Ingegneria del Software T - Gruppo 3

Luca Bartolomei 0000825005
Luigi di Nuzzo 0000824873
Filippo Veronesi 0000832244

Marzo 2020

Indice

1 Abstract

Il progetto riguarda la creazione di un applicativo software gestionale per prevendite elettroniche.
Abbiamo pensato il software per un gruppo di amici che organizzano feste, con obiettivi cardine l'abbattimento di costi, l'ottimizzazione dell'entrata dei partecipanti all'evento e una semplificazione dei conti di bilancio.
L'idea di fondo è di utilizzare, come sostitutivo alla prevendita cartacea, un documento digitale in grado di far entrare il cliente dopo relativo check all'entrata.
Il risparmio economico ottenuto è ovviamente importante in confronto alla vendita tradizionale. Tuttavia bisogna tenere in conto dei problemi tecnologici che si possono verificare durante la vendita e l'entrata: problemi di connessione Internet, incompatibilità dei dispositivi dei clienti, training del personale addetto alle entrate, eccetera.
Vengono gestiti, oltre alle prevendite e i relativi clienti, anche la pianificazione dell'evento e i vari membri organizzativi, con relativa suddivisione dei ruoli.
Per facilitare i conti di bilancio è disponibile una sezione in cui è possibile ricavare statistiche sull'andamento dell'evento.

2 Analisi dei requisiti

2.1 Requisiti del sistema

2.1.1 Requesiti Funzionali

- R1F - Gli utenti devono essere identificati tramite username.
- R2F - Gli utenti sono autenticati tramite credenziali di username e password.
- R3F - L'accesso ad uno staff, da parte di un utente, avviene tramite codice di accesso.
- R4F - La registrazione di utenti è a carico dell'amministratore di sistema.
- R5F - Possibilità di cambiare la password personale dell'utente.
- R6F - Ogni membro di uno staff può ricoprire dei ruoli: cassiere, PR, amministratore.
- R7F - Il ruolo di cassiere riguarda la timbratura di prevendite all'ingresso di un evento.
- R8F - Il ruolo di PR riguarda la vendita di prevendite a clienti.
- R9F - Il ruolo di amministratore riguarda la gestione dei membri, degli eventi, delle tipologie di prevendita di un evento e della visualizzazione di tutte statistiche.
- R10F - Ogni utente registrato nel gestionale può diventare membro di uno o più staff.
- R11F - La timbratura di una prevendita valida permette al cliente di entrare all'evento, ovviamente lo staff potrà effettuare ulteriori controlli non previsti dal sistema e decidere di far entrare un cliente.
- R12F - Si prevede la generazione di un documento elettronico da consegnare al cliente, associato alla prevendita, per il riconoscimento all'entrata dell'evento.
- R13F - La vendita di una prevendita elettronica consiste nella consegna al cliente di un documento digitale di qualche forma, associato alla prevendita elettronica venduta.
- R14F/NF - Un amministratore può concedere/revocare i ruoli a qualsiasi membro dello staff. Unico vincolo è che rimanga almeno un amministratore.
- R15F - Possibilità di cambiare il codice di accesso dello staff da parte di un amministratore.
- R16F - Per gestione degli eventi di uno staff si intende la possibilità di vedere gli staff di un evento, di creane uno nuovo e di poter modificare un evento dello staff.
- R17F - La gestione delle tipologie di prevedita di un evento indica l'aggiunta, la modifica e la rimozione delle tipologie di prevendita.
- R18F - Un evento è composto da un nome, una descrizione, un periodo temporale di svolgimento e un luogo.
- R19F - Un evento può essere annullato, anche se ci sono prevendite vendute.
- R20F - Una tipologia di prevendita serve ad associare alla prevendita un prezzo, una descrizione e un periodo di vendita a tutte le prevendite con la stessa tipologia.
- R21F - Una prevendita può essere annullata e/o rimborsata.
- R22F - Le statistiche di un membro sono suddivise per ruolo coperto all'interno dello staff: cassiere o PR.
- R23F - Ogni prevedita è nominativa.
- R23Fbis - Il software prevede la possibilità di gestire più staff.

2.1.2 Requisiti non funzionali

- R2NF - La password degli utenti deve essere lunga almeno 8 caratteri.
- R3NF - Il codice di accesso allo staff deve essere lungo almeno 4 caratteri.
- R4NF - Ogni utente registrato può creare al massimo uno staff.
- R5NF - Requisito fondamentale è il basso costo del prodotto software.
- R6NF - I membri non amministratori possono vedere solo le statistiche personali.
- R7NF - I membri non amministratori possono solo vedere le tipologie di prevendite associate ad un evento.
- R8NF - I membri non amministratori possono solo vedere gli eventi dello staff.
- R9NF - Il periodo di vendita delle prevendite deve essere antecedente il periodo dell'evento.
- R10NF - Una prevendita annullata e/o rimborsata rimane tale.
- R11NF - Un evento annullato rimane tale.
- R12NF - Si prevedono più forme di consegna del documento digitale, per affrontare le eterogeneità.
- R13NF - La password fornita dall'amministratore di sistema a tempo di registrazione va cambiata immediatamente dopo il login.
- R14NF - Ogni prevendita venduta è associata ad una sola tipologia di prevendita.
- R15NF - La tipologia di prevendita associata non è modificabile.
- R16NF - Il prezzo di una tipologia di prevendita è modificabile solo se non sono state vendute prevendite con quella determinata tipologia.
- R17NF - Quando un PR vende una prevendita, essa viene associata ad esso.
- R18NF - Quando un Cassiere timbra una prevendita, essa viene associata ad esso.
- R19NF - Prima della vendita il cliente sceglierà una tipologia di prevendita associata all'evento a cui vuole partecipare.
- R20NF - La timbratura può essere fatta solo una volta.
- R21NF - L'interfaccia utente deve garantire velocità d'utilizzo, soprattutto per il cassiere.
- R21NFbis - Quando un utente crea uno staff ne diventa membro e amministratore.

