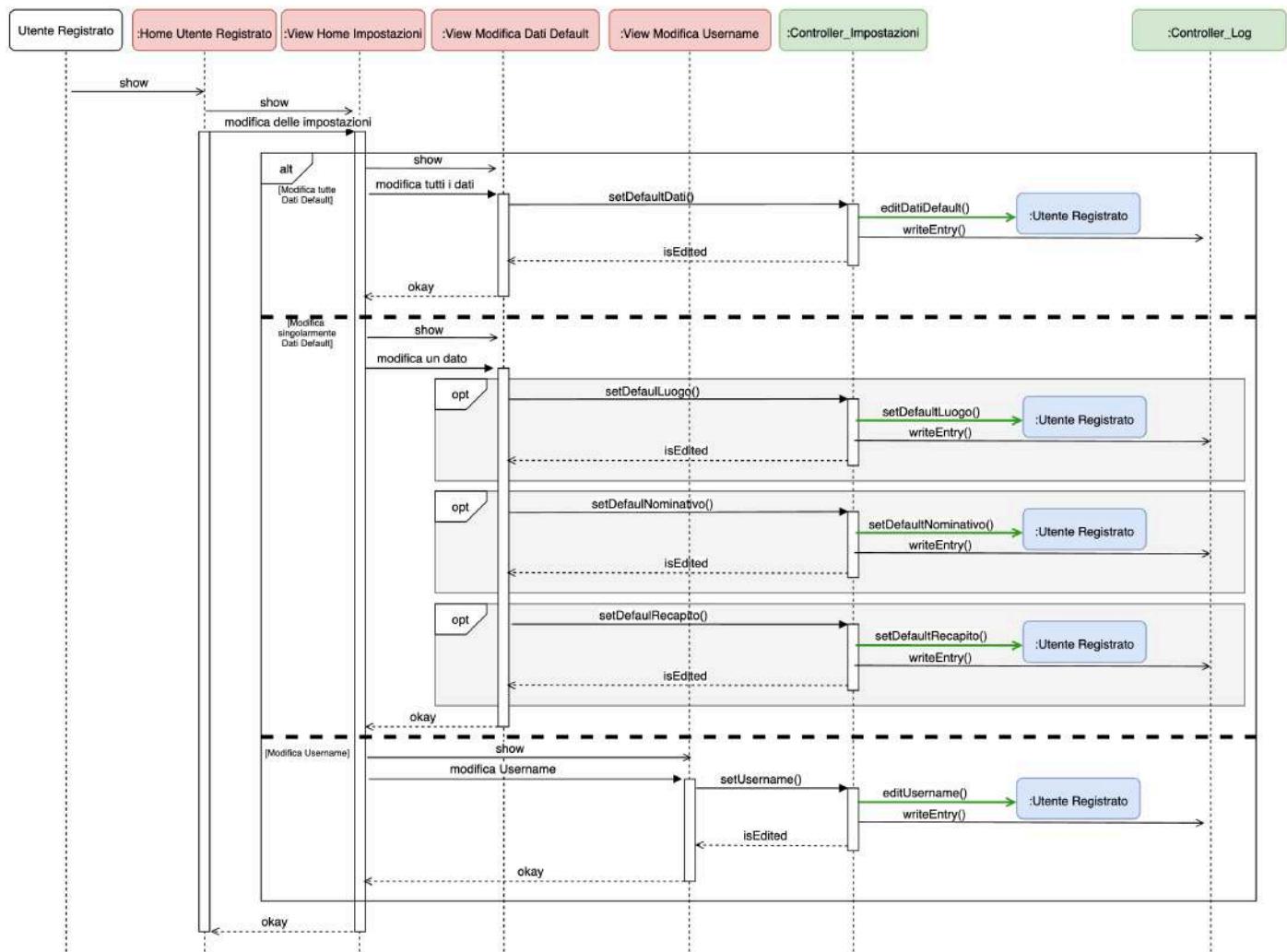
## Diagramma di sequenza: Aggiunta Partecipazione



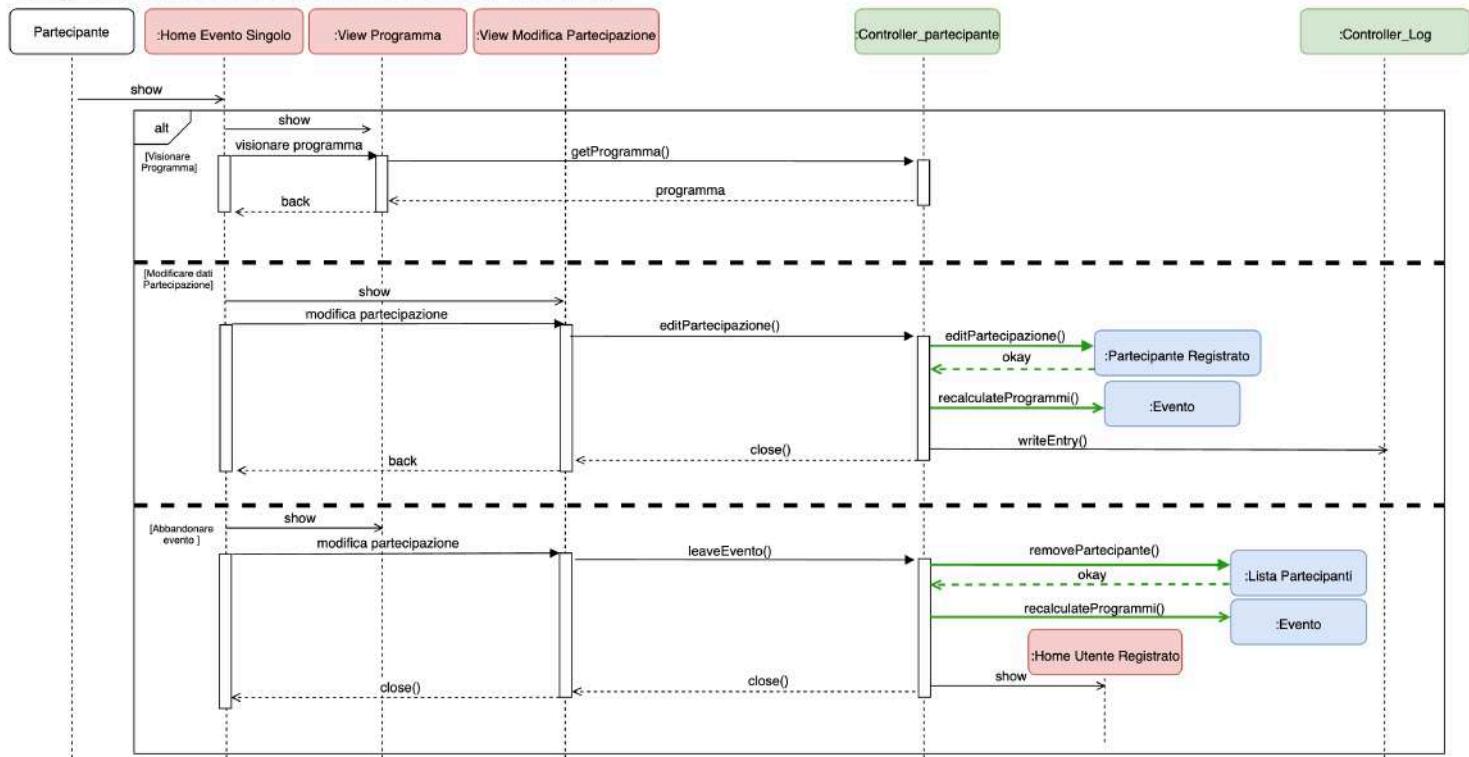
## Diagramma di sequenza: Creazione di un nuovo evento



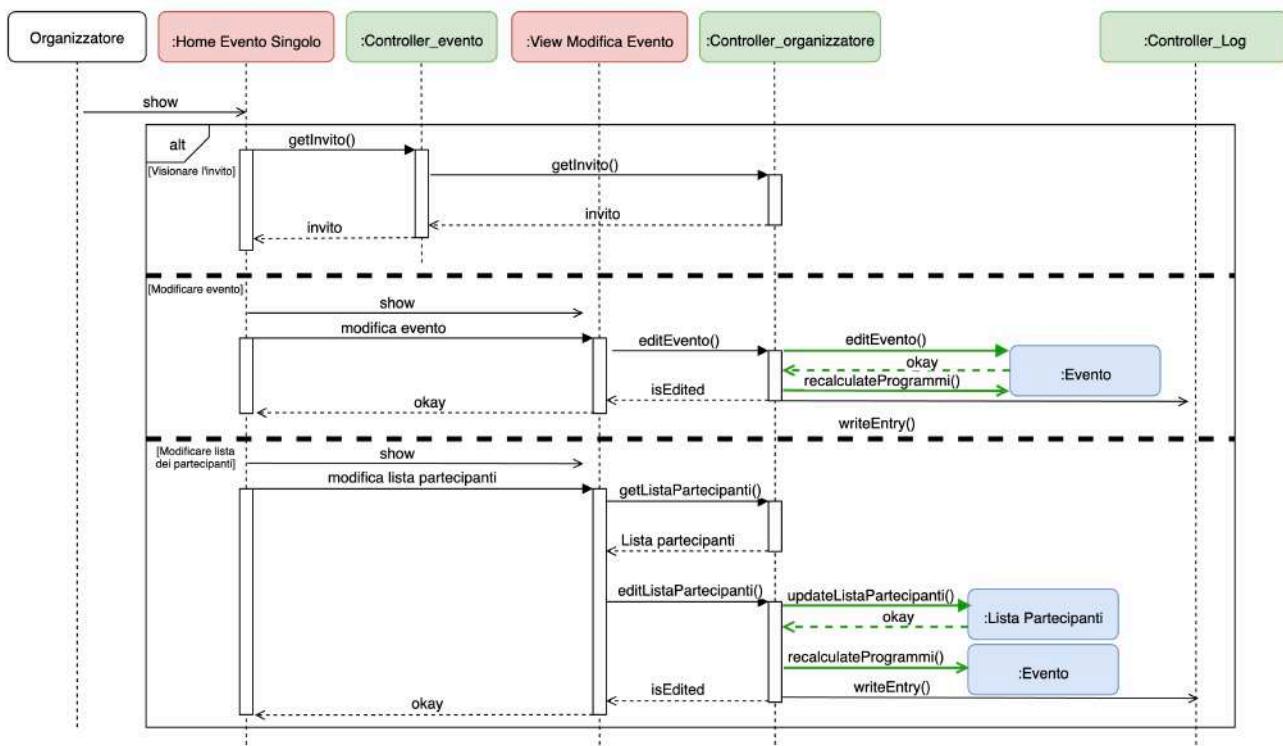
## Diagramma di sequenza: Modifica delle Impostazioni



## Diagramma di sequenza: Gestione evento Partecipante

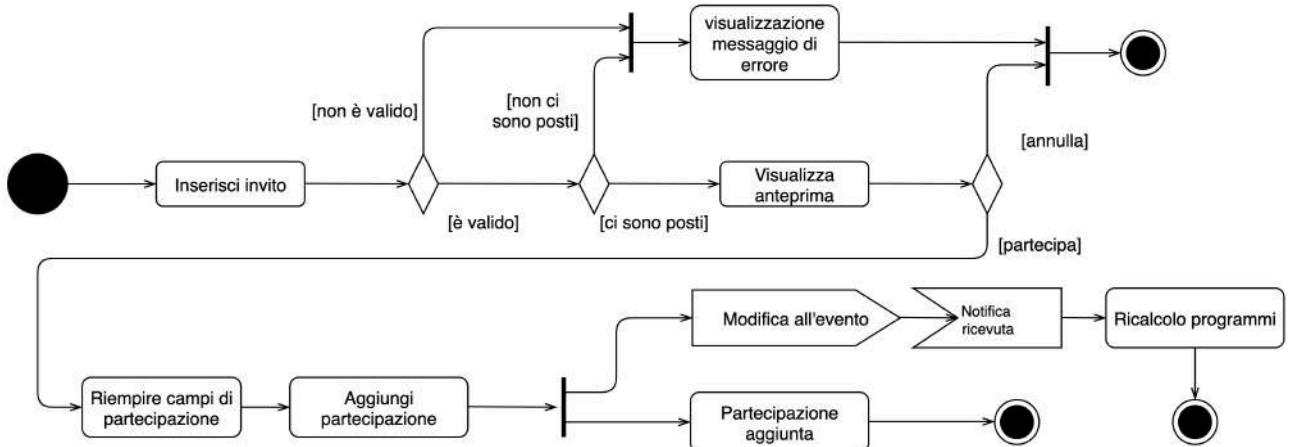


## Diagramma di sequenza: Gestione evento Organizzatore



# Architettura Logica: Comportamento

## Aggiunta Partecipazione



## Piano di Lavoro

Il progetto e lo sviluppo dell'applicazione sono assegnati ai diversi membri del team nel seguente modo:

Package	Progetto	Sviluppo
Dominio	Luca Levita, Giacomo Cerino, Jacopo Jop	Luca Levita, Giacomo Cerino
Gestione_utente	Luca Levita, Giacomo Cerino, Jacopo Jop	Giacomo Cerino
Gestione_eventi	Luca Levita, Giacomo Cerino, Jacopo Jop	Giacomo Cerino, Luca Levita
Log	Luca Levita, Giacomo Cerino, Jacopo Jop	Luca Levita
Autenticazione	Luca Levita, Giacomo Cerino, Jacopo Jop	Luca Levita
Interfaccia_utente	Luca Levita, Giacomo Cerino, Jacopo Jop	Jacopo Jop, Giacomo Cerino
Interfaccia_evento	Luca Levita, Giacomo Cerino, Jacopo Jop	Jacopo Jop
Interfaccia_impostazioni	Luca Levita, Giacomo Cerino, Jacopo Jop	Giacomo Cerino
Interfaccia_login	Luca Levita, Giacomo Cerino, Jacopo Jop	Jacopo Jop

I tempi di rilascio previsti sono i seguenti:

- Progettazione entro il 30 maggio
- Sviluppo delle singole parti con collaudo unitario entro il 12 giugno
- Integrazione e test dell'intero sistema entro il 30 giugno

### **Funzionalità prototipo:**

Il prototipo provvederà a fornire le seguenti funzionalità:

- Funzionalità del client
- Database prototipale relazionale
- Connessioni in chiaro
- Calcolo casuale dei gruppi senza mappa integrata né indicazioni

Si richiede al Team progettazione di tenere conto dei seguenti sviluppi futuri.

### **Sviluppi futuri:**

Successivamente alla produzione e alla presentazione del prototipo, se si riceverà una richiesta da parte del mercato, si prevede l'implementazione delle seguenti funzionalità:

- Integrazione delle API di google grazie all'acquisto di una licenza da sviluppatore
- Cifratura delle comunicazioni tra client e server (accennata nella progettazione)
- Implementazione dell'applicazione anche all'interno di un bot di Telegram
- Interfaccia utente facile, intuitiva, fluida e adatta a più dispositivi
- Supporto per più lingue
- Gestione notifiche