2.2 Analisi del dominio

2.2.1 Glossario

Voce	Definizione	Sinonimi
Amministratore di sistema	Utente con privilegi di sistema aggiuntivi.	
Privilegio di sistema	Autorizzazione intrinseca concessa ad un amministratore di sistema che riguarda la gestione del software stesso. Non riguarda gli staff.	
Staff	Gruppo di utenti con lo scopo di organizzare eventi.	Ente organizzatore.
Utente	Persona registrata nel software gestione.	
Cliente	Persona che vuole partecipare ad un evento di uno staff.	
Membro	Utente che è iscritto ad uno staff.	Organizzatore
PR	Membro di uno staff che si occupa della vendita di prevendite.	
Cassiere	Membro di uno staff che si occupa dell'entrata dei clienti ad un evento.	
Amministratore	Membro di uno staff che si occupa della gestione dello staff stesso.	
Ruolo	Autorizzazione che ha il membro all'interno dello staff.	Autorizzazione
Evento	Avvenimento registrato dallo staff, per il quale è possibile vendere prevendite e registrare ingressi.	Festa
Tipologia Prevendita	Modello associato ad un evento, la quale dà le caratteristiche di prezzo e descrizione alla prevendita venduta.	Tipo Prevendita
Prevendita	Biglietto venduto anticipatamente, che consente l'entrata all'evento pagato.	Ticket, Prevendita Elettronica
Statistiche	Informazioni di carattere gestionale, riguardo ad un evento o a un membro dello staff.	
Documento digitale	Si tratta di una risorsa digitale, consegnata al cliente, che serve a identificare una prevendita venduta.	
Operazione	Comando richiesto al software gestionale da parte di un utente.	
Login	Operazione per identificare e autenticare un utente.	Accesso utente, Autenticazione
Timbratura	Operazione svolta da un cassiere svolta per validare una prevendita di un cliente.	Convalida della prevendita
Credenziali	Coppia di valori username e password utilizzata per l'autenticazione dell'utente.	
Username	Stringa di caratteri alfanumerici. Serve a identificare l'utente	
Password	String di caratteri alfanumerici. Può contenere caratteri speciali.	
Periodo di vendita	Periodo temporale in cui la prevendita è vendibile ai clienti.	
Anullamento Prevendita	Operazione che consiste nell'annullare una prevendita acquistata da un cliente.	
Rimborso Prevendita	Operazione eseguita su una prevendita annullata, al fine di tenere traccia del conto economico dello staff.	
Annullamento Evento	Operazione eseguita da un amministratore quando decide di annullare l'evento.	

2.3 Analisi dei requisiti

2.3.1 Casi d'uso

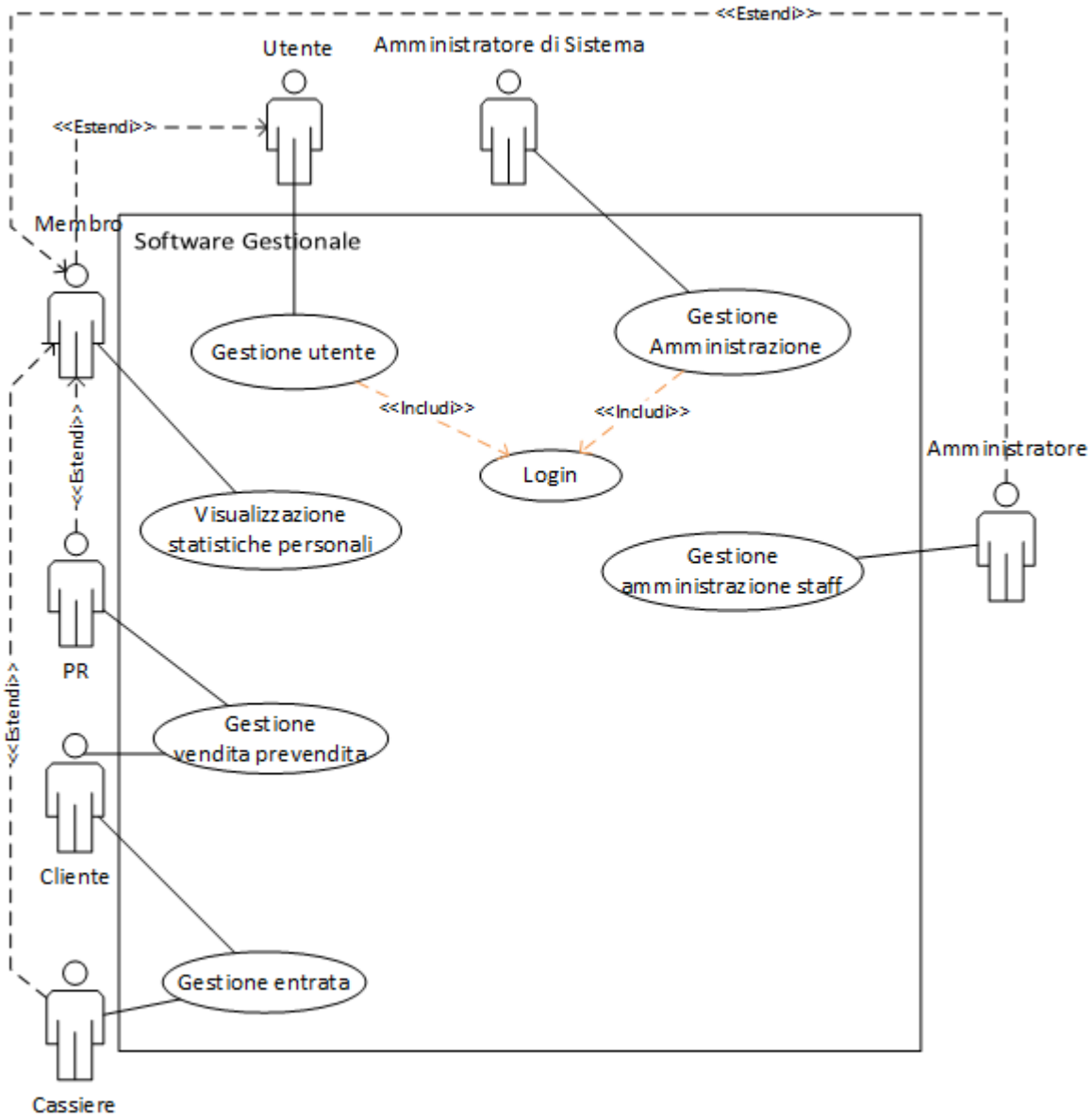

Figura 1: use_cases.vsdx

L'utilizzo del login (frecce arancioni) è necessario per i casi d'uso.
Abbiamo differenziato i casi d'uso delle statistiche perché si tratta di concetti diversi.

2.3.2 Scenari

Titolo	Login
Descrizione	Autenticazione nel software gestionale
Attori	Utente, Amministratore di Sistema
Caso d'uso	Login
Relazioni	
Precondizioni	L'utente deve essere registato nel sistema
Postcondizioni	L'utente è autenticato nel sistema
Scenario principale	1. Impostazione di una schermata per l'inserimento di username e password. 2. L'utente inserisce i dati del proprio account. 3. L'utente esegue l'operazione. 4. Il sistema verifica le credenziali. 5. Dopo l'autenticazione viene mostrata una schermata principale.
Scenari alternativi	A) L'utente sbaglia credenziali: 1. Notifica all'utente. 2. Ritorno alla schermata di login. B) L'utente ha sbagliato troppe volte le credenziali: 1. Blocco dell'account. 2. Notifica all'utente. 3. Ritorno alla schermata di login. C) L'utente ha la password di default impostata dall'amministratore di sistema: 1. Passaggio al cambio password. D) L'utente ha l'account bloccato: 1. Notifica all'utente. 2. Ritorno alla schermata di login. E) L'utente è gia autenticato: 1. Mostro la schermata principale.
Requisiti non funzionali	R2NF, R7NF, R16NF
Punti aperti	

Titolo	Gestione Amministrazione
Descrizione	Amministrazione del software gestionale
Attori	Amministratore di sistema
Caso d'uso	Gestione Amministrazione
Relazioni	Login
Precondizioni	L'utente deve essere registato nel sistema
Postcondizioni	L'utente è autenticato nel sistema
Scenario principale	1. Login. 2. Viene mostrata una schermata con le azioni possibili: a. Registrazione utente 3. L'Amministratore di sistema sceglie l'operazione desiderata. 4. Il sistema cambia vista a seconda della scelta.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Registrazione utente
Descrizione	Una persona ha richiesto l'accesso al sistema gestionale e l'amministratore vuole registrarlo.
Attori	Amministratore di sistema
Caso d'uso	Gestione Amministrazione
Relazioni	
Precondizioni	L'amministratore di sistema è autenticato
Postcondizioni	
Scenario principale	1. L'amministratore si informa riguardo l'utente, chiedendo informazioni per la registrazione. 2. L'utente fornisce i dati all'amministratore. 3. L'amministratore sceglie username e password iniziale per l'utente, utilizzando criteri di sicurezza.
Scenari alternativi	A) Dati dell'utente da registrare in conflitto o non validi: 1. Notifica all'amministratore. 2. L'amministratore cerca di correggere i dati. 3. L'amministratore ripete l'operazione.
Requisiti non funzionali	R2NF
Punti aperti	

Titolo	Gestione utente
Descrizione	Gestione degli utenti presenti nel software gestionale
Attori	Utente
Caso d'uso	Gestione utente
Relazioni	Login
Precondizioni	L'utente deve essere registato nel sistema
Postcondizioni	L'utente è autenticato nel sistema
Scenario principale	1. Login. 2. Viene mostrata una schermata con le azioni possibili: a. Cambia Password. b. Accedi staff. c. Crea staff. d. Scegli Staff 3. L'utente sceglie l'operazione desiderata. 4. Il sistema cambia vista a seconda della scelta.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Crea staff
Descrizione	Creazione di uno staff
Attori	Utente
Caso d'uso	Gestione utente
Relazioni	
Precondizioni	Utente già autenticato
Postcondizioni	
Scenario principale	1. L'utente accede alla creazione staff. 2. Il sistema fornisce una vista di creazione. 3. L'utente inserisce i dati del nuovo staff. 4. L'utente esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	A) L'utente ha già creato uno staff: 1. Notifica all'utente 2. Ritorno alla schermata precedente.
Requisiti non funzionali	R3NF
Punti aperti	

Titolo	Cambia password
Descrizione	Cambio della password di un utente
Attori	Utente
Caso d'uso	Gestione utente
Relazioni	
Precondizioni	Utente già autenticato
Postcondizioni	
Scenario principale	1. L'utente accede al cambio password. 2. Il sistema fornisce una vista. 3. L'utente inserisce i dati per il cambio di password. 4. L'utente esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R2NF
Punti aperti	

Titolo	Accedi allo staff
Descrizione	Permettere di diventare membro di uno staff
Attori	Utente
Caso d'uso	Gestione utente
Relazioni	
Precondizioni	Utente già autenticato
Postcondizioni	
Scenario principale	1. L'utente accede alla schermata di accedi staff. 2. Il sistema fornisce una vista. 3. L'utente inserisce i dati per l'accesso allo staff. 4. L'utente esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	A) L'utente è già membro dello staff: 1. Notifica all'utente. 2. Ritorno alla schermata precedente. B) L'utente ha fornito un codice di accesso errato: 1. Notifica all'utente.
Requisiti non funzionali	R3NF
Punti aperti	

Titolo	Visualizza statistiche personali
Descrizione	Permettere di visualizzare le statistiche personali di un membro come PR o cassiere. Non è associato direttamente a PR e a cassiere perché i ruoli di un membro potrebbero cambiare.
Attori	Membro
Caso d'uso	Visualizza statistiche personali
Relazioni	
Precondizioni	Il membro è già autenticato
Postcondizioni	
Scenario principale	1. Il membro accede alla schermata di vusualizzazione delle statistiche personali. 2. Il sistema fornisce una vista con tutte le statistiche. 3. Quando il membro ha finito la consultazione, ritorna alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Gestione vendita prevendita
Descrizione	Permette di gestire la vendita di una prevendita: aggiunta prevendita, annulla prevendita
Attori	PR, Cliente
Caso d'uso	Gestione vendita prevendita
Relazioni	
Precondizioni	Il PR è già autenticato
Postcondizioni	
Scenario principale	1. Il PR accede alla schermata di gestione prevenite. 3. Il sistema fornisce una vista con le opzioni possibili. 4. Il PR sceglie l'operazione: a. Aggiungi prevendita. b. Annulla prevendita. c. Lista prevendite. 5. Il PR inserisce i dati relativi all'operazione, consultando il cliente se necessario. 6. Il PR esegue l'operazione. 7bis. Nel caso di aggiungi prevendita si consegna al cliente il documento digitale. 8. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R8NF, R9NF, R10NF, R12NF, R14NF, R15NF, R17NF, R19NF
Punti aperti	

Titolo	Gestione entrata
Descrizione	Permette di gestire l'entrata di un cliente: timbratura dell'entrata
Attori	Cassiere, Cliente
Caso d'uso	Gestione entrata
Relazioni	
Precondizioni	Il Cassiere è già autenticato
Postcondizioni	
Scenario principale	1. Il Cassiere accede alla schermata di gestione entrata. 2. Il sistema fornisce una vista con le opzioni disponibile. 3. Il Cassiere sceglie l'operazione: a.Timbra entrata. b. Lista entrate. 4. Il Cassiere inserisce i dati dell'operazione, consultando il cliente se necessario. 5. Il cassiere esegue l'operazione. 6. Il Cassiere informa il cliente riguardo l'esito dell'operazione, se necessario. 7. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R18NF, R20NF
Punti aperti	

Titolo	Gestione Amministrazione staff
Descrizione	Gestione dello staff dell'utente
Attori	Amministratore
Caso d'uso	Gestione Amministrazione staff
Relazioni	
Precondizioni	Utente autenticato
Postcondizioni	
Scenario principale	1. Viene mostrata una schermata con le azioni possibili: a. Gestione membri. b. Cambia codice di accesso. c. Visualizzazione statistiche. d. Gestione eventi. e. Gestione tipologie prevendite. 2. L'amministratore sceglie l'operazione desiderata. 3. Il sistema cambia vista a seconda della scelta.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Cambia codice accesso
Descrizione	Cambio del codice di accesso allo staff
Attori	Amministratore
Caso d'uso	Gestone amministrazione staff
Relazioni	
Precondizioni	L'amministratore è già autenticato.
Postcondizioni	
Scenario principale	1. L'amministratore accede al cambio codice di accesso. 2. Il sistema fornisce una vista. 3. L'amministratore inserisce i dati per il cambio del codice. 4. L'amministratore esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R3NF
Punti aperti	

Titolo	Visualizza statistiche
Descrizione	Permettere di visualizzare le statistiche dello staff.
Attori	Amministratore
Caso d'uso	Gestone amministrazione staff
Relazioni	
Precondizioni	L'amministratore è già autenticato.
Postcondizioni	
Scenario principale	1. L'amministratore accede alla schermata di vusualizzazione. 2. Il sistema fornisce una vista con tutte le statistiche. 3. Quando l'amministratore ha finito la consultazione, ritorna alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R6NF
Punti aperti	