# Piano del Collaudo

```
1 //Test classe del dominio Evento
2 [TestFixture]
3 public class TestEvento
4 {
5     private Evento _e;
6     [SetUp]
7
8     public void EventoSetup() {
9
10         Organizzatore _organizzatore = new
11             Organizzatore("Gmail@gmail.com", "password", "GimmyMail", "5963412532");
12         Luogo _lEvento = new Luogo ("Via S. Ruffillo 45", "40141", "Bologna", "B0", "Italia");
13         _e = new Evento(_organizzatore, "Festa di bocciatura", "avvenimento usuale dopo ogni
14             appello di ingegneria del software", new DateTime(2021, 6, 11, 15, 0, 0), new
15             DateTime(2021, 6, 11, 18, 0, 0), 50, _lEvento );
16     }
17     [Test]
18     public void TestMethod() {
19         Assert.That(_e.GetOrganizzatore().GetUsername(), Is.EqualTo("Gmail@gmail.com"));
20         Assert.That(_e.GetNome(), Is.EqualTo("Festa di bocciatura"));
21         Assert.That(_e.GetDescrizione(), Is.EqualTo("avvenimento usuale dopo ogni appello di
22             ingegneria del software"));
23         Assert.That(_e.GetLuogo().GetIndirizzo(), Is.EqualTo("Via S. Ruffillo 45"));
24         Assert.That(_e.GetLuogo().GetCAP(), Is.EqualTo("40141"));
25         Assert.That(_e.GetLuogo().GetCitta(), Is.EqualTo("Bologna"));
26         Assert.That(_e.GetLuogo().GetProvincia(), Is.EqualTo("B0"));
27         Assert.That(_e.GetLuogo().GetNazione(), Is.EqualTo("Italia"));
28         Assert.That(_e.GetData_Ora_inizio(), Is.EqualTo(new DateTime(2021, 6, 11, 15, 0, 0)));
29         Assert.That(_e.GetData_Ora_fine(), Is.EqualTo(new DateTime(2021, 6, 11, 18, 0, 0)));
30         Assert.That(_e.GetMaxPartecipanti(), Is.EqualTo(50));
31         Assert.That(_e.GetListaPartecipanti().getNumeroPartecipanti(), Is.EqualTo(0)); //è
32             vuota perchè non abbiamo inserito nessun partecipante
33
34         //editEvento
35         _e.editEvento("Festa di bocciatura", "Bottom text", new DateTime(2021, 6, 11, 15, 0, 0),
36             new DateTime(2021, 6, 11, 18, 0, 0), 100, _lEvento );
37         Assert.That(_e.GetDescrizione(), Is.EqualTo("Bottom text"));
38         Assert.That(_e.GetMaxPartecipanti(), Is.EqualTo(100));
39
40         string invito = _e.GetInvito();
41         Assert.That(invito, Is.Not.EqualTo(null));
42
43
44         Assert.That(_e.GetGruppi.Count(), Is.EqualTo(0));
45     }
46 }
```

```

1 //Test classe del dominio Utente Registrato
2 [TestFixture]
3 public class TestUtenteRegistrato
4 {
5     private UtenteRegistrato _ur;
6     [SetUp]
7
8     public void UtenteRegistratoSetUp() {
9         _ur = new UtenteRegistrato("Alice@alice.it","password");
10        Luogo _l = new Luogo ("Via S. Donato 130", "40057", "Granarolo dell'Emilia",
11        "BO","Italia")
12        _ur.SetNominativoDefault("Aly");
13        _ur.SetLuogoDefault(_l);
14        //_ur.SetRecapitoDefault("123548912");
15    }
16    [Test]
17    public void TestMethod() {
18        Assert.That(_ur.GetUsername(), Is.EqualTo("Alice@alice.it"));
19        Assert.That(_ur.Autentica("Alice@alice.it","password"), Is.EqualTo(true));
20        Assert.That(_ur.HasNominativo(), Is.EqualTo(true));
21        Assert.That(_ur.GetNominativo(), Is.EqualTo("Aly"));
22        Assert.That(_ur.HasLuogo(), Is.EqualTo(true));
23        Assert.That(_ur.GetLuogo().GetIndirizzo(), Is.EqualTo("Via S. Donato 130"));
24        Assert.That(_ur.GetLuogo().GetCAP(), Is.EqualTo("40057"));
25        Assert.That(_ur.GetLuogo().GetCitta(), Is.EqualTo("Granarolo dell'Emilia"));
26        Assert.That(_ur.GetLuogo().GetProvincia(), Is.EqualTo("BO"));
27        Assert.That(_ur.GetLuogo().GetNazione(), Is.EqualTo("Italia"));
28        Assert.That(_ur.HasRecapito(), Is.EqualTo(false));
29        Assert.That(_ur.GetRecapito(), Is.EqualTo(null));
30
31        Organizzatore _organizzatore = new
32        Organizzatore("Gmail@gmail.com","password","GimmyMail","5963412532");
33        Luogo _l = new Luogo ("Via S. Donato 130", "40057", "Granarolo dell'Emilia",
34        "BO","Italia")
35
36        Luogo _lEvento = new Luogo ("Via S. Ruffillo 45", "40141", "Bologna", "BO","Italia");
37
38        Evento _e = new Evento(_organizzatore, "Festa di bocciatura", "Bottom text", new
39        DateTime(2021,6,11,15,0,0), new DateTime(2021,6,11,18,0,0), 50, _lEvento);
40
41        string invito = _e.GetInvito();
42        Assert.That(invito, Is.Not.EqualTo(null));
43        Assert.That(_ur.InserisciInvito(invito), Is.EqualTo(_e));
44        Assert.That(_ur.AcceptInvito(_ur.InserisciInvito(invito),_ur.GetNominativo(),"32544334
45        2665",true,false,0,_ur.GetLuogo(),false,false,0,null), Is.EqualTo(true));
46        Assert.That(_e.GetListaPartecipanti().getNumeroPartecipanti(), Is.EqualTo(1));
47
48        _ur.editUsername("Alice@gmail.com");
49        Assert.That(_ur.GetUsername(), Is.EqualTo("Alice@gmail.com"));
50
51        Luogo _l2 = new Luogo ("Via S. Martino 150", "40127", "Gainazzo", "MO","Italia")
52        _ur.editDatiDefault(_l2, "Ally", "32562182562");
53        Assert.That(_ur.HasNominativo(), Is.EqualTo(true));
54        Assert.That(_ur.GetNominativo(), Is.EqualTo("Ally"));
55        Assert.That(_ur.HasLuogo(), Is.EqualTo(true));
56        Assert.That(_ur.GetLuogo().GetIndirizzo(), Is.EqualTo("Via S. Martino 150"));
57        Assert.That(_ur.GetLuogo().GetCAP(), Is.EqualTo("40127"));
58        Assert.That(_ur.GetLuogo().GetCitta(), Is.EqualTo("Gainazzo"));
59        Assert.That(_ur.GetLuogo().GetProvincia(), Is.EqualTo("MO"));
60        Assert.That(_ur.GetLuogo().GetNazione(), Is.EqualTo("Italia"));
61        Assert.That(_ur.HasRecapito(), Is.EqualTo(true));
62        Assert.That(_ur.GetRecapito(), Is.EqualTo("32562182562"));
63    }
64 }

```

```

1 //Test classe del dominio Partecipante Registrato
2 [TestFixture]
3 public class TestPartecipanteRegistrato
4 {
5     private PartecipanteRegistrato _pr;
6
7     [SetUp]
8     public void PartecipanteRegistratoSetup() {
9         Luogo _lprAndata = new Luogo ("Via S. Ragozza 32", "40125", "Bologna", "BO","Italia");
10        _pr = new PartecipanteRegistrato("Jerry", "56232178965", true, false, 0 _lprAndata,
11        * false, false, 0, null);
12    }
13
14    [Test]
15    public void TestMethod() {
16        Assert.That(_pr.GetOrganizzatore().GetUsername(), Is.EqualTo("Gmail@gmail.com"));
17        Assert.That(_pr.GetNominativo(), Is.EqualTo("Jerry"));
18        Assert.That(_pr.GetRecapito(), Is.EqualTo("56232178965"));
19        Assert.That(_pr.HasPartecipazioneCarpoolAndata(), Is.EqualTo(true));
20        Assert.That(_pr.HasDisponibilitaVeicoloAndata(), Is.EqualTo(false));
21        Assert.That(_pr.GetPostiDisponibili(), Is.EqualTo(0));
22
23        Assert.That(_pr.GetLuogoAndata().GetIndirizzo(), Is.EqualTo("Via S. Ragozza 32"));
24        Assert.That(_pr.GetLuogoAndata().GetCAP(), Is.EqualTo("40125"));
25        Assert.That(_pr.GetLuogoAndata().GetCitta(), Is.EqualTo("Bologna"));
26        Assert.That(_pr.GetLuogoAndata().GetProvincia(), Is.EqualTo("BO"));
27        Assert.That(_pr.GetLuogoAndata().GetNazione(), Is.EqualTo("Italia"));
28
29        Assert.That(_pr.HasPartecipazioneCarpoolAndata(), Is.EqualTo(false));
30
31        //editPartecipazione
32        _pr.editPartecipazione("Tom", "56232178965", true, false, 0 _lprAndata, false, false,
33        * 0, null)
34        Assert.That(_pr.GetNominativo(), Is.EqualTo("Tom"));
35    }

```

# Progettazione

## Progettazione Architetturale

### Requisiti non funzionali

Nell'Analisi del Problema (Tabella Vincoli) sono emersi diversi requisiti non funzionali che impongono dei vincoli al sistema:

- Integrità dei dati
- Usabilità
- Il sistema deve essere distribuito
- L'accesso diretto al database è permesso soltanto all'amministratore
- Le interazioni con il database devono avvenire in maniera sicura e controllata

Nello specifico caso in esame, Usabilità e Sicurezza non si intralciano tra loro.

L'Usabilità impatta molto sulla struttura delle interfacce, che andranno progettate in modo tale da permettere una navigazione intuitiva tra le varie maschere dell'applicazione. (es. pochi bottoni, esplicativi e con una chiara funzione associata).

L'applicazione deve essere veloce nelle sue operazioni. Eventuali ritardi (nell'ordine dei millisecondi) potrebbero essere causati da processi di convalidazione dell'input o di verifica dell'output; tuttavia ciò non è un problema poiché la nostra applicazione non ha vincoli real-time da soddisfare. Inoltre gli utenti del sistema sono esseri umani e perciò difficilmente si accorgono di ritardi durante le risposte.

Considerando la tipologia di sistema che deve essere sviluppato, non si ritiene particolarmente critico l'aspetto di sicurezza dei dati in quanto da come si evince dalla "Tabella di Valutazione Beni", l'unica perdita di dati di rilievo è comunque una perdita contenuta, che può portare soltanto ad un danno di immagine all'applicazione. Difficilmente gli attacchi porteranno a conseguenze legali.

Per garantire l'integrità dei dati prevediamo di utilizzare un database di tipo relazionale che, opportunamente configurato e progettato, controllerà automaticamente l'integrità dei dati e delle operazioni prevenendo danni provocati da utenti e amministratori umani.

Riguardo al vincolo di accesso al database esclusivo per l'amministratore utilizzeremo metodologie che non permettono l'SQL injection nelle nostre richieste.

Per sopperire al vincolo del sistema distribuito utilizzeremo l'architettura client-server.

## Scelta dell'architettura

Dal punto di vista architetturale, la migliore scelta per questa applicazione è un'architettura client-server a tre livelli:

- L1: Client

Il client sarà un'applicazione mobile utilizzabile soltanto da utenti semplici. Non è infatti prevista alcuna funzionalità da amministratore.

Esso si conserverà al server con una connessione in chiaro. In futuro si prevede di aggiornare le comunicazioni al protocollo TLS, in modo da soddisfare il requisito della protezione delle comunicazioni.

- L2: Server

Ci sarà un solo server che comunica con tutti i client e gestirà l'interazione tra i client e il database. Il quale sarà attivo su un'altra macchina.

- L3: Persistenza

Per quanto riguarda la persistenza abbiamo optato per un database relazionale situato in un server indipendente. Questa scelta è motivata dalla possibile presenza di picchi di richieste.

Ad esempio nel momento di creazione dell'evento (e della conseguente moltitudine di richieste di aggiunta di partecipazione), oppure anche al momento della partenza per raggiungere il luogo di un evento (momento nel quale si avrà una grande richiesta di visionare i programmi) si concentreranno molte richieste. Adottando questo metodo, possiamo evitare congestioni e rallentamenti del sistema lasciando che i due server si dividano il carico di lavoro; l'uno con la gestione delle richieste e l'altro col mantenimento dei dati.

Sarà così possibile servire le richieste in maniera concorrente senza perdita di prestazioni e limitando il costo in termini di tempo.

L'interfacciamento con il DBMS avverrà attraverso la metodologia "forza bruta" utilizzando i metodi CRUD

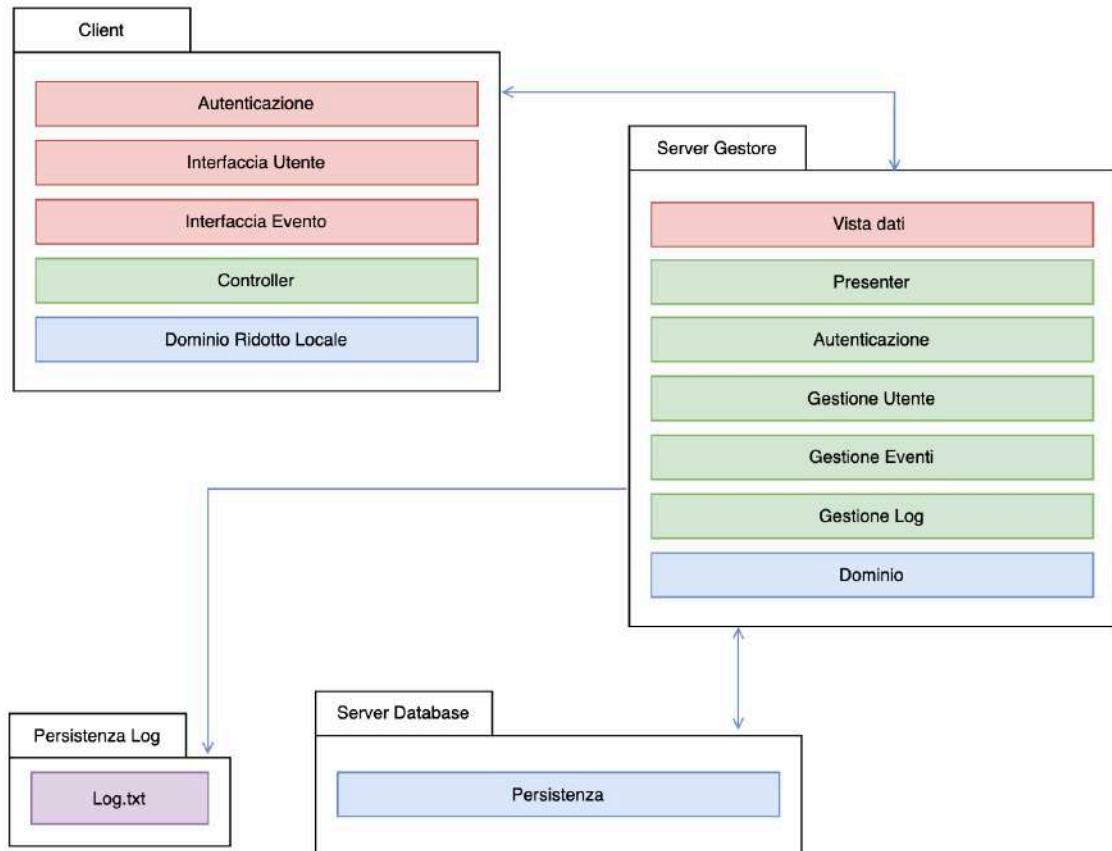
## Pattern Architetturali

Abbiamo deciso di adottare da parte del Server Gestore un pattern architetturale di tipo MVP provvisto di un componente View minimo e passivo il cui unico scopo è il passaggio di dati al client.

Invece per quanto riguarda il Client, abbiamo deciso di utilizzare il pattern MVC orientato principalmente alla presentazione di ciò che il server Gestore fornisce. La parte di logica di business sarà minima. La parte di model sarà ristretta alle sole classi del dominio consultabili dall'utente

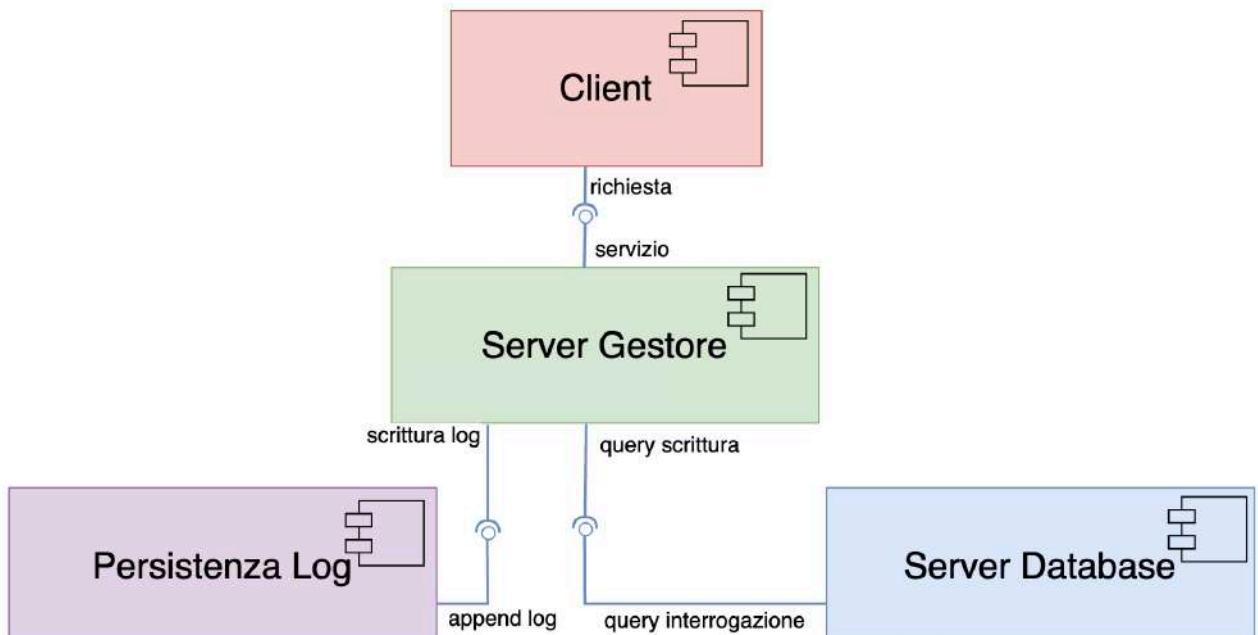
## Architettura del sistema

Nella figura sottostante è presente il diagramma che spiega l'architettura del sistema



## Diagramma dei componenti

Di seguito è riportata l'architettura a componenti del sistema



## Scelte Tecnologiche

Per quanto riguarda la comunicazione tra il client e il server gestore abbiamo optato per l'utilizzo del formato di dati Json vista la sua facilità di utilizzo, la presenza di librerie già pronte, la leggerezza dei messaggi e la comprensibilità del testo anche da parte di operatori umani.

## Progettazione di Dettaglio

### Struttura

Nel dominio inserito nel client saranno apportati opportuni cambiamenti al fine di limitare la vista del client ai soli oggetti che gli competono.

Tutta la progettazione del nostro applicativo si basa sul Dependency Inversion Principle. Ovvero tutte le classi manipolatrici di dati avranno un'interfaccia che permetterà anche di modificarne l'implementazione in modo facile, senza dover intervenire sul codice di moduli concettualmente distaccati.

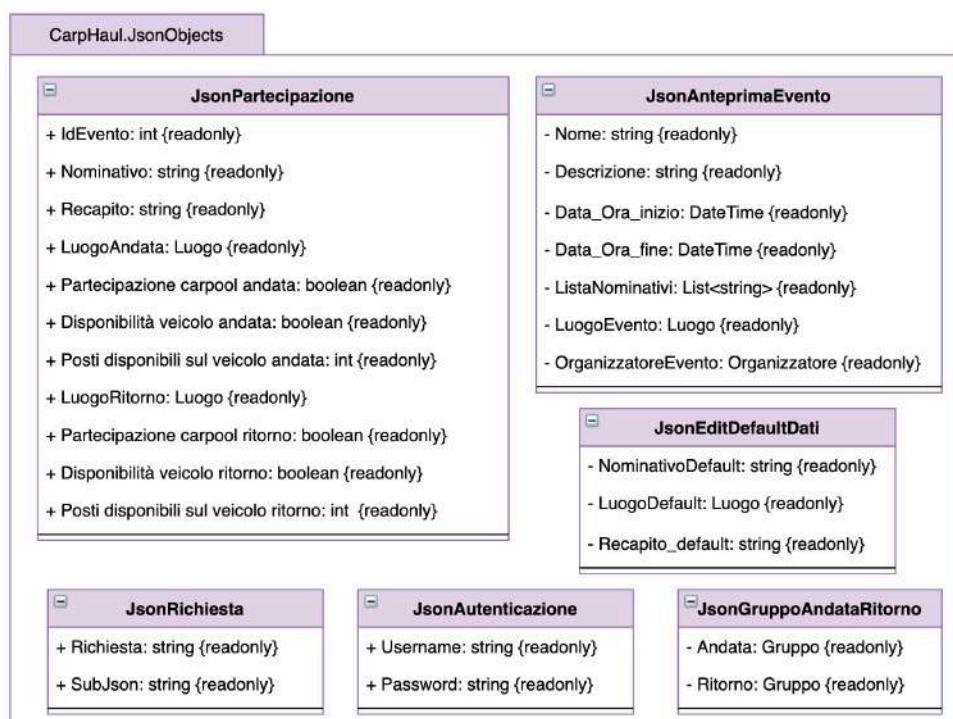
Avendo un solo Client l'Interface Segregation Principle è per forza soddisfatto poichè il client è di un unico tipo.