Titolo	Gestione eventi
Descrizione	Permettere di gestire gli eventi dello staff.
Attori	Amministratore
Caso d'uso	Gestone amministrazione staff
Relazioni	
Precondizioni	L'amministratore è già autenticato
Postcondizioni	
Scenario principale	1. L'amministratore accede alla schermata di vusualizzazione. 2. Il sistema fornisce una vista con le scelte: a. Aggiungi evento. b. Modifica evento. c. Annulla evento. d. Lista eventi. 3. Una volta scelta l'operazione, l'amministratore inserisce i dati richiesti. 4. L'amministratore esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R8NF, R11NF
Punti aperti	

Titolo	Gestione tipologie prevendite
Descrizione	Permettere di gestire le tipologie associate alle prevendite di un evento.
Attori	Amministratore
Caso d'uso	Gestone amministrazione staff
Relazioni	
Precondizioni	L'amministratore è già autenticato
Postcondizioni	
Scenario principale	1. L'amministratore accede alla schermata di vusualizzazione. 2. Il sistema fornisce una vista con le scelte: a. Aggiungi tipologia evento. b. Modifica tipologia evento. c. Elimina tipologia evento. d. Lista tipologie evento. 3. Una volta scelta l'operazione, l'amministratore inserisce i dati richiesti. 4. L'amministratore esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	R9NF, R16NF
Punti aperti	

Titolo	Gestione membri
Descrizione	Permettere di gestire i membri dello staff, concedendo i ruoli o rimuovendo i membri.
Attori	Amministratore
Caso d'uso	Gestone amministrazione staff
Relazioni	
Precondizioni	L'amministratore è già autenticato.
Postcondizioni	
Scenario principale	1. L'amministratore accede alla schermata di vusualizzazione. 2. Il sistema fornisce una vista con le scelte: a. Modifica ruoli membro. b. Rimuovi membro. c. Lista membri. 3. Una volta scelta l'operazione, l'amministratore inserisce i dati richiesti. 4. L'amministratore esegue l'operazione. 5. Ritorno alla schermata precedente.
Scenari alternativi	A) L'amministratore si vuole rimuovere ma è l'unico amministratore rimasto nello staff: 1. Notifica all'utente. 2. Ritorno alla schermata precedente.
Requisiti non funzionali	R14F/NF
Punti aperti	

2.4 Analisi del rischio

2.4.1 Valutazione dei beni

Bene	Valore	Esposizione
Sistema gestionale	Alto Registra tutte le prevendite di uno staff. Critico dal punto di vista della sicurezza.	Alta Perdita d'immagine del software e degli staff, spesa in tempo per il ripristino del backup, se il sistema fallisce è necessario acquistare prevendite fisiche. Possibili perdite finanziare.
Informazioni su utenti	Medio-Alto Informazioni relative agli utenti del sistema, anche le credenziali, le quali potrebbero permettere l'accesso ai dati dello staff e relativi clienti.	Alta Perdita d'immagine del software, un membro con ruolo amministrativo potrebbe compromettere temporaneamente uno staff fino a ripristino backup. Possibile perdita economica dovuta all'aggiunta di prevendite.
Informazioni su prevendite	Alto Rappresentano i guadagni per uno staff, nonché il biglietto d'ingresso di un cliente.	Alta Perdita d'immagine del software: lo staff ha perso informazioni riguardo i ricavi. Perdita d'immagine dello staff: il cliente non riesce a entrare. Possibile perdita economica se vengono aggiunte prevendite non dovute.
Informazioni su clienti	Medio Le informazioni sono ridotte al solo nome e cognome (R23F).	Bassa
Informazioni su amministratori di sistema	Alto Ha poteri sul sistema gestionale.	Alta Possibilità di perdere informazioni di sistema. Perdita d'immagine del software.

2.4.2 Analisi minacce e controlli

Minaccia	Probabilità	Controllo	Fattibilità
Furto d'identità (utente)	Media-Bassa. Username e password scelti dall'amministratore di sistema.	Log degli accessi, blocco dell'utente.	Costo minimo
Furto d'identità (amministratore di sistema)	Bassa	Log degli accessi	Costo minimo
Alterazione dei dati nella comunicazione Man in the Middle	Medio-Alta. Molte comunicazione avvengono nel sistema distribuito client/server.	Uso di canale sicuro	Costo basso anche se si utilizza un mix tra chiave simmetrica e asimmetrica.
DoS/DDoS	Bassa	Sistema locale durante l'evento	Costo medio, richiesta di risorse hardware portatili.
Contraffazione della prevendita	Media	Sistema anti-contraffazione, prevendita nominativa	Costo minimo
Furto della prevendita	Media	Prevendita nominativa	Costo minimo

2.4.3 Analisi della tecnologia dal punto di vista della sicurezza

Tecnologia	Vulnerabilità
Cifratura delle comunicazioni	Utilizzo di una **doppia cifratura** simmetrica e asimmetrica con le loro relative vulnerabilità: **Cifratura Simmetrica**: 1. Tempo di vita della chiave 2. Memorizzazione della chiave 3. Lunghezza della chiave **Cifratura Asimmetrica**: 1. Lunghezza della chiave 2. Memorizzazione della chiave privata L'utilizzo di cifrature standard non recenti potrebbero causare vulnerabilià ulteriori.
Autenticazione tramite credenziali	**Social engineering**: un utente potrebbe essere vittima di un attacco di social engineering, volto al furto delle credenziali. **Password cracking**: attraverso metodologie di cracking delle password, l'attaccante risale alla password di un utente o amministratore di sistema. **Negligenza**: L'utente volontariamente o per sbaglio fornisce le credenziali a terzi.
Architettura Client/Server	**Attacco DDoS/DoS**: l'obiettivo dell'attacco è quello di negare il servizio offerto agli utenti. **Attacco Man in the Middle**: un malintenzionato potrebbe riuscire a sottrarre informazioni o effettuare operazioni illecite. **Sniffing della comunicazione**: un malintezionato potrebbe riusire ad intercettare una comunicazione e a sottrarne informazioni. **Deadlock**: Il servitore potrebbe trovarsi in una situzione di stallo generata da una richiesta ciclica. **Esposizione dei client**: I client sono esposti, dato che devono connettersi al servitore.

2.4.4 Security Use Case e Misuse Case

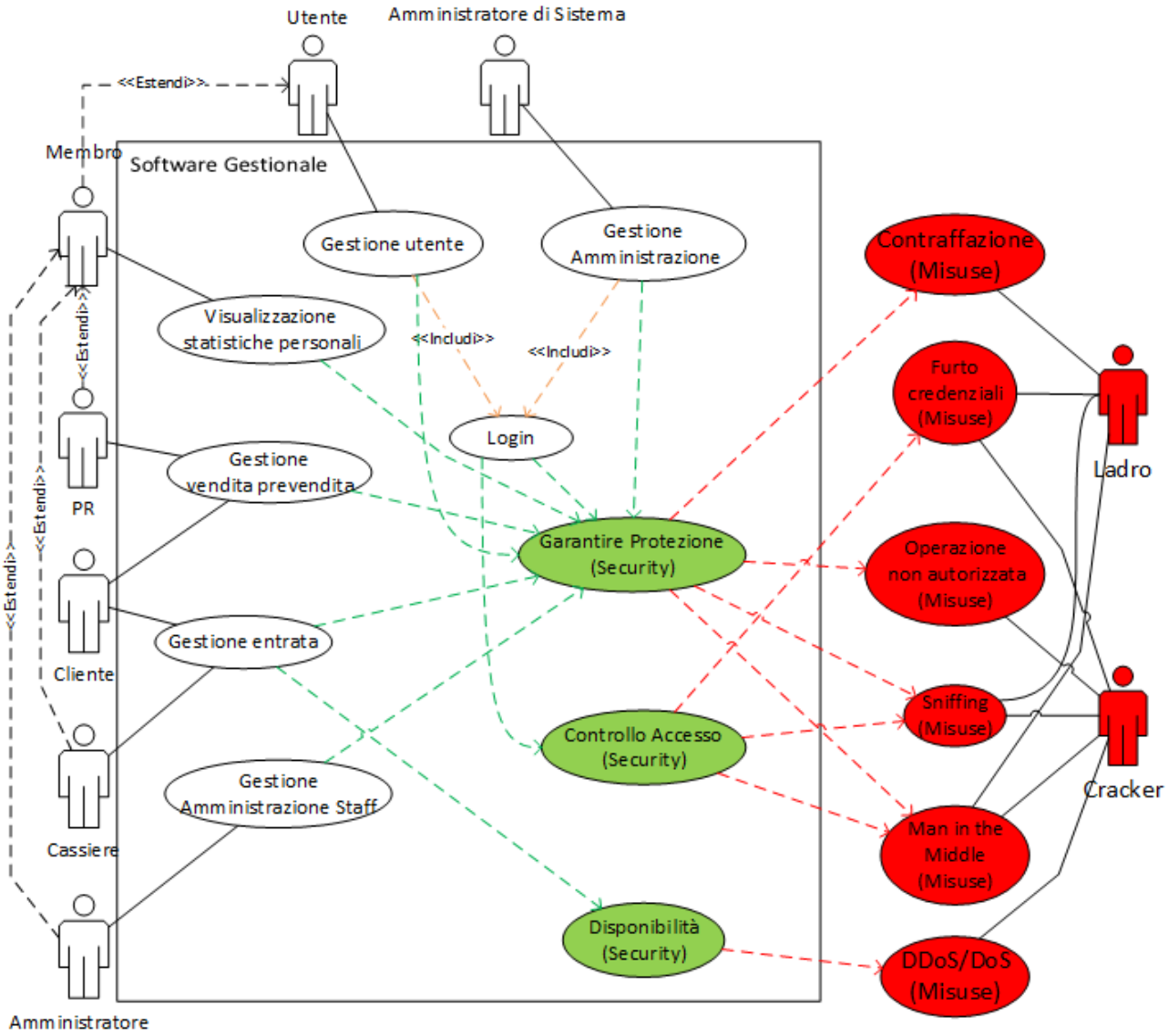

Figura 2: security_use_cases_misuse_cases.vsdx

Caso d'uso	Garantire Protezione	
Percorso d'uso	Garantire Protezione dei dati persistenti	
Misuse case	Man in the Middle, Sniffing, Furto credenziali	
Descrizione	I dati persistenti devono essere protetti.	
Rischi alla sicurezza	Un utente malintenzionato potrebbe modificare o compromettere i dati relativi al software gestionale, come gli utenti registrati o le prevendite di un evento.	
Precondizioni	1. Il sistema è già stato utilizzato dall'amministratore di sistema per la registrazione degli utenti e/o uno staff ha già registrato delle prevendite per un evento. 2. L'attaccante ha i mezzi necessari per tentare di modificare i dati persistenti.	
Postcondizioni	Il sistema blocca il tentativo di modifica o compromissione dei dati persistenti.	
Scenario principale	**Sistema**	**Attaccante**
		L'attaccante cerca di modificare dati persistenti del sistema gestionale.
	Il sistema blocca l'attacco e notifica l'amministratore di sistema.	
Scenario di attacco avvenuto con successo	**Sistema**	**Attaccante**
		L'attaccante riesce ad accedere/-modificare dati persistenti.
	Il sistema registra nei log le modifiche effettuate.	
	L'amministratore di sistema nota i log anomali e opera di conseguenza.	

Caso d'uso	Garantire Protezione	
Percorso d'uso	Garantire Protezione dei dati della comunicazione	
Misuse case	Man in the Middle, Sniffing, Furto credenziali	
Descrizione	I dati che viaggiano nelle comunicazioni devono essere protetti.	
Rischi alla sicurezza	Un utente malintenzionato potrebbe leggere o alterare i dati in transito in una comunicazione.	
Precondizioni	1. Il malintezionato ha la possibilità di intercettare i dati. 2. Il malintenzionato ha la possibilità di modificare i dati. 3. ha i mezzi per spedire il messaggio modificato al destinatario.	
Postcondizioni	Il sistema rileva il tentativo di modifica della comunicazione.	
Scenario principale	**Sistema**	**Attaccante**
	Il sistema si occupa di garantire una comunicazione dei messaggi sicura.	
		L'attaccante rileva un messaggio in transito, lo interecetta, lo modifica e lo inoltra all'utente finale.
	Il sistema riceve il messaggio modificato e rileva il tentativo di lettura o modifca, rifiutandolo e segnalandolo nei log.	
Scenario di attacco avvenuto con successo	**Sistema**	**Attaccante**
	Il sistema si occupa di garantire una comunicazione dei messaggi sicura.	
		L'attaccante riesce ad alterare la comunicazione senza che il sistema se ne accorga.
	Il sistema registra nei log le modifiche effettuate.	
	L'amministratore di sistema nota i log anomali e opera di conseguenza.	

Caso d'uso	Garantire Protezione	
Percorso d'uso	Garantire Protezione rispetto alle contraffazioni	
Misuse case	Contraffazione	
Descrizione	Devo garantire che le prevendite lette siano autentiche.	
Rischi alla sicurezza	Un utente malintenzionato potrebbe falsificare una prevendita, mediante documento digitale consegnato.	
Precondizioni	1. Il malintezionato conosce la struttura di una prevendita e del documento digitale. 2. Il malintenzionato ha la possibilità di falsificare la prevendita.	
Postcondizioni	Il sistema rileva che la prevendita non è autentica.	
Scenario principale	**Sistema**	**Attaccante**
	Il sistema si occupa di garantire l'autenticità delle prevendite consegnate.	
		L'attaccante crea una prevendita falsificata e tenta il riconoscimento da parte del sistema.
	Il sistema rileva che la prevendita è contraffatta, rifiutandola e segnalandola nei log.	
Scenario di attacco avvenuto con successo	**Sistema**	**Attaccante**
	Il sistema si occupa di garantire l'autenticità delle prevendite consegnate.	
		L'attaccante riesce a creare una prevendita che illuda il sistema.
	Il sistema registra nei log l'entrata effettuata.	
	L'amministratore dello staff nota una discrepanza e informa l'amministratore di sistema.	
	L'amministratore di sistema fa una ricerca nei log e tenta di risalire al malfattore.	