## Package di Utilità per Client e Server Gestore

Cominciamo con una presentazione di alcuni package che saranno utilizzati sia su Client che su Server Gestore per permettere loro la comunicazione.

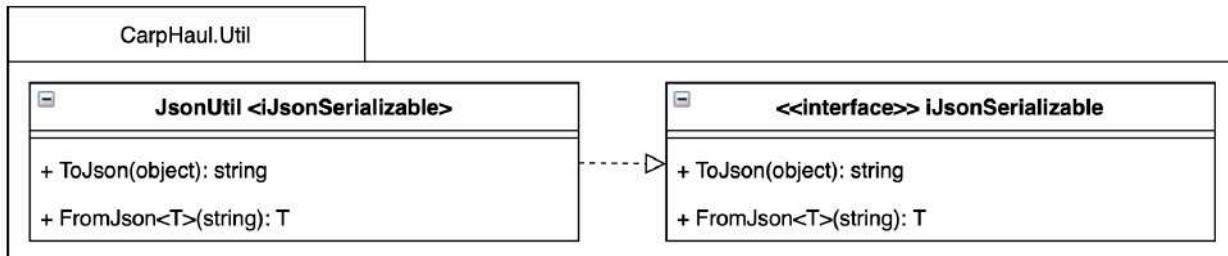
### Package: CarpHaul.JsonObjects

Sono classi che permettono la decodifica tra stringa in formato Json e oggetto legato al linguaggio. Infatti non tutte le richieste del Client sono convertibili in entità del dominio, ed è a questo che servono le suddette classi.



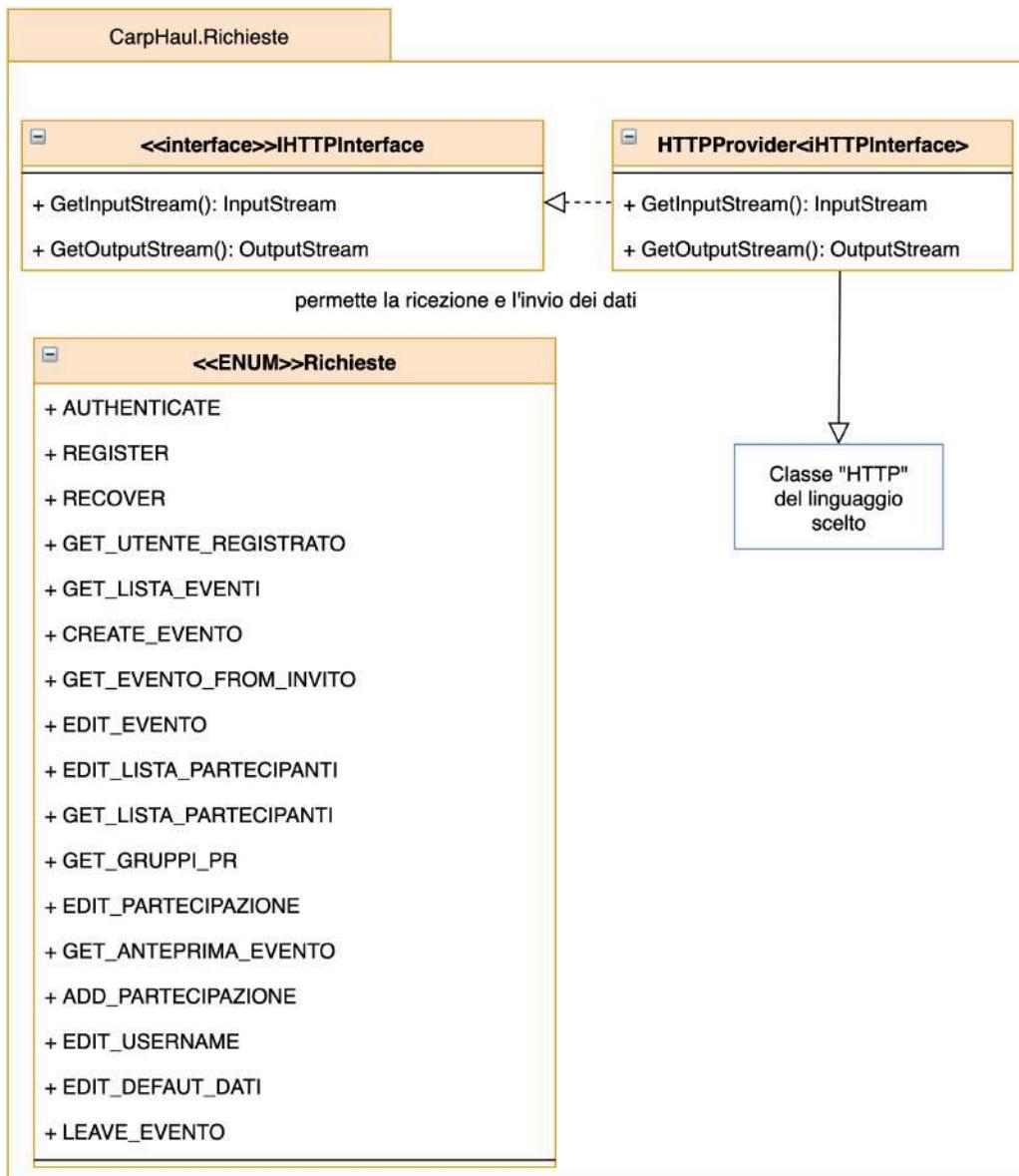
## Package: CarpHaul.Util

Sono classi di utilità che contengono funzioni che permettono la conversione e la decodifica di stringhe in formato Json e Oggetti.



## Package: CarpHaul.Richieste

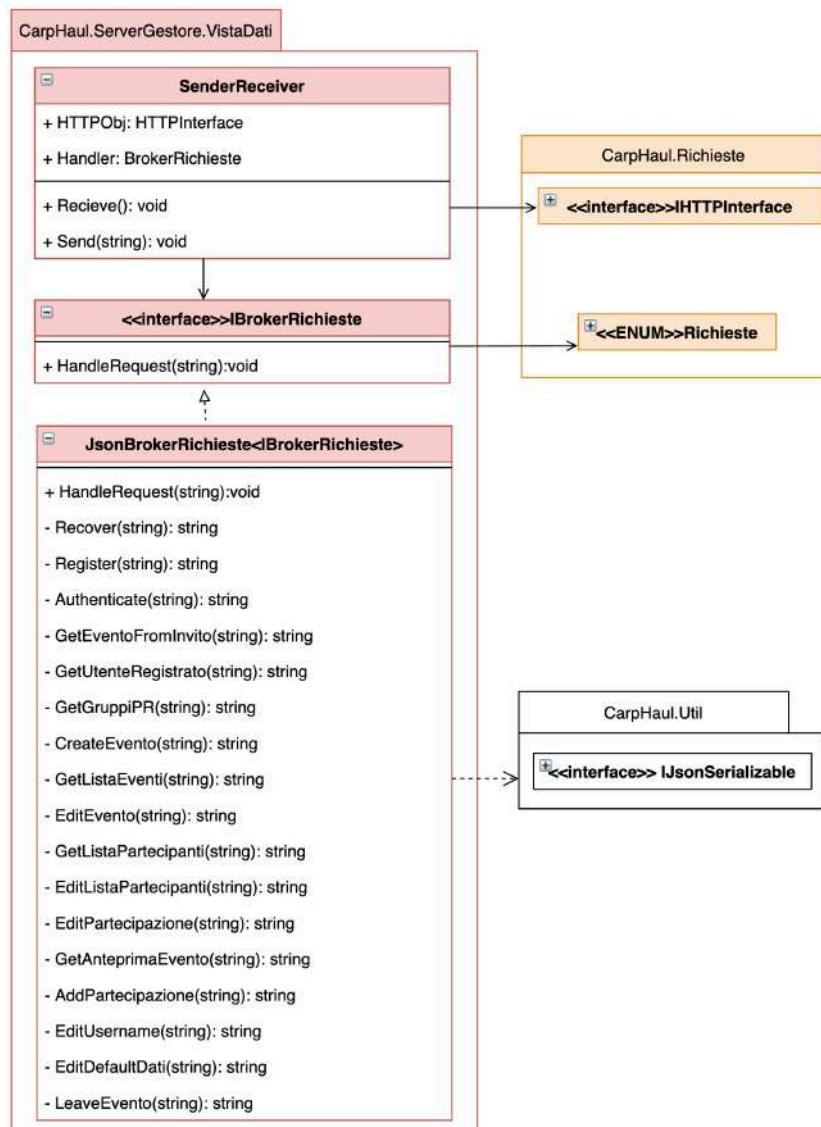
Classe che si interfaccia col sistema astraendosi dal linguaggio per gestire le funzionalità HTTP per le richieste. Abbiamo utilizzato un enumerativo per differenziare le diverse richieste che il client può inviare. Questo meccanismo potrà essere cambiato utilizzando diversi URL per ogni richiesta.



## Struttura Server Gestore

Come accennato in precedenza la struttura è basata su MVP:

- Model: Si tratta della parte che mantiene le informazioni ottenute dal database. Conoscendo la struttura esatta del database e utilizzando le operazioni CRUD, delega tutto il lavoro di ricerca al DBMS, essendo questo ottimizzato per le operazioni di manipolazione dei dati.
- View: Implementata come vista passiva, si occupa della comunicazione con il client, provvedendo alla formattazione dei dati in Json e al loro invio.
- Presenter: è il componente che permette la comunicazione fra model e View, e contiene all'interno tutta la business logic necessaria per ogni operazione.



La classe SenderReceiver esporrà il metodo Receive che sarà chiamato all'arrivo di una richiesta HTTP a seconda del linguaggio di programmazione che verrà scelto per l'implementazione.

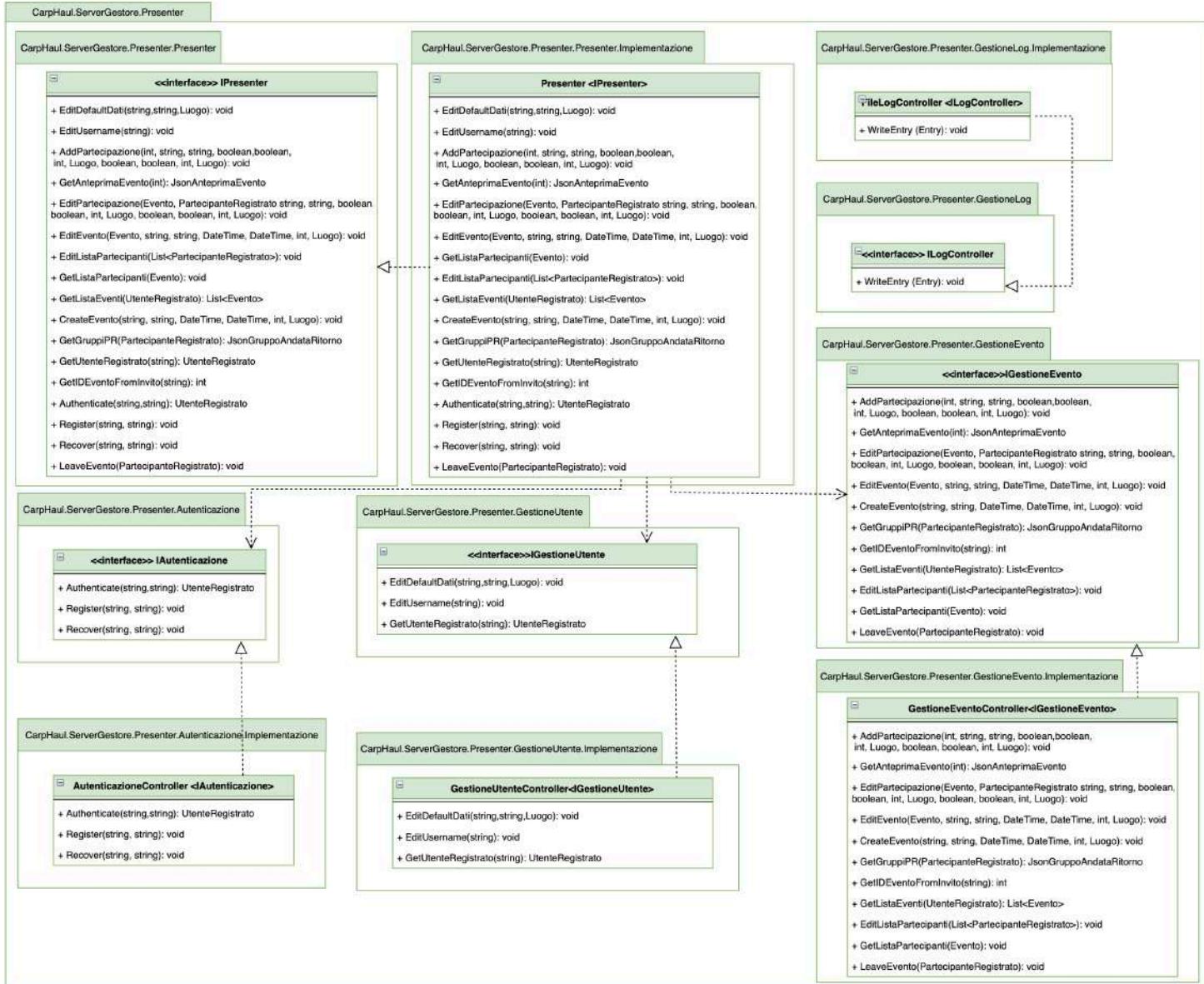
Il formato del nostro Json sarà strutturato così:

```

{
    "richiesta": "TIPO_RICHIESTA",
    "subJson": "{<Json specifico della
    richiesta>}"
}
    
```

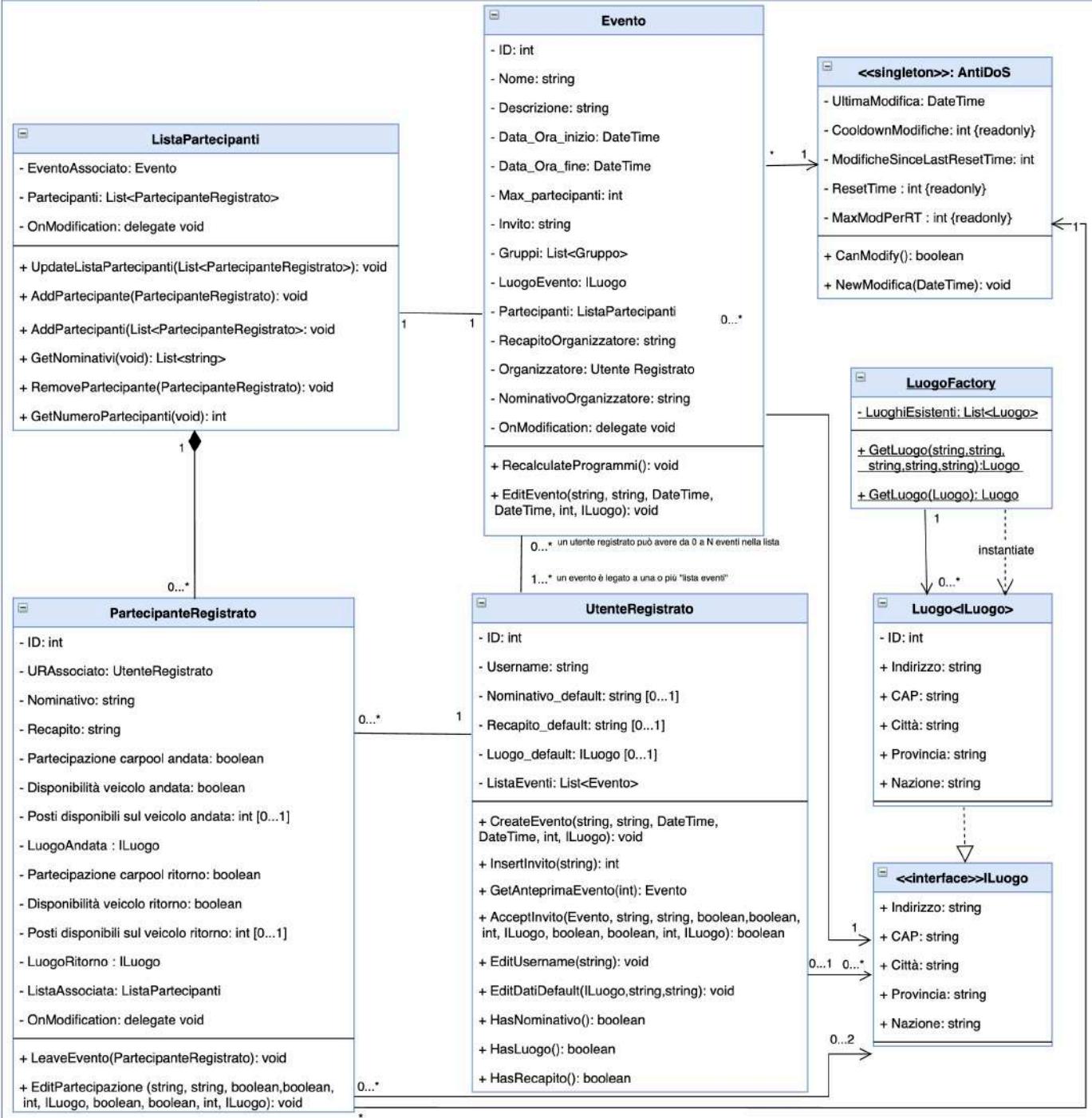
Dove “richiesta” è uno degli elementi presenti nell’enumerativo Richieste e subJson è il Json particolare per la richiesta stessa.

Strutturandolo così, rendiamo facile la deserializzazione in modo da capire in primo luogo ciò che richiede l’utente e successivamente comprendere il messaggio caratteristico della richiesta. Come accennato in precedenza, questo meccanismo può essere sostituito in fase di implementazione dedicando ad ogni tipo di richiesta un URL diverso.



L'interfaccia IPresenter mette a disposizione tutte le funzioni che il Client può richiedere.

Per rispettare il Single Responsibility Principle, la classe che implementerà l'interfaccia IPresenter delegherà ad altre interfacce minori e specializzate, l'azione da compiere. Esse saranno logicamente divise rispetto al tipo di interazione che avranno con il dominio. In questo modo si eviteranno anche dipendenze transitive.



Il diagramma soprastrante mostra una porzione di dominio dell'applicativo nel quale sono utilizzati il pattern OBSERVER (Delegate Based) per quanto riguarda il ricalcolo dei programmi, e il pattern FLYWEIGHT per la gestione dei luoghi.