Caso d'uso	Controllo Accesso	
Percorso d'uso	Controllo sull'autenticazione dell'utente	
Misuse case	Furto credenziali	
Descrizione	Il sistema deve proteggere l'utente da intrusioni di malintenzionati.	
Rischi alla sicurezza	Un malintenzionato ruba i dati di identificazione e autenticazione ad un utente.	
Precondizioni	1. Il malintezionato ha la possibilità di trovare l'username. 2. Il malintenzionato ha la possibilità di tentare l'accesso al sistema.	
Postcondizioni	Il sistema rileva il tentativo di accesso fraudolento e blocca l'utente.	
Scenario principale	**Sistema**	**Attaccante**
		L'attaccante tenta un attacco di password cracking.
	Il sistema rileva l'attacco e blocca l'account dell'utente.	
Scenario di attacco avvenuto con successo	**Sistema**	**Attaccante**
		L'attaccante riesce a scoprire la password.
	Il sistema fornisce l'accesso al sistema.	
		Il malintenzionato tenta di reperire tutte le informazioni prima di essere scoperto.
	Il sistema notifica nei log le operazioni effettuate.	
	L'amministratore di sistema nota i log anomali e opera di conseguenza.	

Caso d'uso	Controllo Accesso	
Percorso d'uso	Controllo sulle operazioni eseguite dagli utenti	
Misuse case	Operazione non autorizzata	
Descrizione	Il sistema deve evitare che utenti non autorizzati possano eseguire operazioni non concesse. Deve essere valido sia per i ruoli di uno staff, sia per il ruolo di amministratore di sistema.	
Rischi alla sicurezza	Un utente normale esegue senza autorizzazione operazioni riservate all'amministratore di sistema. Un membro di uno staff esegue senza autorizzazione operazioni riservate ad un ruolo che non ha.	
Precondizioni	1. L'utente ha credenziali validi per l'accesso. 2. L'utente non ha i privilegi per cui sta tentando di effettuare l'operazione.	
Postcondizioni	Il sistema rileva il tentativo di esecuzione non autorizzata e blocca l'operazione.	
Scenario principale	**Sistema**	**Attaccante**
		L'utente tenta di eseguire un operazione non consentita.
	Il sistema verifica le autorizzazioni dell'utente e blocca l'operazione non autorizzata. Il sistema aggiunge un record di log su quanto accaduto.	

Caso d'uso	Disponibilità	
Misuse case	DDoS/DoS	
Descrizione	Il sistema deve cercare di proteggersi da attacchi DDoS/DoS.	
Rischi alla sicurezza	Un malintenzionato vuole impedire l'accesso al sistema durante un evento.	
Precondizioni	1. Il malintezionato dispone dei mezzi per effettuare l'attacco.	
Postcondizioni	Il sistema si protegge dalla minaccia.	
Scenario principale	**Sistema**	**Attaccante**
		L'attaccante tenta un attacco DDoS/DoS.
	Il sistema rileva l'attacco e informa l'amministratore di sistema, mentre cerca di attuare misure per ridurre l'efficacia dell'attacco.	
	Il sistema riesce a contenere l'attacco, dato che l'attaccante non ha sufficiente potenza per mettere fuori uso il sistema.	
	L'amministratore di sistema controlla lo stato e agisce di conseguenza.	
Scenario di attacco avvenuto con successo	**Sistema**	**Attaccante**
		L'attaccante tenta un attacco DDoS/DoS.
	Il sistema rileva l'attacco e informa l'amministratore di sistema, mentre cerca di attuare misure per ridurre l'efficacia dell'attacco.	
	Il sistema non riesce a contenere l'attacco e va fuori uso.	
	L'amministratore di sistema vede che il sistema di fuori uso e tenta un ripristino locale per favorire l'accesso al sistema durante l'evento.	

2.4.5 Requisiti di Protezione dei Dati

Dall'analisi del rischio sono emersi altri requisiti riguardanti la protezione dei dati:

- Creazione di un sistema di log per il tracciamento delle operazioni svolte nel sistema gestionale, garantendo la persistenza dei log. L'amministratore di sistema è l'unico a poter accedere ai log. Per evitare l'uscita di informazioni riservate, cercare di memorizzare nei log solo le informazioni utili per l'analisi da parte dell'amministratore.
- Creare un sistema per il blocco dell'utente dopo un certo numero di tentativi. L'amministratore di sistema ha la possibilità di sbloccare l'utente.
- I dati scambiati devono essere protetti con cifratura sicura per evitare alterazione e lettura di dati privati.
- I dati persistenti del sistema devono essere protetti da eventuali intrusioni.
- Necessità di un sistema per il controllo delle autorizzazioni degli utenti.
- Il sistema deve essere in grado di ridurre l'impatto di attacchi DoS, per esempio tramite blacklisting di IP o tramite servizi esterni, come l'uso di reverse proxy, ovviamente configurato a dovere, non come il sistema dell'INPS.
- Inoltre il sistema deve essere strutturato in maniera da consentire il ripristino locale ad un evento da parte dell'amministratore di sistema, nel caso che l'attacco DoS funzioni.
- Il sistema deve prevedere un sistema di anti-contraffazione per garantire l'autenticità delle preventite.

2.4.6 Requisiti di sistema aggiornati

REQUISITI FUNZIONALI

- R24F - Creare un sistema di log per il tracciamento delle operazioni automatico.
- R25F - L'amministratore di sistema deve avere il pieno controllo dei log.
- R26F - Creare un sistema per il bloccaggio dell'account dopo vari tentativi. L'amministratore di sistema ha l'autorizzazione a sbloccare l'account.

REQUISITI NON FUNZIONALI

- R22NF - I dati della comunicazione devono essere protetti.
- R23NF - I dati persistenti devono essere protetti.
- R24NF - Il sistema deve verificare le autorizzazioni degli utenti e dei membri.
- R25NF - Il sistema deve cercare di garantire la disponibilità del servizio.
- R26NF - Il sistema deve essere strutturato in modo da essere replicabile localmente.
- R27NF - Il sistema deve essere in grado di riconoscere prevendite contraffatte.
- R28NF - Il blocco dell'account deve avvenire dopo 3 tentativi.

2.4.7 Glossario aggiornato

Voce	Definizione	Sinonimi
Log	Registro dove vengono salvate informazioni per risalire ad operazioni critiche svolte. Composto da una serie di voci.	Registro
Voce di log	Si tratta di una riga del log.	Record del log
Blocco dell'utente	Operazione automatica eseguita per la protezione del sistema. L'utente non può più effettuare l'operazione di autenticazione con successo.	Blocco dell'account utente
Sblocco dell'utente	Operazione eseguita da un amministratore di sistema, per il ripristino di un utente con autenticazione bloccata.	Sblocco dell'account utente
Verifica Contraffazione	Sistema volto a verificare l'autenticità di un documento digitale.	