## Pattern OBSERVER

Sono stati inseriti degli attributi di tipo delegate dentro alle classi del dominio che, se modificate, devono invocare il ricalcolo dei programmi. In questo modo è stato possibile implementare il Pattern Observer: Delegate Based, inserendo dentro a questi la funzione pertinente.

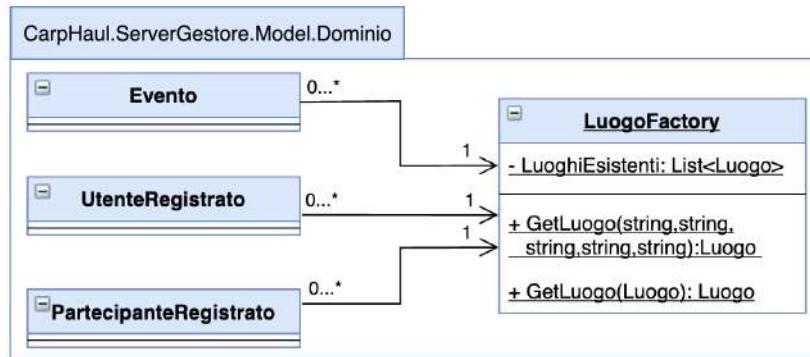
Benché il metodo `RecalculateProgrammi()` (colui che sarà delegato a queste variabili) potrebbe essere raggiunto da ognuna delle classi sopra citate attraverso un percorso

passante delle associazioni esistenti, abbiamo deciso di adoperare questa variabile in modo da ottimizzare le prestazioni e pensando all'aggiunta di un'eventuale futura funzionalità legata alla modifica di un evento.

In questo modo sarà possibile modificare la lista dei delegati aggiungendo l'operazione senza toccare il codice delle classi del dominio.

Per astrarre e appiattire il concetto davanti a tutte le classi coinvolte, è stato aggiunto l'attributo delegate anche nella classe che possiede l'implementazione del metodo RecalculateProgrammi().

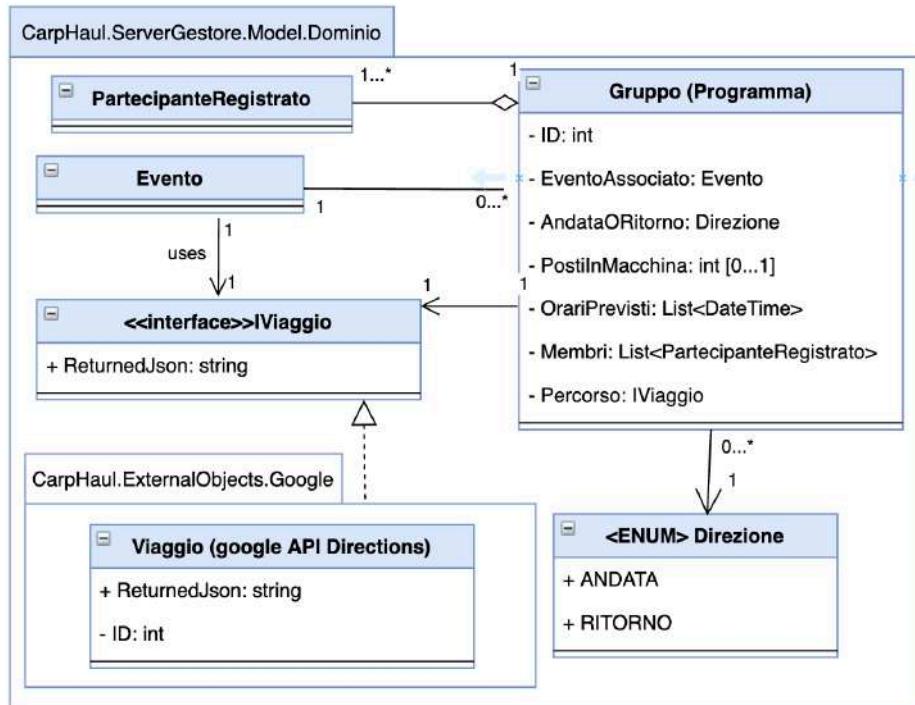
## Pattern FLYWEIGHT



Per quanto riguarda i Luoghi, data la probabilità elevata che più utenti abbiano collegati gli stessi luoghi (partenza, ritorno, evento), si è deciso di utilizzare il pattern FLYWEIGHT per la gestione e distribuzione di questi.

La classe statica LuogoFactory all'avvio del server precaricherà al suo interno tutti i luoghi esistenti nel database. A nessuna classe sarà consentito di istanziare un nuovo luogo se non passando dalla factory, la quale prima di creare uno nuovo verificherà che non sia già presente nella sua lista.

La factory espone due metodi per ottenere un Luogo: uno che ha bisogno di tutti gli attributi interni di Luogo e un altro che accetta un oggetto Luogo. Questo secondo metodo confronta il Luogo con la sua lista interna. Se troverà un riscontro restituirà il riferimento interno alla lista, altrimenti aggiungerà il puntatore ricevuto al suo elenco e restituirà lo stesso.

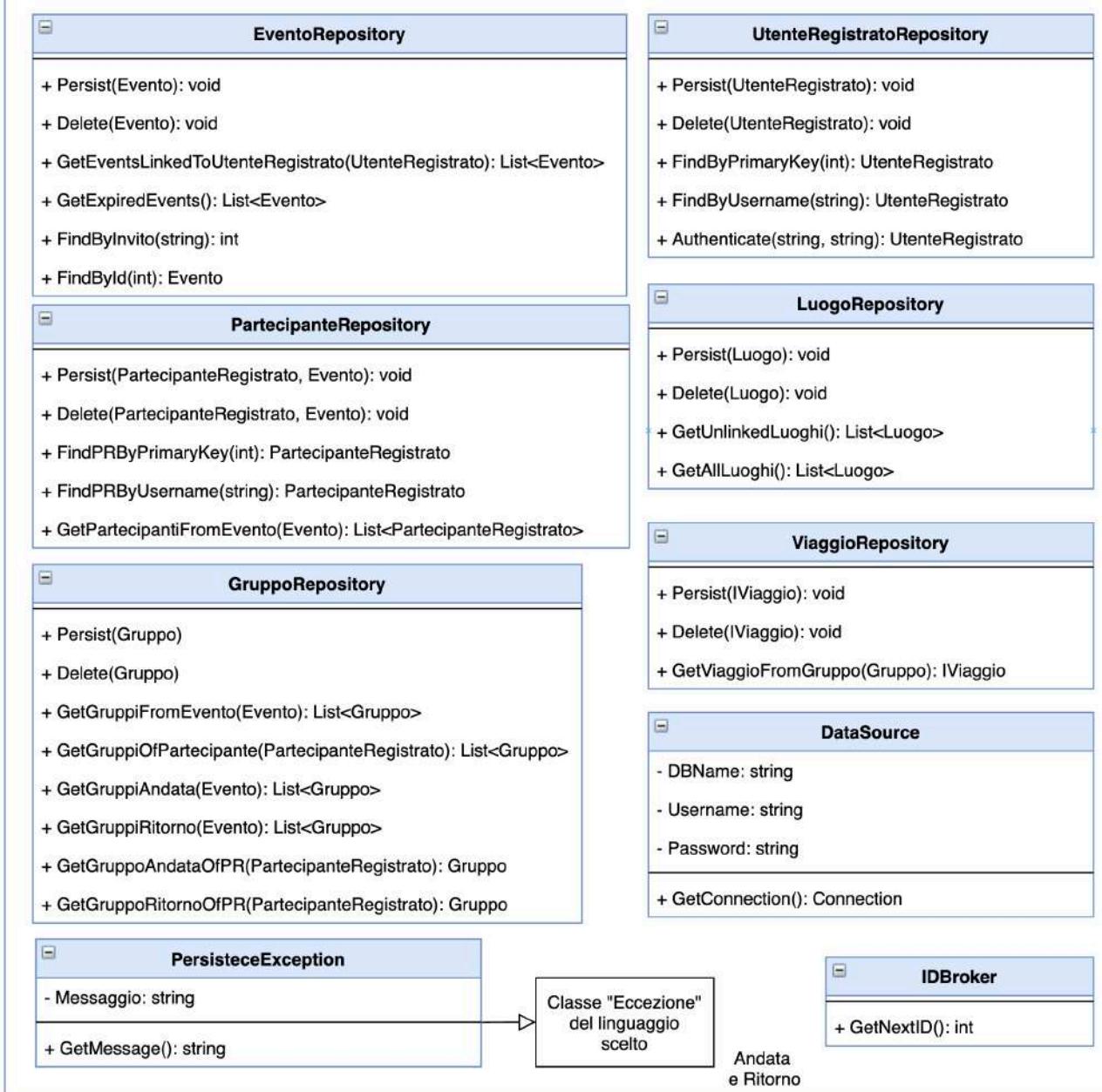


Qui sopra è illustrata la porzione di dominio che riguarda i gruppi e l'interfacciamento con l'oggetto esterno Viaggio.

La classe Gruppo ha un attributo enumerativo AndataORitorno che indica se si sta riferendo al viaggio di andata o al viaggio di ritorno.

Non ha attributi di tipo ILuogo poiché li può ottenere grazie ai riferimenti con PartecipanteRegistrato ed Evento.

Per rispettare la metodologia del Design for Change abbiamo previsto la possibilità in futuro di richiedere le indicazioni stradali ad un altro provider di mappe e percorsi (cioè non più GoogleMaps e le sue API). Questo grazie all'implementazione di un'interfaccia che si interpone tra la nostra classe (che interagisce con il provider) e l'utilizzatore del risultato del calcolo.



Il Model avrà le classi di dominio che interagiranno con il database attraverso le classi Repository.

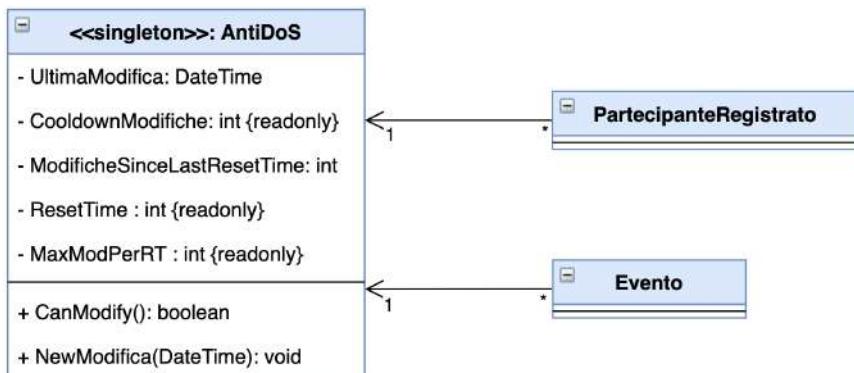
La classe `IDBroker` sarà interpellata alla creazione di ogni oggetto e restituirà un ID univoco ottenuto grazie ad una Sequence del DBMS.

`DataSource` è l'oggetto che tutte le classi Repository dovranno interpellare per ottenere un riferimento alla connessione verso il database.

`PersistenceException` è l'interazione che verrà lanciata ogni volta che ci sarà un problema di interazione con il database.

## Pattern SINGLETON

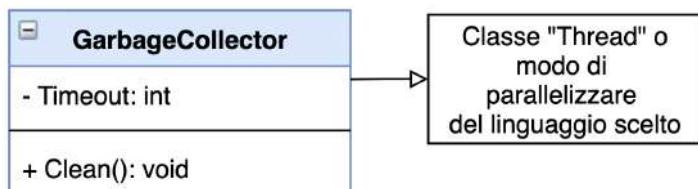
Package: CarpHaul.ServerGestore.Model.Dominio



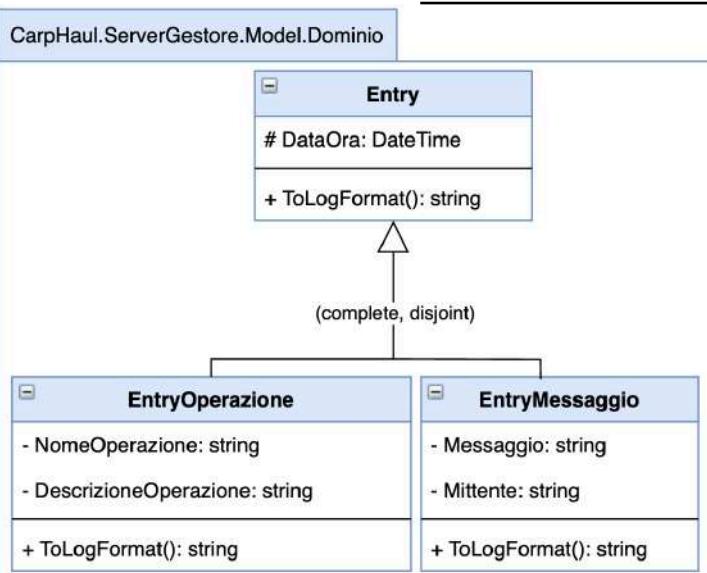
L'oggetto AntiDoS è concepito come Singleton e ha la funzionalità interna di un piccolo AntiDoS locale poiché la funzionalità generale di mitigazione del Denial of Service è delegata al provider dei server dove verrà eseguito il deploy del back-end della nostra applicazione.

Esso funzionerà controllando e bloccando la possibilità di modificare i dati di un evento o di un partecipante dopo un numero di richieste eccessivo in un lasso di tempo prestabilito. Essendo un singleton, la richiesta di questo controllo potrà essere facilmente estesa in futuro a modifiche o richieste di altro genere.

Package: CarpHaul.ServerGestore.Model.Dominio



La classe GarbageCollector si occuperà di controllare a intervalli di tempo regolari la presenza di eventi passati e luoghi senza alcun collegamento e di cancellarli dal database. Questa classe lavorerà in parallelo rispetto al resto poiché sarà un thread o un processo pesante parallelo a seconda del linguaggio di programmazione scelto.

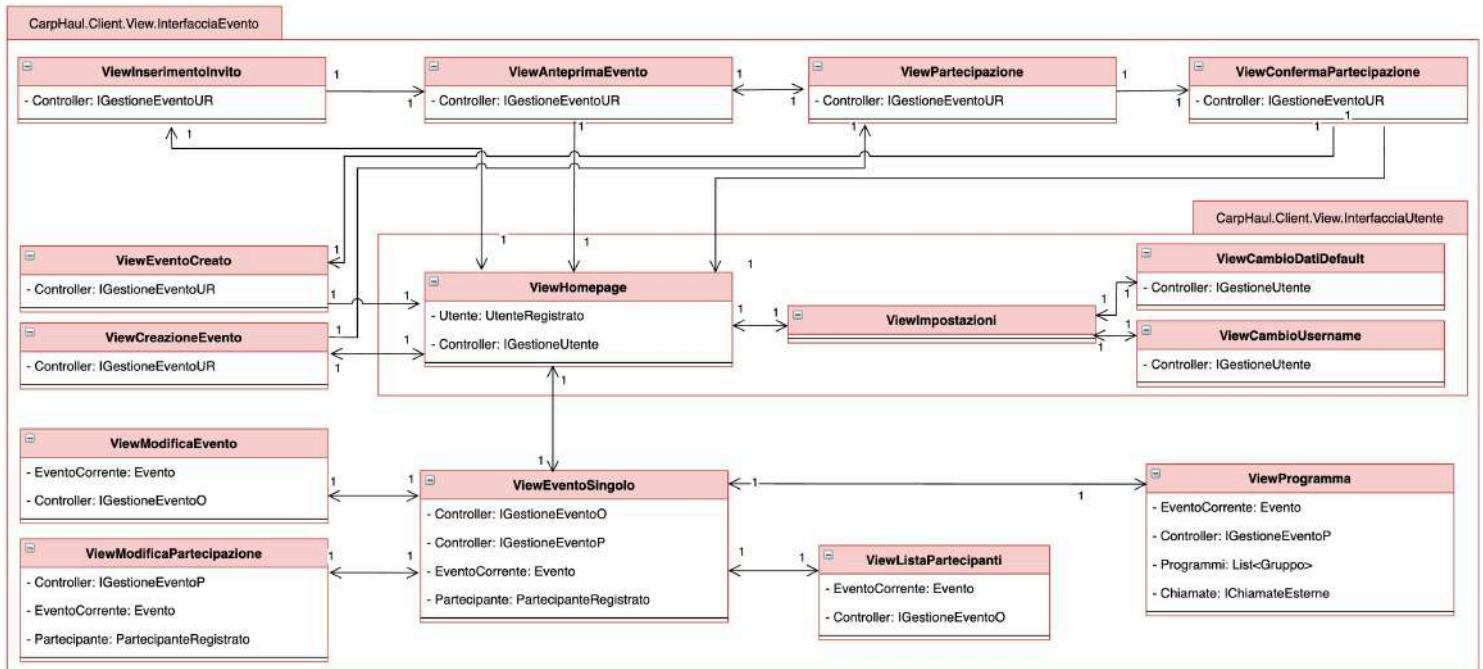


La porzione di Log del dominio verrà utilizzata solamente dai sotto controller del presenter. Un controller ad hoc per i log penserà a scriverli su file.

## Struttura Client

La struttura del client è basata su MVC:

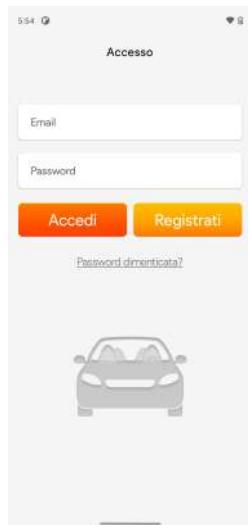
- Model: Elaborerà i dati per la ricezione e l'invio verso il Server Gestore.
- View: Sarà il componente principale del client e provvederà ad interfacciarsi con l'utente in maniera intuitiva ed efficiente.
- Controller: sarà il responsabile della gestione degli input della vista.



Ad ogni View presente nel diagramma qui sopra corrisponde una maschera (vedere più avanti).

Associata ad ognuna delle View vi è anche un controller dal nome riconducibile. Benché nel diagramma non siano presenti le connessioni grafiche (frecce) tra le View e i controller, esse sono esplicitate da riferimenti presenti negli attributi delle View stesse.