2.4.8 Casi d'uso aggiornati

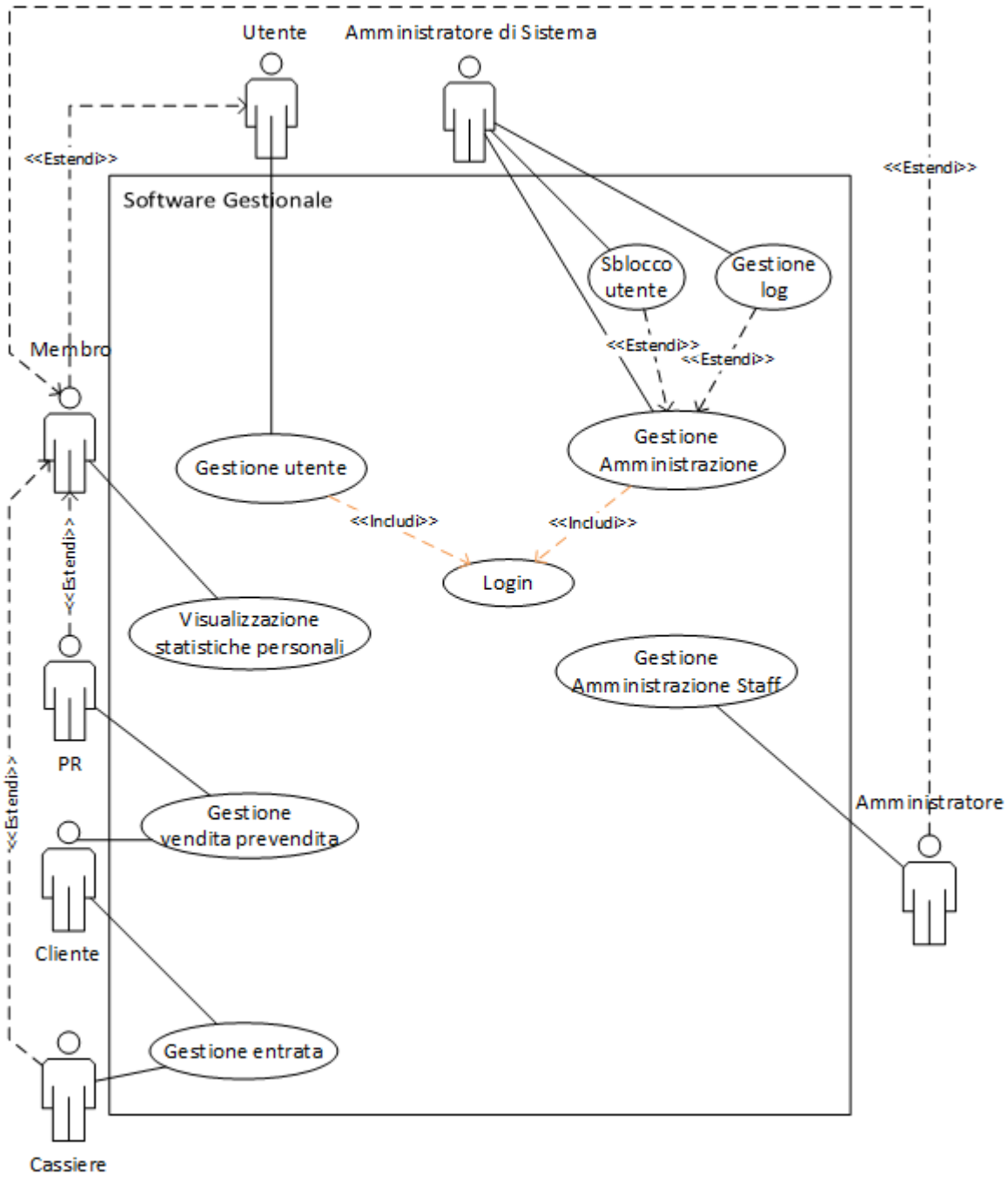

Figura 3: use_cases_aggiornato.vsdx

Vengono aggiunti due casi d'uso: Gestione log e Sblocco utente. Entrambi casi d'uso sono associati ad un amministratore di sistema.

2.4.9 Scenari aggiornati

Titolo	Gestione log
Descrizione	Gestione del log del software gestionale
Attori	Amministratore di sistema
Relazioni	Gestione Amministrazione
Precondizioni	
Postcondizioni	
Scenario principale	1. L'amministratore di sistema è già autenticato. 2. L'amministratore di sistema sceglie una funzione: a. Aggiungi log b. Leggi log 3. L'amministratore di sistema esegue l'operazione. 4. Ritorno alla schermata precedente.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Sblocco utente
Descrizione	Caso in cui un utente è bloccato e chiede aiuto all'amministratore di sistema.
Attori	Amministratore di sistema
Relazioni	Gestione Amministrazione
Precondizioni	
Postcondizioni	
Scenario principale	1. L'amministratore di sistema è già autenticato. 2. L'amministratore si informa riguardo l'utente. 3. L'utente si spiega di quanto accaduto con l'amministratore. 4. Se decide di sbloccarlo allora inserisce i dati nell'apposita vista. 5. L'amministratore esegue l'operazione e l'utente viene sbloccato.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Titolo	Login
Descrizione	Autenticazione nel software gestionale.
Attori	Utente, Amministratore di Sistema
Relazioni	
Precondizioni	L'utente deve essere registrato nel sistema.
Postcondizioni	L'utente è autenticato nel sistema.
Scenario principale	1. Impostazione di una schermata per l'inserimento di username e password. 2. L'utente inserisce i dati del proprio account. 3. L'utente esegue l'operazione. 4. Il sistema verifica le credenziali. 5. Dopo l'autenticazione viene mostrata una schermata principale.
Scenari alternativi	A) L'utente sbaglia credenziali: 1. Notifica all'utente. 2. Ritorno alla schermata di login. B) L'utente ha sbagliato troppe volte le credenziali: 1. Blocco dell'account. 2. Notifica all'utente. 3. Ritorno alla schermata di login. C) L'utente ha la password di default impostata dall'amministratore di sistema: 1. Passaggio al cambio password. D) L'utente ha l'account bloccato: 1. Notifica all'utente. 2. Ritorno alla schermata di login. E) L'utente è gia autenticato: 1. Mostro la schermata principale.
Requisiti non funzionali	R2NF, R7NF, R16NF, R28NF
Punti aperti	