# Associazione View - Maschere



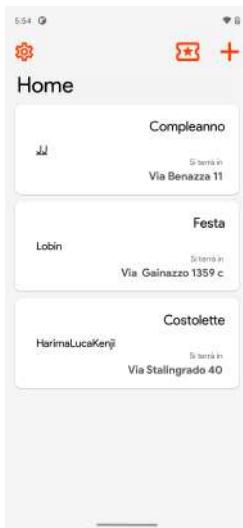
ViewLogin



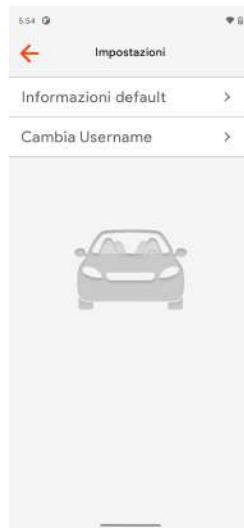
ViewRegistrazione



ViewRecovery



ViewHomepage



ViewImpostazioni



ViewCambioUsername



ViewCambioDatiDefault



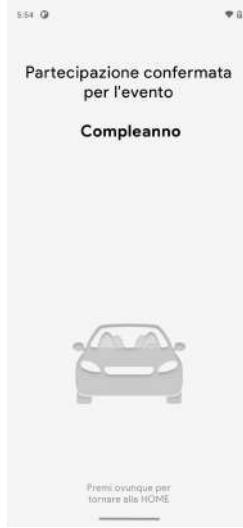
ViewInserimentoInvito



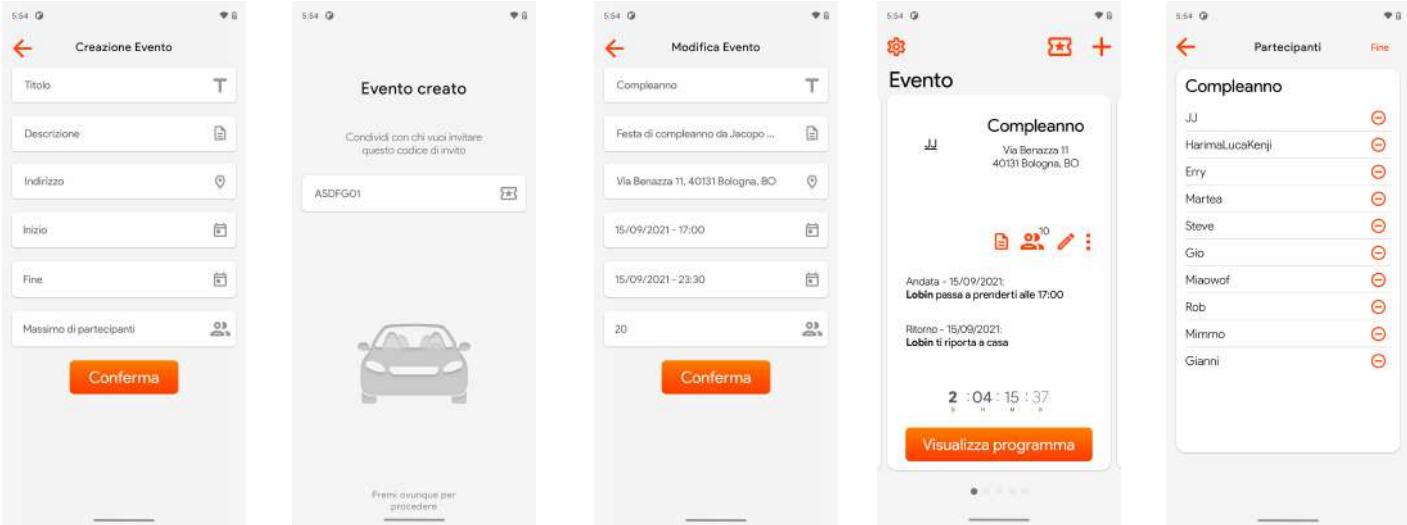
ViewAnteprimaEvento



ViewPartecipazione



ViewConfermaPartecipazione

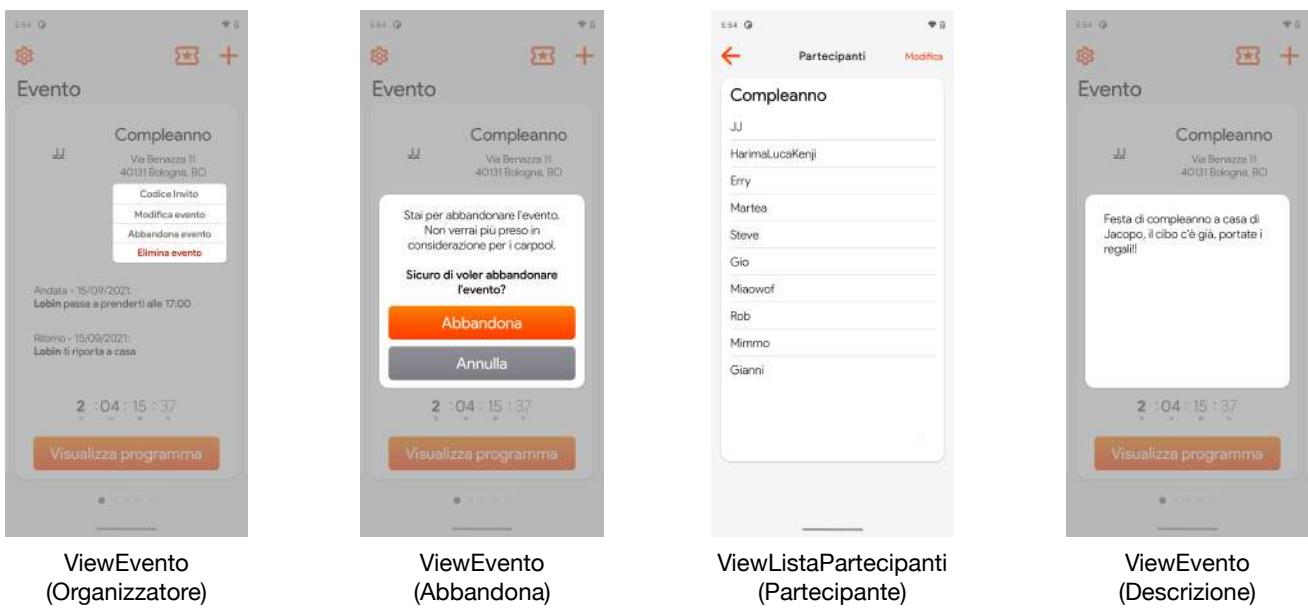
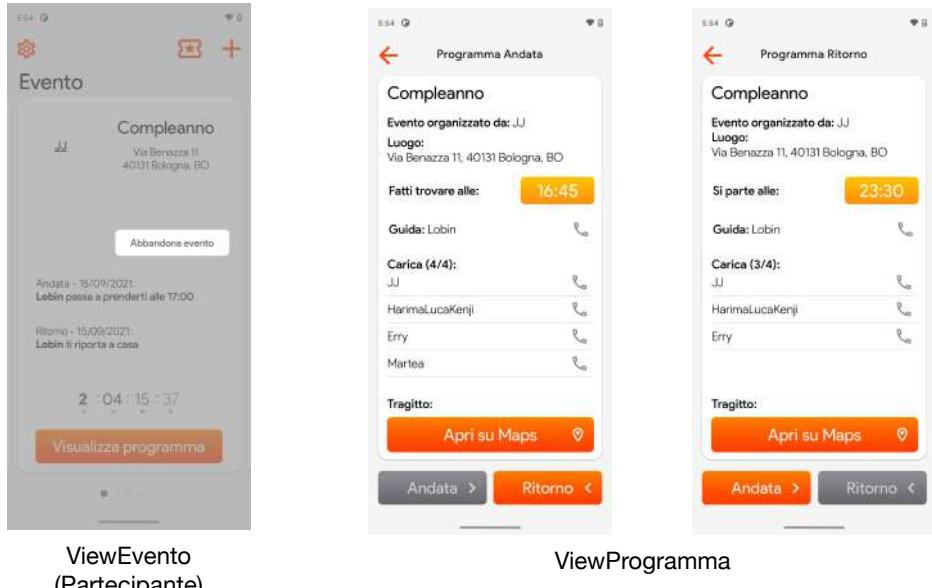


ViewCreazioneEvento

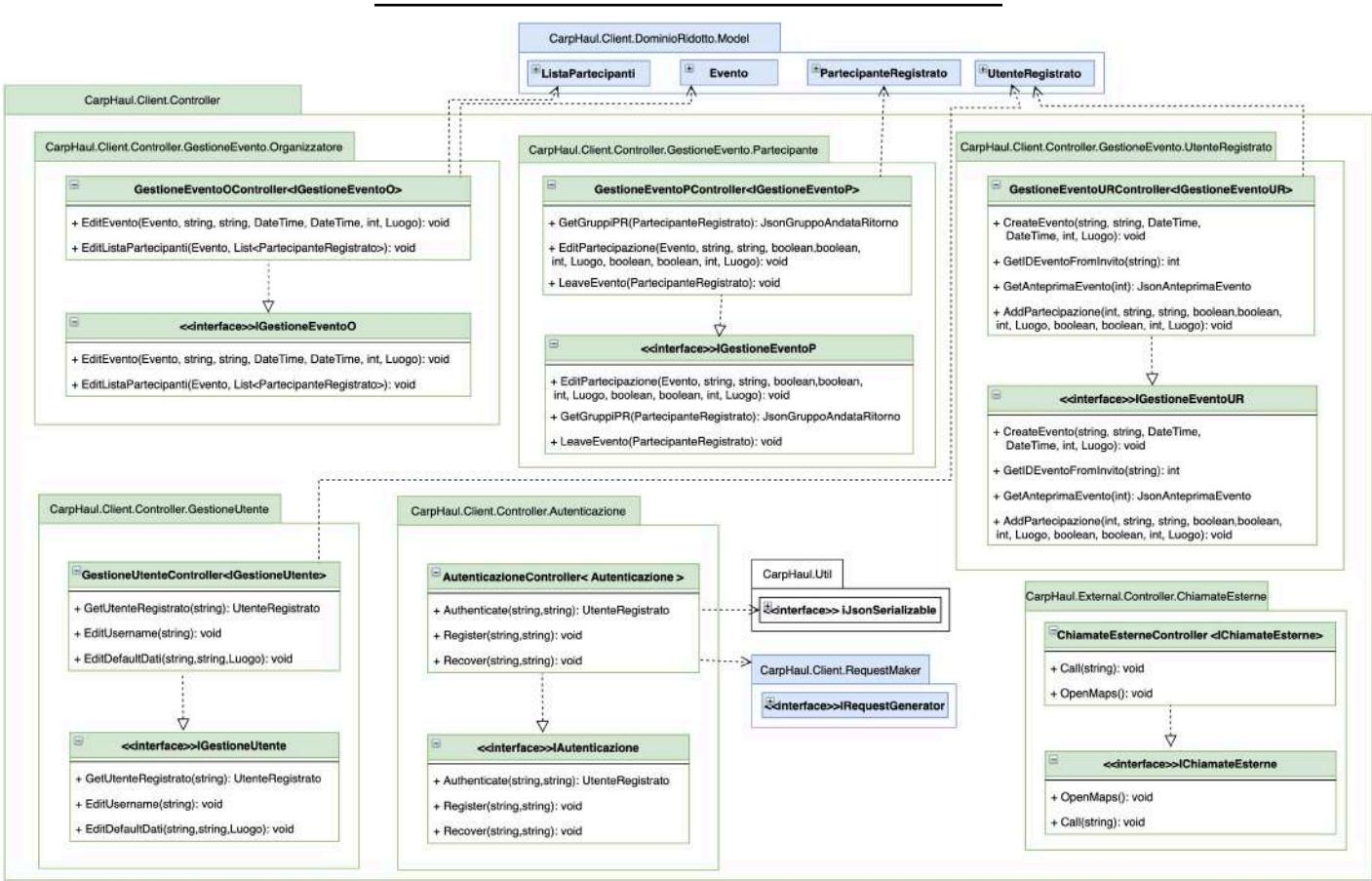
ViewEventoCreateo

ViewModificaEvento

ViewEventoSingolo

ViewListaPartecipanti  
(Organizzatore)ViewEvento  
(Organizzatore)ViewEvento  
(Abbandona)ViewListaPartecipanti  
(Partecipante)ViewEvento  
(Descrizione)ViewEvento  
(Partecipante)

ViewProgramma

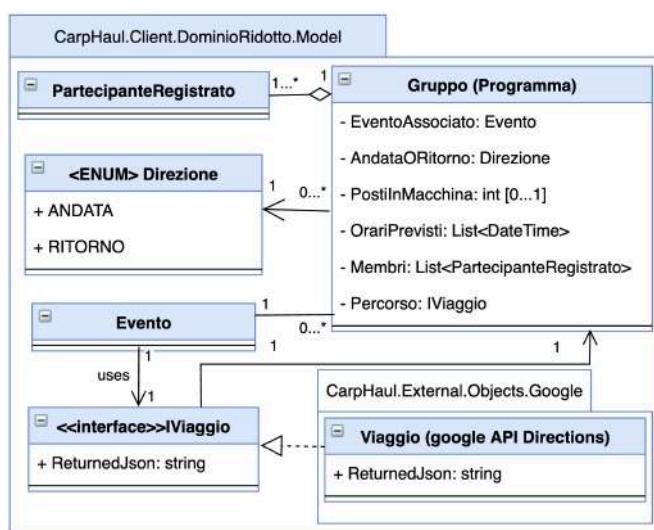
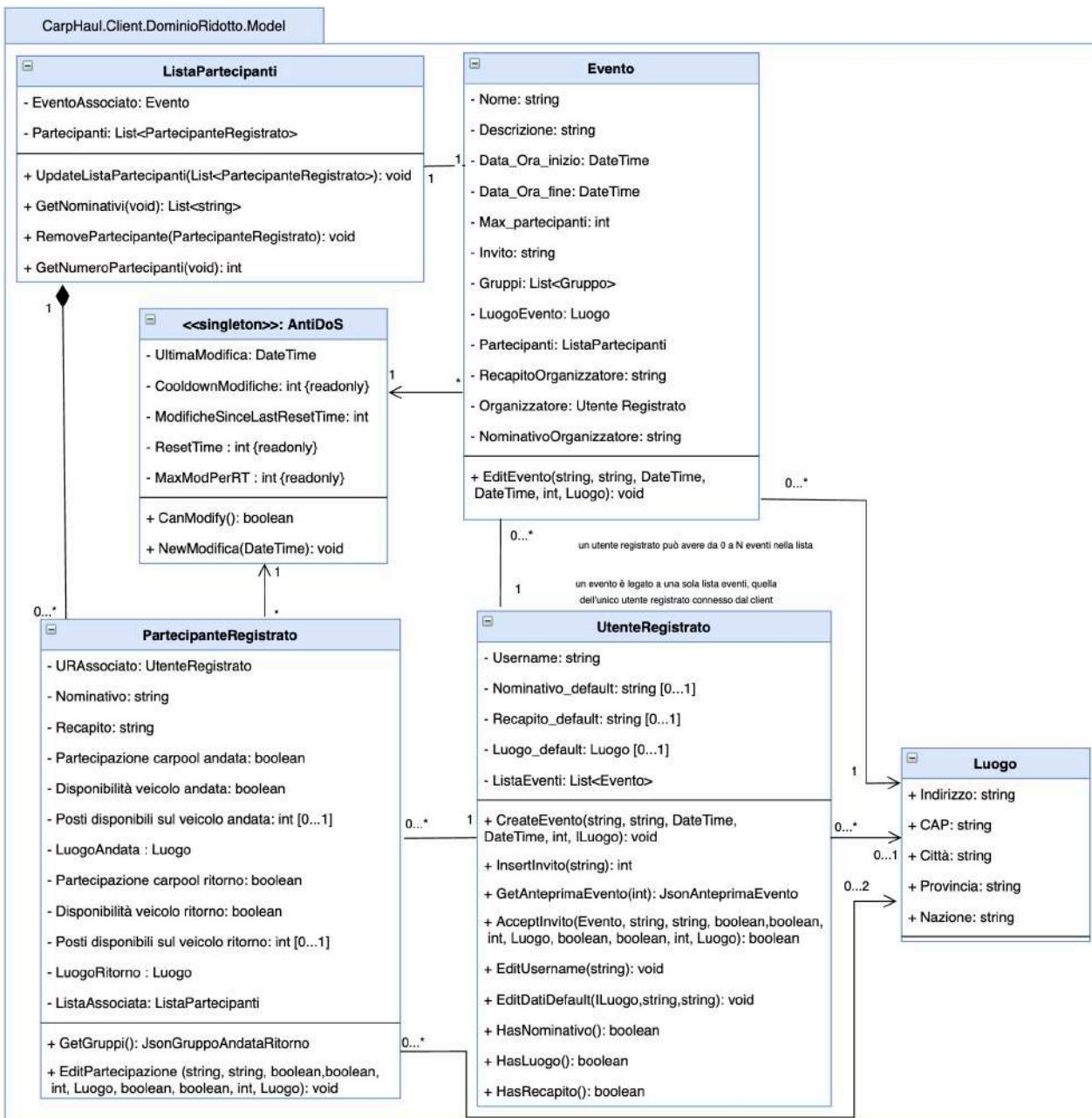


Quasi tutti i controller, per interagire con il server, faranno richieste alla rispettiva entità del model per il tipo di risorsa desiderato.

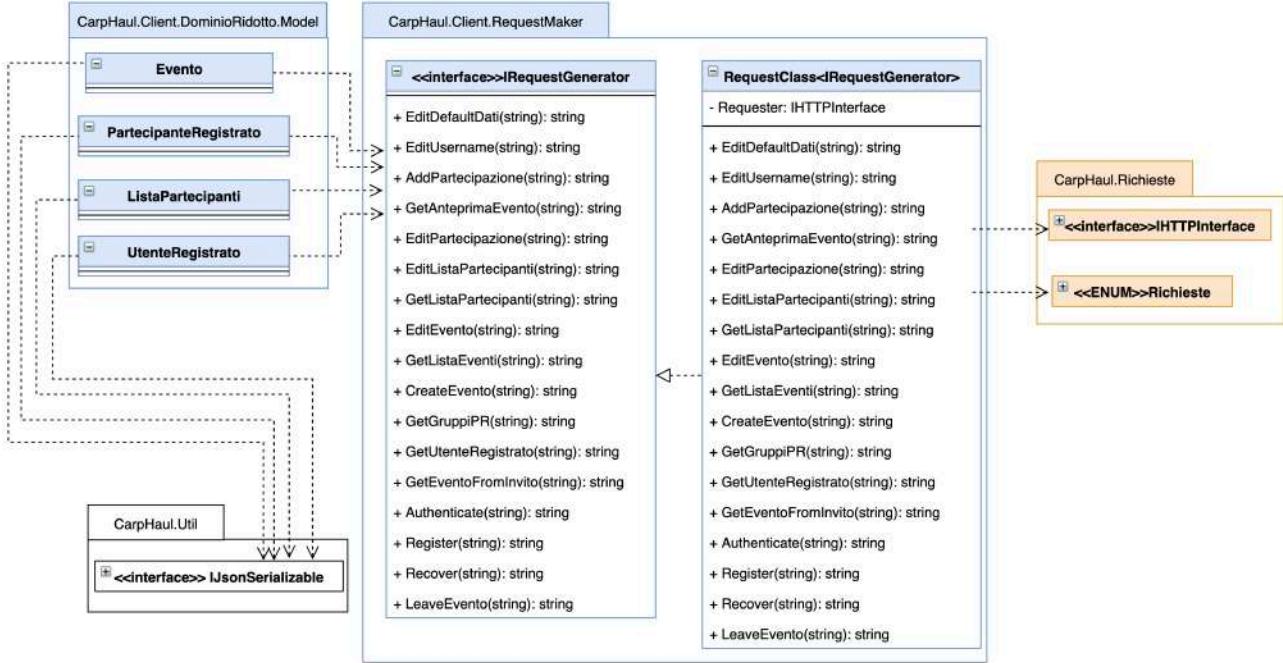
L'unica eccezione risiede nel controller Autenticazione, il quale in autonomia richiederà, attraverso l'interfaccia IRequestGenerator, al server l'azione voluta.

Questo perchè nessuna classe del dominio può essere associata al comportamento che questo package espone.

Come accennato in precedenza la logica di business è praticamente assente visto che tutto il suo svolgimento è delegato al ServerGestore



Riscontriamo delle differenze tra il dominio del Server Gestore e quello del Client poiché quest'ultimo è stato riadattato per le minori esigenze dell'applicazione. Dentro ad alcune classi del dominio sono presenti dei metodi che consentono, tramite un'opportuna interfaccia esposta in seguito, la comunicazione verso il Server Gestore.

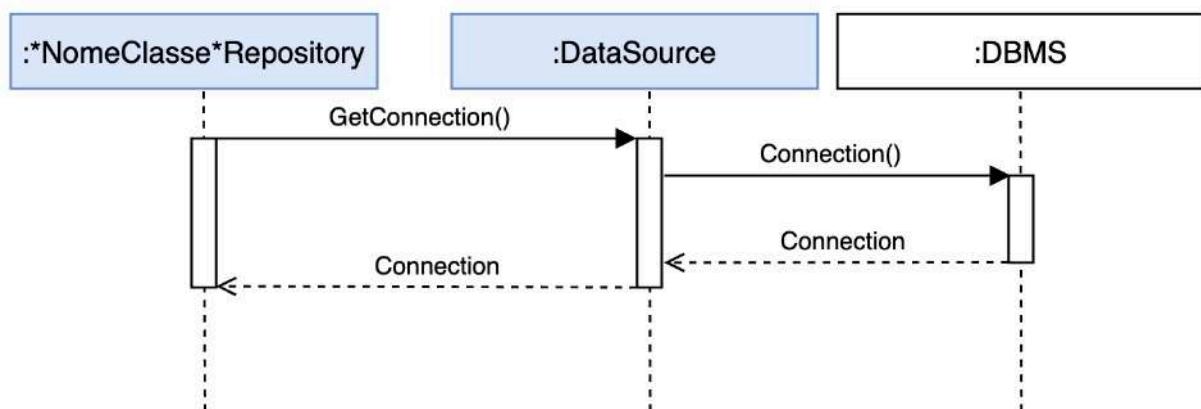


L'interfaccia **IRequestGenerator** espone tutti i possibili servizi che il Client può richiedere al Server Gestore. Saranno le classi del dominio stesso ad invocare l'opportuna funzione passandole i dati in formato Json.

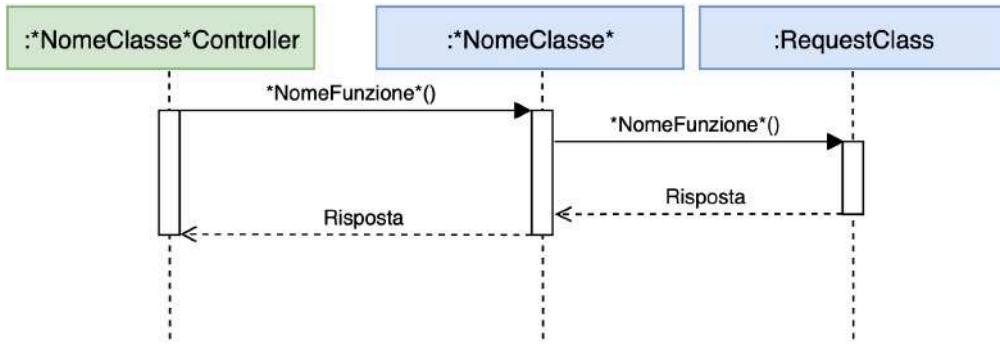
Nell'implementazione di questa interfaccia si utilizza il protocollo HTTP. Questa scelta può essere cambiata facilmente (ad esempio sostituendola con l'utilizzo di WebSocket) scrivendo una nuova classe che implementi l'interfaccia **IRequestGenerator**.

## Interazione

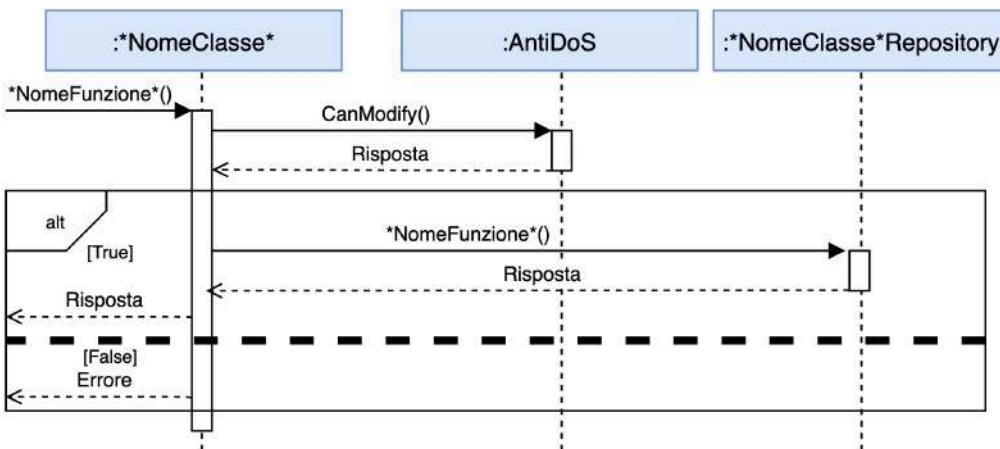
Cominciamo a illustrare l'interazione descrivendo il comportamento che avranno tutte le classi che vorranno interagire con il database. Non sarà perciò riportato nei diagrammi successivi questo comportamento, ma nella realtà questo passaggio dovrà essere sempre eseguito.



Ogni volta che il Client vorrà inviare delle richieste al Server Gestore, dovrà sempre passare per la classe del dominio competente, la quale provvederà a chiamare la RequestClass per l'invio. Ciò è riportato nel seguente diagramma e sarà omesso in quelli successivi.

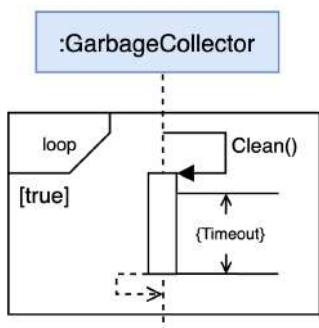


Per ogni tentativo di modifica a qualsiasi dato di un Evento (Partecipanti, informazioni evento ecc...) verrà preventivamente eseguito un controllo per evitare attacchi DoS. Questo controllo verrà eseguito allo stesso modo sia su Client che su Server in modo da prevenire sia attacchi di singoli che attacchi di gruppo. Anche in questo caso ciò che è riportato nel seguente diagramma verrà sottinteso in seguito



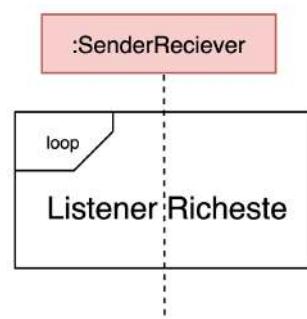
Di seguito verranno riportati i diagrammi di sequenza di alcune operazioni

**GarbageCollector**



Itera in autonomia a intervalli di tempo "Timeout" facendo fetch da database e cancellando istanze di Evento passate e istanze di Luogo senza alcun riferimento

**SenderReceiver**



Listener in attesa di richieste dai Client che all'arrivo di ogni messaggio invoca in maniera asincrona la funzione HandleRequest() (vedere in seguito)

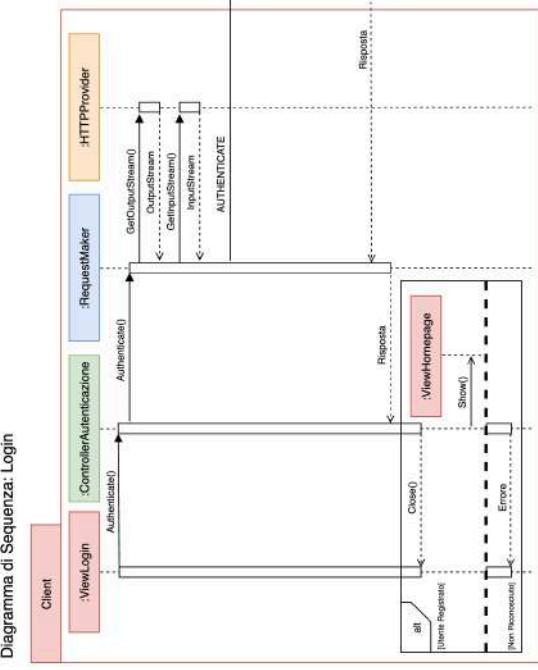


Diagramma di Sequenza: Login

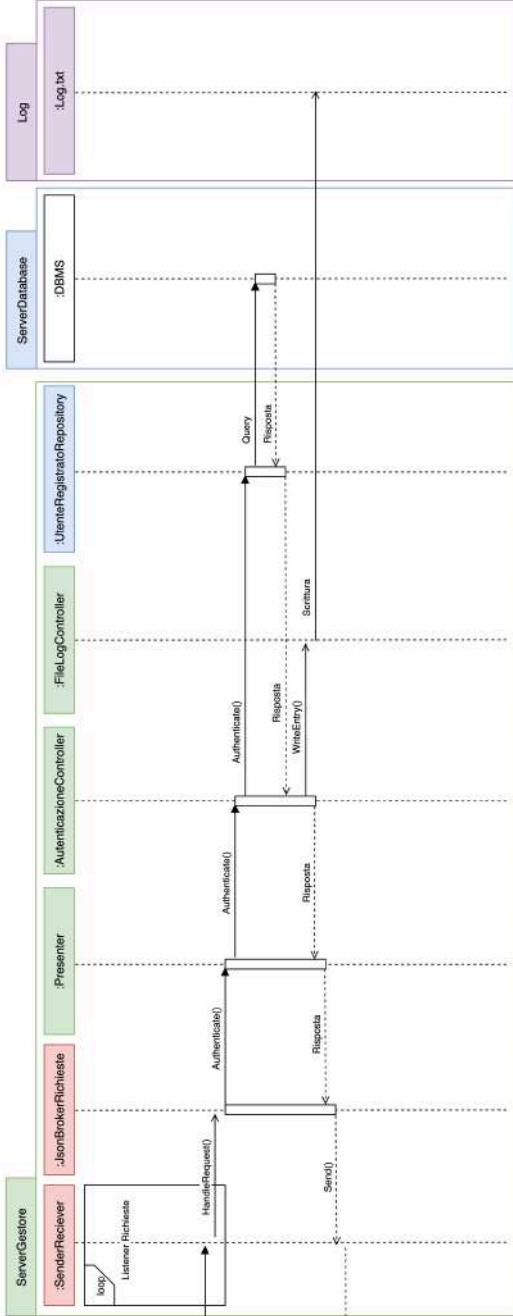


Diagramma di Sequenza: Registrazione

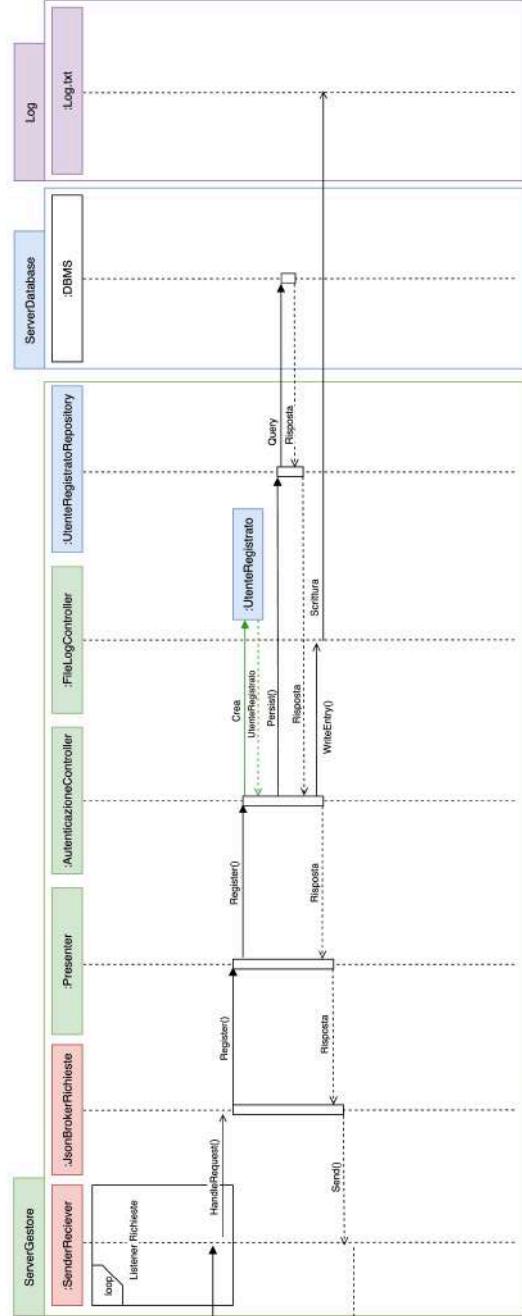


Diagramma di Sequenza: Inserimento Invito Válido

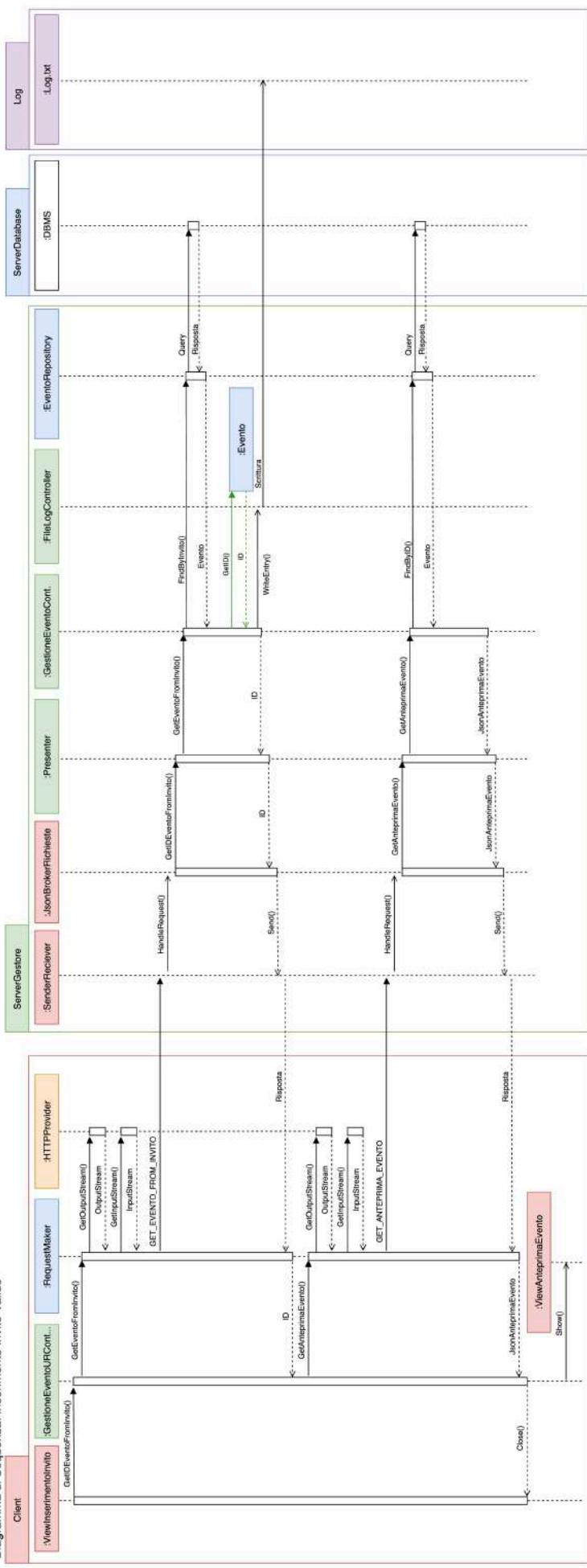


Diagramma di Sequenza: Modifica Partecipazione

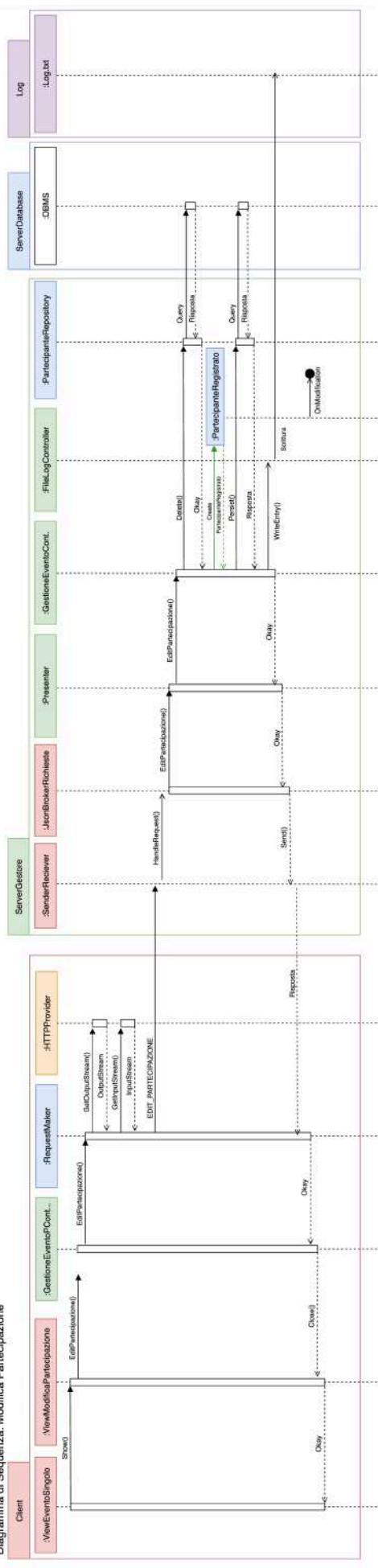


Diagramma di Sequenza: AbbandonaEvento

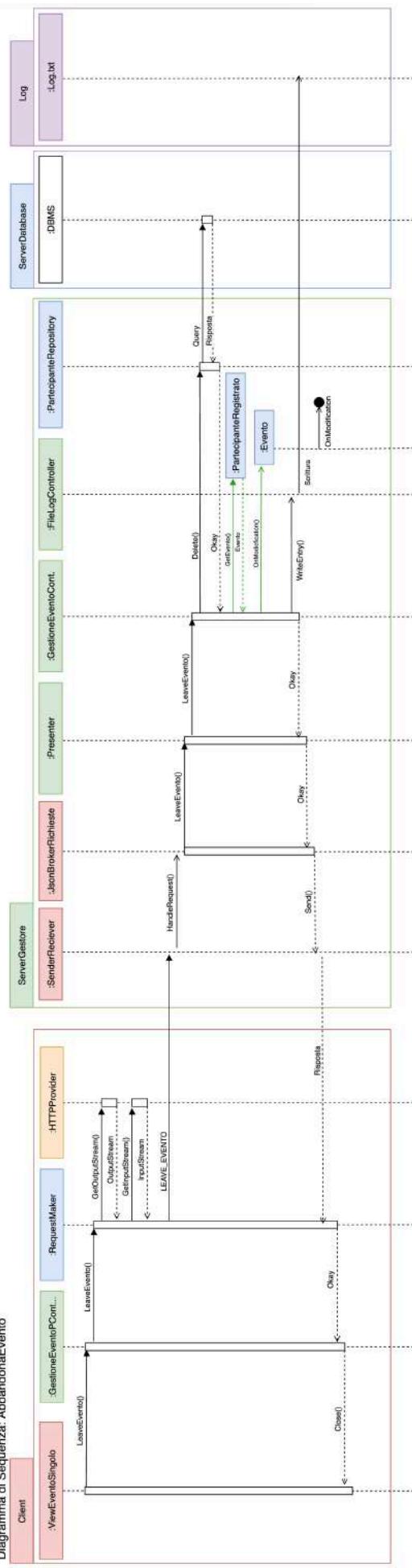


Diagramma di Sequenza: OnModification

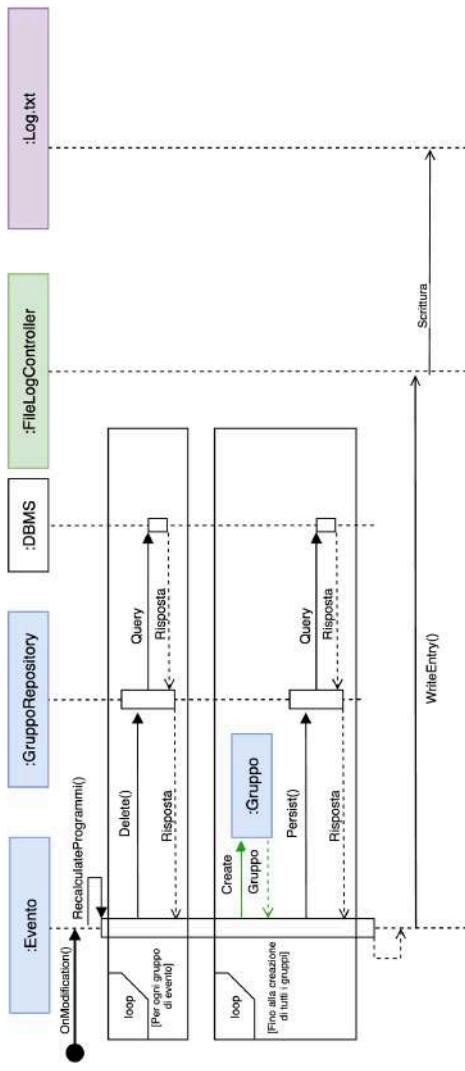


Diagramma di Sequenza: Aggiunta Partecipazione

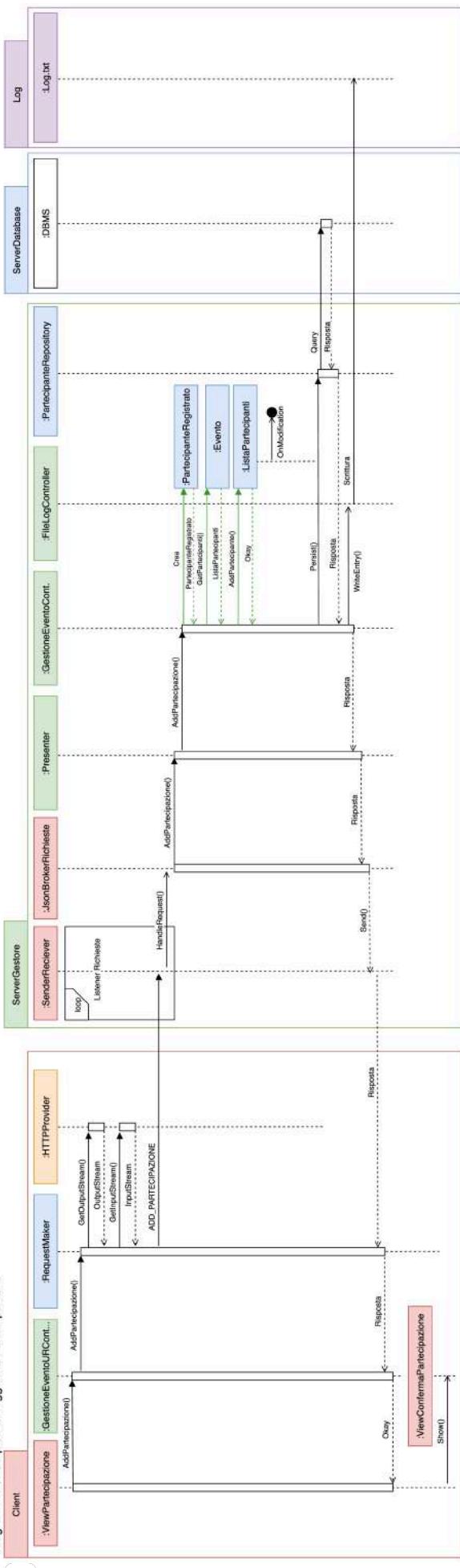
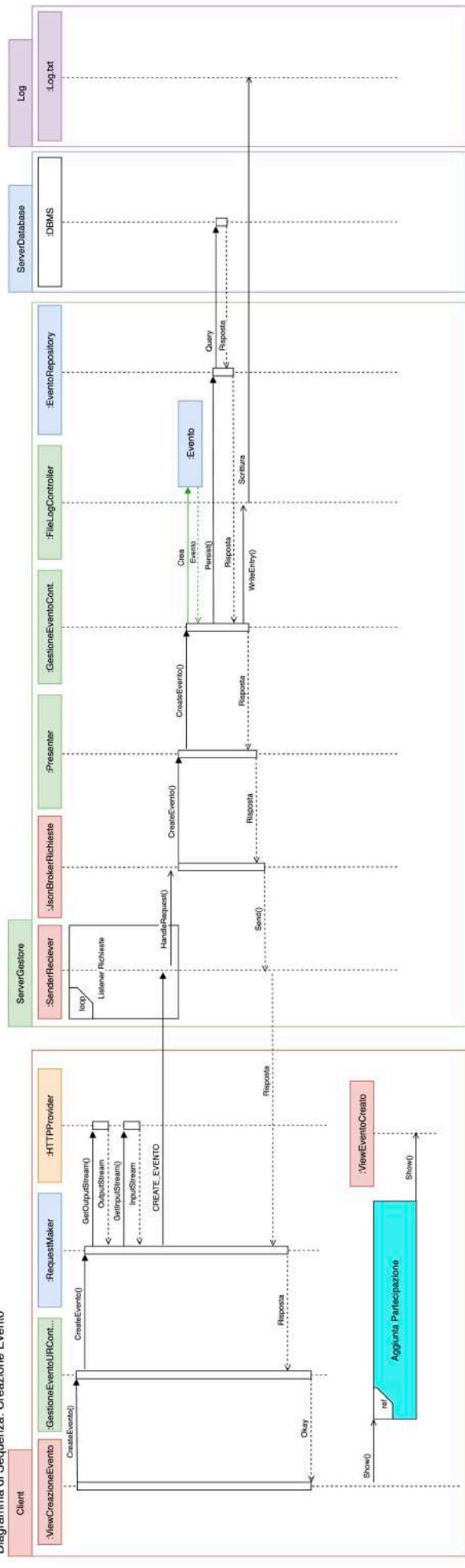


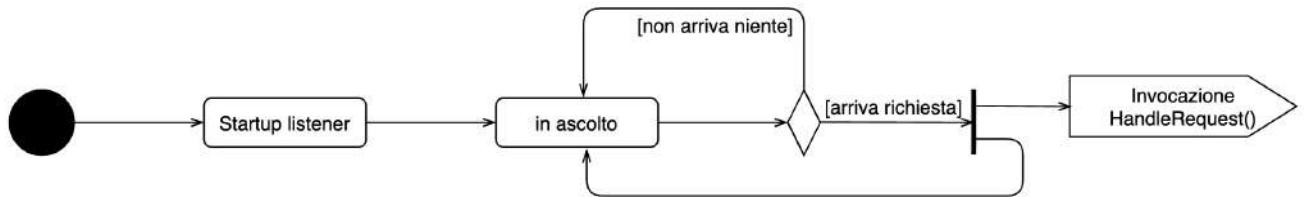
Diagramma di Sequenza: Creazione Evento



## Comportamento

Riportiamo qui solo i diagrammi delle attività necessari a comprendere il funzionamento di elementi aggiunti in progettazione.

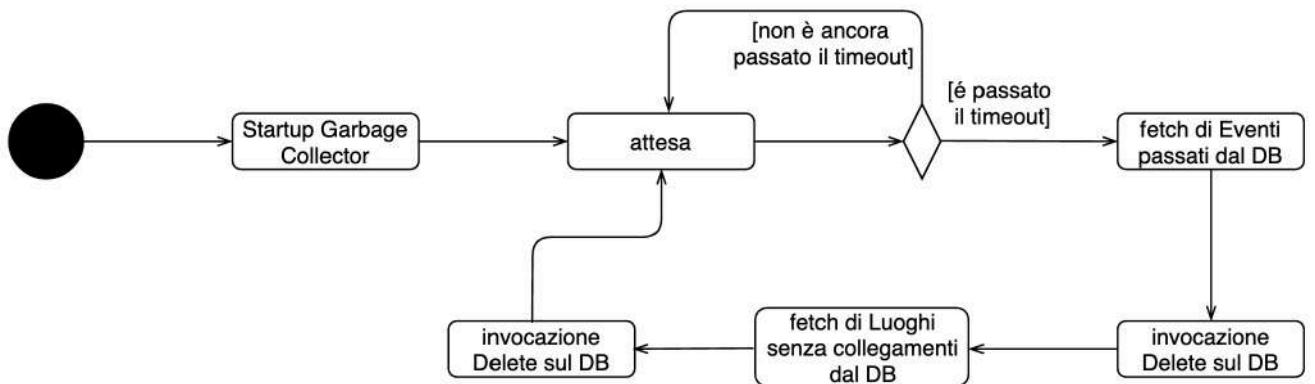
### ServerReceiver:



### JsonBrokerRichieste:



### GarbageCollector:



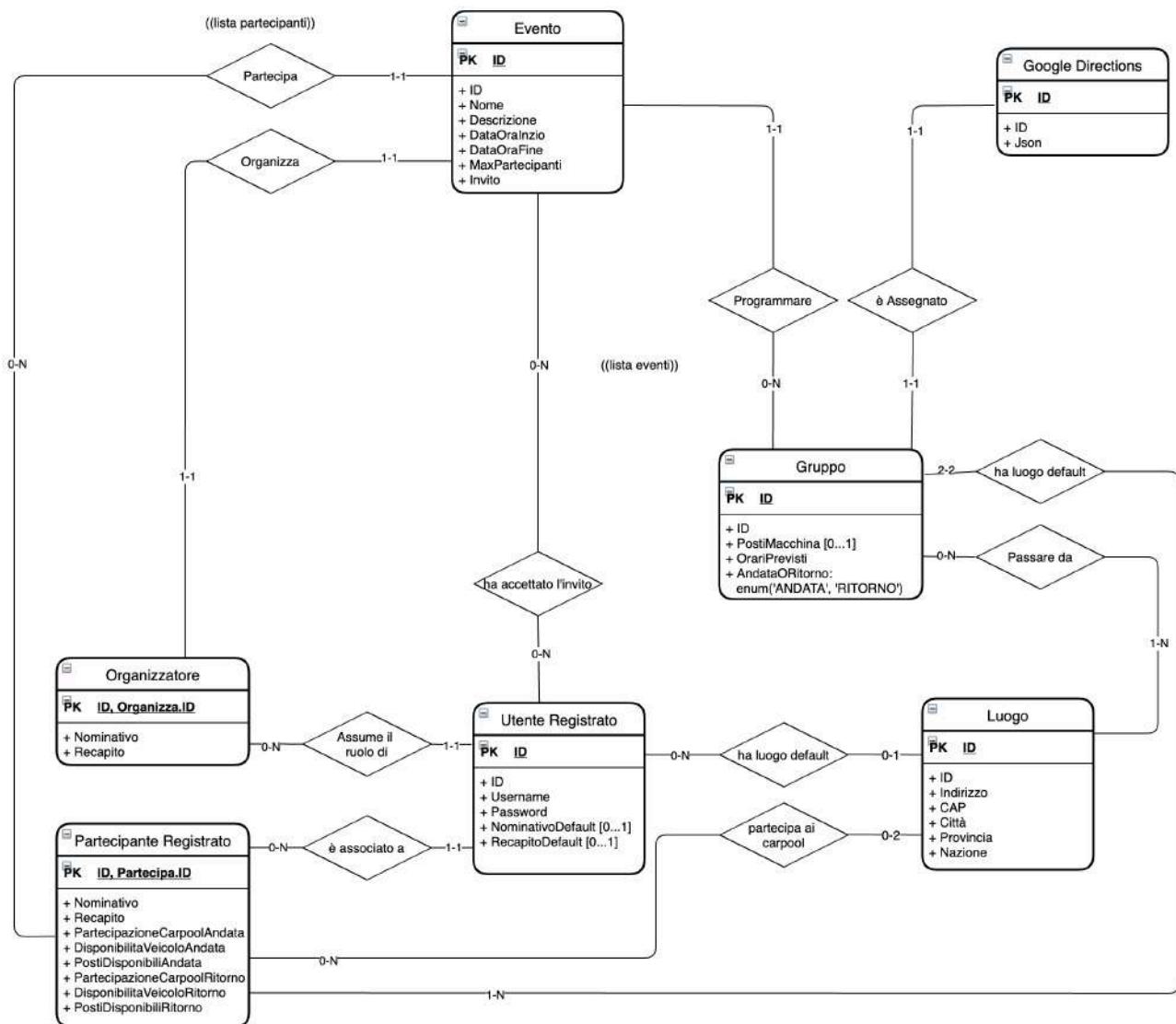
# Progettazione della Persistenza

È necessario l'utilizzo di un ServerDatabase e di un DBMS dove sarà possibile conservare e manipolare i dati necessari al corretto funzionamento dell'applicazione. Inoltre è previsto che questo risieda in una macchina diversa rispetto al Server Gestore e che il suo indirizzo IP sia conosciuto solo da quest'ultimo.

Nel ServerDatabase ci saranno vincoli di accesso ristretti ad una whitelist di IP circoscritti solo all'indirizzo del server gestore (o di più indirizzi in caso di replicazioni della macchina in futuro). Così si mitigherà inoltre la possibilità di attacco.

Saranno previsti in ogni caso servizi di backup ad intervalli di tempo regolari.

## Diagramma E-R



L'entità Utente Registrato ha come chiave primaria l'attributo ID (ID surrogato) e lo Username deve essere unico; la password deve essere codificata con l'algoritmo hash SHA-256. Gli altri attributi invece possono essere Null.

Evento non può possedere campi Null e il MaxPartecipanti avrà un valore di default modificabile.

Luogo non può possedere campi Null.

Organizzatore non può possedere campi Null.

Gruppo non può possedere campi Null.

L'attributo Json dell'entità Google Directions rappresenta l'intero Json restituito dalle API di GoogleMaps.

Il Partecipante Registrato ha i seguenti vincoli sugli attributi:

Se PartecipazioneCarpoolAndata è false, allora DisponibilitaVeicoloAndata,

PostiDisponibiliAndata e l'associazione a Luogo (per l'andata) saranno Null,

Se PartecipazioneCarpoolAndata è true, allora

l'associazione con Luogo non potrà essere Null

se DisponibilitaVeicoloAndata è false, PostiDisponibiliAndata sarà Null

se DisponibilitaVeicoloAndata è true, PostiDisponibiliAndata sarà un intero positivo

E rispettivamente ciò varrà per i campi del ritorno.

## Log

Per la persistenza dei file di log verrà utilizzato un file di testo Log.txt risiedente su ServerGestore sul quale sarà effettuata un'operazione append per ogni chiamata al metodo WriteEntry().

Il messaggio sarà generato con un formato ad hoc a seconda del tipo di Entry.

## Formato File Log

Il file di log contiene un report delle azioni svolte da ServerGestore riportando anche data e ora. Il formato dei log è il seguente

[Data - Ora] Messaggio Personalizzato della Entry

## Progettazione del Collaudo

Nei seguenti test ogni volta che ci si riferisce ad un campo di una classe nella forma \*NomeOggetto\*.\*NomeAttributo\* , in realtà non si intende un accesso diretto all'attributo ma una richiesta di accesso a proprietà, in quanto si sta usando standard C#. Gli attributi all'interno delle classi saranno infatti chiamati con la seguente forma: "\_nomeAttributo"

```

1 //Test Client.Controller.GestioneEvento.Organizzatore
2 [TestFixture]
3 public class TestGestioneEventoController
4 {
5
6     private List<PartecipanteRegistrato> _prList;
7     private Evento _e;
8
9     [SetUp]
10    public void ControllerSetup() {
11        _e = new Evento ("Festa di bocciatura", "Bottom text", new
12        .
13        DateTime(2021,6,11,15,0,0), new DateTime(2021,6,11,18,0,0), 100, new Luogo
14        .
15        ("Via S. Ruffillo 45", "40141", "Bologna", "BO","Italia") );
16        _prList = new List<PartecipanteRegistrato>;
17        for(int i=0; i<10; i++) {
18            _prList += new PartecipanteRegistrato(_e, "partecipante "+i,
19            .
20            "56232178965", false, false, 0, null, false, false, 0, null);
21        }
22    }
23
24    [Test]
25    public void TestMethod() {
26
27        Assert.That(_e.Nome, Is.EqualTo("Festa di bocciatura"));
28        Assert.That(_e.Data_Ora_inizio, Is.EqualTo(new DateTime(2021, 6, 11, 15, 0,
29        .
30        0)));
31        Assert.That(_e.Data_Ora_fine, Is.EqualTo(new DateTime(2021, 6, 11, 18, 0,
32        .
33        0)));
34        Assert.That(_e.Partecipanti.GetNumeroPartecipanti(), Is.EqualTo(10));
35        for (int i = 0; i < 10; i++) {
36            Assert.That(_e.Partecipanti.GetNominativi[i], Is.EqualTo("partecipante
37            .
38            "+i));
39        }
40
41        EditEvento(_e, "Festa di super bocciatura", "Bottom text", new DateTime(2021,
42        .
43        6, 10, 15, 0, 0), new DateTime(2021, 6, 10, 18, 0, 0), 100, new Luogo("Via S.
44        .
45        Ruffillo 45", "40141", "Bologna", "BO", "Italia"));
46        Assert.That(_e.Nome, Is.EqualTo("Festa di super bocciatura"));
47        Assert.That(_e.Data_Ora_inizio, Is.EqualTo(new DateTime(2021, 6, 10, 15, 0,
48        .
49        0)));
50        Assert.That(_e.Data_Ora_fine, Is.EqualTo(new DateTime(2021, 6, 10, 18, 0,
51        .
52        0)));
53
54        _prList.RemoveAt(2);
55        _prList.RemoveAt(7);
56        EditListaPartecipanti(_e, _prList);
57        Assert.That(_e.Partecipanti.GetNumeroPartecipanti(), Is.EqualTo(8));
58        for (int i = 0; i < 10, i++) {
59            if (i == 2 || i == 7)
60                continue;
61            Assert.That(_e.Partecipanti.GetNominativi[i], Is.EqualTo("partecipante "
62            .
63            + i));
64        }
65    }
66}

```

```

1 //Test Client.Controller.GestioneEvento.UtenteRegistrato
2 [TestFixture]
3 public class TestGestioneEventoURController
4 {
5
6     private UtenteRegistrato _ur;
7
8     [SetUp]
9     public void UtenteRegistratoSetUp() {
10         _ur = new UtenteRegistrato("Alice@alice.it","password");
11         Luogo _lEvento = new Luogo ("Via S. Ruffillo 45", "40141", "Bologna", "BO","Italia");
12     }
13
14     public string GetEvento( ){
15         List<Evento> _eventi = _ur.GetListaEventi();
16         foreach(Evento e in _eventi){
17             if(e.GetName() == "Festa di bocciatura" &&
18                 e.GetLuogo() == _lEvento &&
19                 e.GetData_Ora_inizio() == new DateTime(2021,6,11,15,0,0))
20                 return e;
21         }
22         return null;
23     }
24
25     [Test]
26     public void TestMethod() {
27         CreateEvento("Festa di bocciatura", "Bottom text", new DateTime(2021,6,11,15,0,0),
28         new DateTime(2021,6,11,18,0,0), 100,_lEvento );
29
30         string invito = GetEvento().Invito;
31
32         JsonAnteprimaEvento _ant_ev = GetAnteprimaEvento(GetIdFromInvito(invito));
33         Assert.That(_ant_ev, Is.Not.EqualTo (null));
34         Assert.That(_ant_ev.GetName(), Is.EqualTo("Festa di bocciatura"));
35         Assert.That(_ant_ev.GetDescrizione(), Is.EqualTo("avvenimento usuale dopo ogni
36         appello di ingegneria del software"));
37         Assert.That(_ant_ev.GetLuogo().GetIndirizzo(), Is.EqualTo("Via S. Ruffillo 45"));
38         Assert.That(_ant_ev.GetLuogo().GetCAP(), Is.EqualTo("40141"));
39         Assert.That(_ant_ev.GetLuogo().GetCitta(), Is.EqualTo("Bologna"));
40         Assert.That(_ant_ev.GetLuogo().GetProvincia(), Is.EqualTo("BO"));
41         Assert.That(_ant_ev.GetLuogo().GetNazione(), Is.EqualTo("Italia"));
42         Assert.That(_ant_ev.GetData_Ora_inizio(), Is.EqualTo(new DateTime(2021,6,11,15,0,0)));
43         Assert.That(_ant_ev.GetData_Ora_fine(), Is.EqualTo(new DateTime(2021,6,11,18,0,0)));
44         Assert.That(_ant_ev.GetOrganizzatore(), Is.EqualTo(_ur));
45         Assert.That(_ant_ev.GetListaNominativi().Count, Is.EqualTo(0)); //è vuota perchè non
46         .abbiamo inserito nessun partecipante
47
48         //add Partecipazione
49         AddPartecipazione(_ant_ev.id,"Jerry", "56232178965", false, false, 0, null, false,
50         false, 0, null);
51         Evento _evento = GetEvento();
52
53         Assert.That(_evento.GetListaPartecipanti().Count, Is.EqualTo(1));
54     }
55 }
```

```

1 //Test Client.Controller.GestioneEvento.Partecipante
2 [TestFixture]
3 public class TestGestioneEventoPController
4 {
5
6     private PartecipanteRegistrato _pr;
7     private Evento _e;
8
9     [SetUp]
10    public void PartecipanteRegistratoSetUp() {
11        _e = new Evento ("Festa di bocciatura", "Bottom text", new
12        . DateTime(2021,6,11,15,0,0), new DateTime(2021,6,11,18,0,0), 100, new Luogo
13        . ("Via S. Ruffillo 45", "40141", "Bologna", "BO", "Italia" ) );
14        _pr = new PartecipanteRegistrato(_e, "Jerry", "56232178965", false, false, 0,
15        . null, false, false, 0, null);
16    }
17
18    [Test]
19    public void TestMethod() {
20        Assert.That(_pr.Nominativo, Is.EqualTo("Jerry"));
21        _pr.EditPartecipazione(_e, 'Tom', "56232178965", false, false, 0, null,
22        . false, false, 0, null);
23        Assert.That(_pr.Nominativo, Is.EqualTo("Tom"));
24
25        JsonGruppoAndataRitorno andataRitorno = GetGruppiPR(_pr);
26        Assert.That(andataRitorno.Andata.AndataORitorno,
27        . Is.EqualTo(Direzione.ANDATA));
28        Assert.That(andataRitorno.Ritorno.AndataORitorno,
29        . Is.EqualTo(Direzione.RITORNO));
30        Assert.That(andataRitorno.Andata.Membri.Count, Is.EqualTo(1));
31        Assert.That(andataRitorno.Ritorno.Membri.Count, Is.EqualTo(1));
32        Assert.That(andataRitorno.Andata.EventoAssociato, Is.EqualTo(_e));
33        Assert.That(andataRitorno.Ritorno.EventoAssociato, Is.EqualTo(_e));
34
35        LeaveEvento(_pr);
36        Assert.That(_pr, Is.EqualTo(null));
37    }
38 }

```

```

1 //Test Client.Controller.GestioneUtente
2 [TestFixture]
3 public class TestGestioneUtenteController
4 {
5     [SetUp]
6     public void ControllerSetUp() {
7     }
8
9     [Test]
10    public void TestMethod ()
11    {
12        //prevede che sia già registrato un utente con mail gianno@gmail.com e
13        //che abbia effettuato l'accesso su questo dispositivo.
14        //I dati default di questo utente sono:
15        //nominativo: giannone
16        //recapito: 0123456789
17        //Luogo: "Via S. Ruffillo 45", "40141", "Bologna", "BO","Italia"
18
19        UtenteRegistrato gianno = GetUtenteRegistrato("gianno@gmail.com");
20        UtenteRegistrato jalla = GetUtenteRegistrato("jalla@gmail.com");
21
22        Assert.That(gianno, Is.Not.EqualTo(null));
23        Assert.That(jalla, Is.EqualTo(null));
24        Assert.That(gianno.Username, Is.EqualTo("gianno@gmail.com"));
25
26        EditUsername("jalla@gmail.com");
27        gianno = GetUtenteRegistrato("gianno@gmail.com");
28        jalla = GetUtenteRegistrato("jalla@gmail.com");
29
30        Assert.That(jalla, Is.Not.EqualTo(null));
31        Assert.That(gianno, Is.EqualTo(null));
32        Assert.That(jalla.Username, Is.EqualTo("jalla@gmail.com"));
33
34        Assert.That(jalla.HasNominativo, Is.EqualTo(true));
35        Assert.That(jalla.HasRecapito, Is.EqualTo(true));
36        Assert.That(jalla.HasLuogo, Is.EqualTo(true));
37        Assert.That(jalla.Nominativo_default, Is.EqualTo("giannone"));
38        Assert.That(jalla.Recapito_default, Is.EqualTo("0123456789"));
39        Assert.That(jalla.Luogo_default, Is.EqualTo(new Luogo("Via S. Ruffillo
40        //45", "40141", "Bologna", "BO", "Italia")));
41
42        EditDefaultDati("jallona", "9876543210", null);
43        Assert.That(jalla.HasNominativo, Is.EqualTo(true));
44        Assert.That(jalla.HasRecapito, Is.EqualTo(true));
45        Assert.That(jalla.HasLuogo, Is.EqualTo(false));
46        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona"));
47        Assert.That(jalla.Recapito_default, Is.EqualTo("9876543210"));
48        Assert.That(jalla.Luogo_default, Is.EqualTo(null));
49    }
}

```

```

1 //Test Client.DominioRidotto.Model
2 [TestFixture]
3 public class TestAntiDoS {
4     [SetUp]
5     public void SetUp ()
6     {
7         //ci troviamo nella classe GestioneUtenteController
8     }
9
10    [Test]
11    public void TestMethod ()
12    {
13        //prevede che il MaxModPerRT dell'AntiDoS sia 5
14        //prevede che sia già registrato un utente con mail gianno@gmail.com e che abbia
15        //effettutato l'accesso su questo dispositivo.
16        //I dati default di questo utente sono:
17        //nominativo: giannone
18        //recapito: 0123456789
19        //Luogo: "Via S. Ruffillo 45", "40141", "Bologna", "BO","Italia"
20
21        UtenteRegistrato gianno = GetUtenteRegistrato("gianno@gmail.com");
22
23        EditDefaultDati("jallona1", "9876543210", null);      //prima mod
24        Assert.That(jalla.Nominativo, Is.EqualTo(true));
25        Assert.That(jalla.HasRecapito, Is.EqualTo(true));
26        Assert.That(jalla.HasLuogo, Is.EqualTo(false));
27        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona1"));
28        Assert.That(jalla.Recapito_default, Is.EqualTo("9876543210"));
29        Assert.That(jalla.Luogo_default, Is.EqualTo(null));
30
31        EditDefaultDati("jallona2", "9876543210", null);      //seconda mod
32        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona2"));
33
34        EditDefaultDati("jallona3", "9876543210", null);      //terza mod
35        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona3"));
36
37        EditDefaultDati("jallona4", "9876543210", null);      //quarta mod
38        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona5"));
39
40        EditDefaultDati("jallona5", "9876543210", null);      //quinta mod
41        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona5"));
42
43        EditDefaultDati("jallona6", "9876543210", null);      //sesto tentativo mod
44        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona5")); //non modificato
45
46        EditDefaultDati("jallona7", "9876543210", null);      //settimo tentativo mod
47        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona5")); //non modificato
48
49        EditDefaultDati("jallona6", "9876543210", null);      //ottavo tentativo mod
50        Assert.That(jalla.Nominativo_default, Is.EqualTo("jallona5")); //non modificato
51
52    }
53
54 }

```

# Progettazione per il Deployment

## Deployment

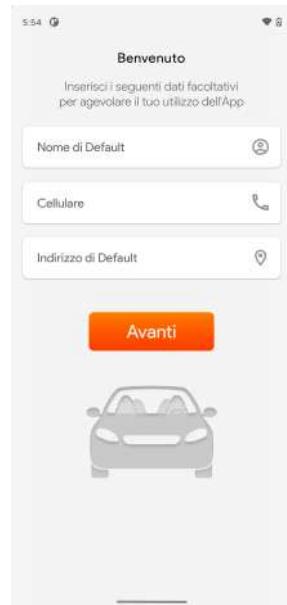
Per l'applicazione Client la gestione degli aggiornamenti sarà delegata completamente allo store del dispositivo, il quale notificherà la presenza di nuove versioni.

Inoltre verranno aggiunte due maschere di utilità:

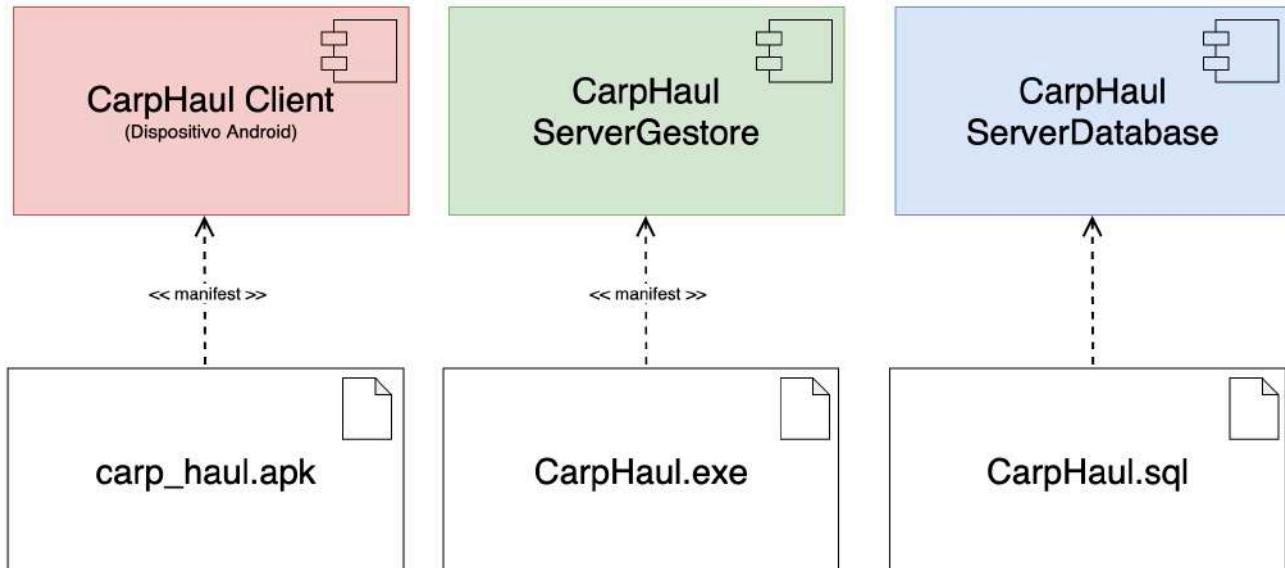
Schermata di evento terminato, qualora il GarbageCollector non avesse ancora eliminato un evento passato e un utente dovesse fare tap su questo, gli verrà presentata la seguente schermata.



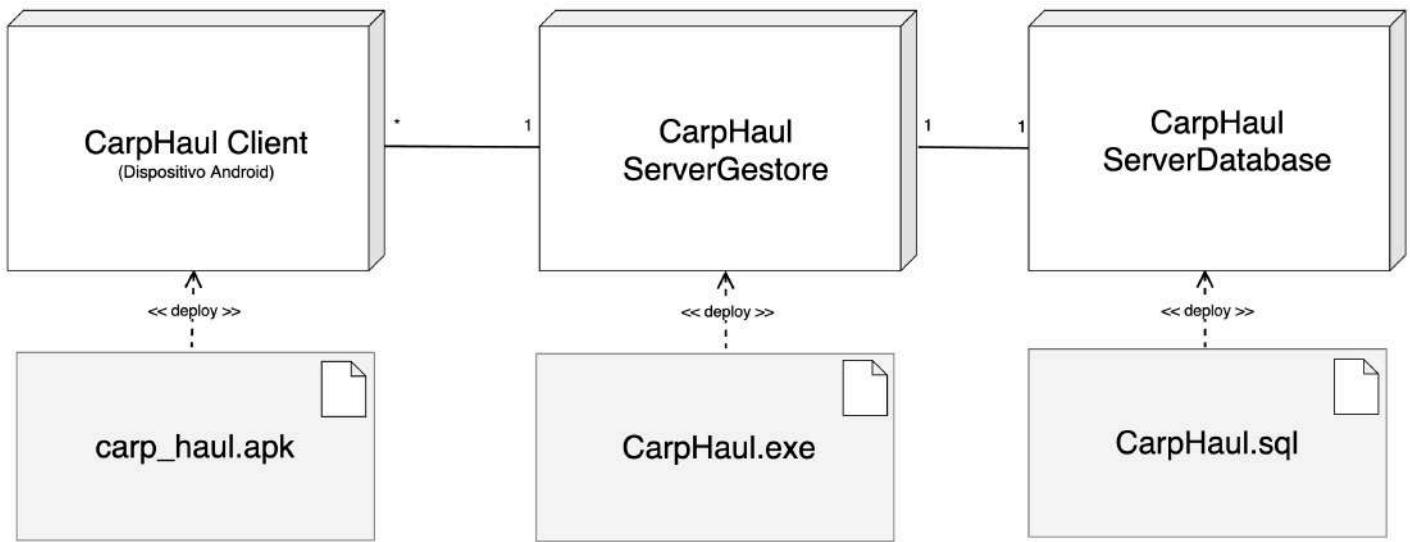
Schermata di Benvenuto, al primo accesso l'applicazione proporrà all'utente di compilare un modulo per salvare i suoi dati di default. Si potrà comunque decidere di saltare l'operazione.



## Artefatti



## Deployment Type-Level



# CarpHaul