Viene aggiunto semplicemente il requisito R28NF

3 Analisi del problema

3.1 Analisi delle funzionalità

3.1.1 Tabella delle funzionalità

Funzionalità	Tipo	Grado di Complessità	Requisiti collegati
Autenticazione	Lettura di dati, interazione con l'esterno	Semplice	R1F, R2F
Gestione Amministrazione	Gestione dei dati, memorizzazione dei dati, interazione con l'esterno	Complesso	
Gestione log	Gestione dei dati, interazione con l'esterno	Semplice	R24F, R25F
Scrittura log	Memorizzazione di dati	Semplice	R24F
Sblocco utente	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Semplice	R26F
Registrazione utente	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R1F, R2F, R4F
Gestione utente	Gestione dei dati, memorizzazione dei dati, interazione con l'esterno	Complesso	
Crea staff	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R23Fbis, R3F, R21NFbis
Accedi staff	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Semplice	R23Fbis, R3F, R10F
Cambia password	Gestione dei dati, interazione con l'esterno	Semplice	R2F, R5F
Gestione amministrazione staff	Gestione dei dati, memorizzazione dei dati, interazione con l'esterno	Complesso	
Visualizzazione statistiche personali	Lettura dati, interazione con l'esterno	Semplice	R22F
Gestione vendita prevendita	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R8F, R12F, R13F, R21F, R23F
Gestione entrata	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R7F, R11F, R12F
Gestione membri	Gestione dei dati, interazione con l'esterno	Complesso	R6F, R9F, R14F/NF
Cambia codice accesso	Gestione dei dati, interazione con l'esterno	Semplice	R15F
Visualizzazione statistiche	Lettura dati, interazione con l'esterno	Semplice	R9F
Gestione eventi	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R9F, R16F, R18F, R19F
Gestione tipologie prevendite	Gestione dei dati, memorizzazione di dati, interazione con l'esterno	Complesso	R9F, R17F, R20F

3.1.2 Tabelle Informazioni/Flusso

Gestione Amministrazione - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli

Gestione log - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Record d'inserimento **Composto da:**	Complesso	Medio	Input	
Timestamp	Semplice	Medio	Input	
Descrizione	Semplice	Medio	Input	Non più di 256 caratteri
Livello di importanza	Semplice	Medio	Input	**Enumeratore**: Basso, Medio, Alto
Record di lettura **Composto da:**	Complesso	Medio	Output	
Timestamp	Semplice	Medio	Output	
Descrizione	Semplice	Medio	Output	Non più di 256 caratteri
Livello di importanza	Semplice	Medio	Output	**Enumeratore**: Basso, Medio, Alto

Scrittura log - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Record d'inserimento **Composto da:**	Complesso	Medio	Input	
Timestamp	Semplice	Medio	Input	
Descrizione	Semplice	Medio	Input	Non più di 256 caratteri
Livello di importanza	Semplice	Medio	Input	**Enumeratore**: Basso, Medio, Alto

Sblocco utente - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Informazioni per lo sblocco	Semplice	Medio	Input	

Registrazione utente - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Nome	Semplice	Medio	Input	Non più di 100 caratteri
Cognome	Semplice	Medio	Input	Non più di 100 caratteri
Telefono	Semplice	Alto	Input	Non più di 20 caratteri
Username	Semplice	Alto	Input	Non più di 20 caratteri
Password	Semplice	Alto	Input	R2NF, non più di 50 caratteri

Gestione Utente - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Staff scelto	Semplice	Medio	Input	

Crea staff - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Nome	Semplice	Basso	Input	Non più di 100 caratteri
Codice di accesso	Semplice	Alto	Input	R3NF, Non più di 50 caratteri

Accedi staff - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Nome	Semplice	Basso	Input	Non più di 50 caratteri
Codice di accesso	Semplice	Alto	Input	R3NF, Non più di 50 caratteri

Cambia password - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Vecchia password	Semplice	Alto	Input	R2NF, Non più di 50 caratteri
Nuova password	Semplice	Alto	Input	R2NF, Non più di 50 caratteri

Gestione amministrazione staff - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Evento scelto	Semplice	Medio	Input	

Visualizzazione statistiche personali - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Entrate svolte	Complesso	Basso	Output	
Composto da:				
Nome evento	Semplice	Basso	Output	
Numero entrate	Semplice	Basso	Output	
Guadagni	Complesso	Basso	Output	
Composto da:				
Nome evento	Semplice	Basso	Output	
guadagni totali	Semplice	Basso	Output	

Gestione vendita prevendita - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Prevendita	Complesso	Medio	Input	R23F, R9NF, R10NF, R14NF, R15NF, R17NF, R19NF, R27NF
Composto da:				
Nome cliente	Semplice	Medio	Input	
Cognome cliente	Semplice	Medio	Input	
Evento associato	Semplice	Medio	Input	
Tipologia prevendita associata	Semplice	Medio	Input	
PR Associato	Semplice	Medio	Input	
Evento scelto	Semplice	Medio	Input	

Gestione entrata - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Prevendita	Complesso	Medio	Input	R23F, R9NF, R10NF, R14NF, R15NF, R17NF, R19NF, R27NF
Composto da:				
Nome cliente	Semplice	Medio	Input	
Cognome cliente	Semplice	Medio	Input	
Evento associato	Semplice	Medio	Input	
Tipologia prevendita associata	Medio	Input		
PR Associato	Medio	Input		
Evento scelto	Semplice	Medio	Input	

Gestione membri - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Informazioni membro	Complesso	Alto	Input	R6F
Composto da:				
Membro associato	Semplice	Alto	Input	
Ruoli	Semplice	Alto	Input	
Evento scelto	Semplice	Medio	Input	

Cambia codice accesso - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Nome dello staff	Semplice	Basso	Input	
Vecchio codice di accesso	Semplice	Alto	Input	R3NF, Non più di 50 caratteri
Nuovo codice di accesso	Semplice	Alto	Input	R3NF, Non più di 50 caratteri

Visualizzazione statistiche - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Entrate evento	Complesso	Basso	Output	
Composto da:				
Nome evento	Semplice	Basso	Output	
Entrate totali	Semplice	Basso	Output	
Guadagni evento	Complesso	Basso	Output	
Composto da:				
Nome evento	Semplice	Basso	Output	
Guadagni totali	Semplice	Basso	Output	
Entrate membro	Complesso	Basso	Output	
Composto da:				
Membro associato	Semplice	Basso	Output	
Entrate totali	Semplice	Basso	Output	
Guadagni membro	Complesso	Basso	Output	
Composto da:				
Membro associato	Semplice	Basso	Output	
Guadagni totali	Semplice	Basso	Output	
Evento scelto	Semplice	Medio	Input	

Gestione eventi - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Evento **Composto da:**	Complesso	Medio	Input	R18F, R19F
Nome	Semplice	Medio	Input	
Descrizione	Semplice	Medio	Input	
Periodo temporale	Semplice	Medio	Input	
Luogo	Semplice	Medio	Input	
Evento scelto	Semplice	Medio	Input	

Gestione tipologie prevendite - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Tipologia prevendita **Composto da:**	Complesso	Medio	Input	R20F
Nome	Semplice	Medio	Input	
Descrizione	Semplice	Medio	Input	
Periodo temporale	Semplice	Medio	Input	
Prezzo	Semplice	Medio	Input	
Evento scelto	Semplice	Medio	Input	

Autenticazione - Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / Privacy	Input/Output	Vincoli
Username	Semplice	Alto	Input	Non più di 20 caratteri
Password	Semplice	Alto	Input	R2NF, non più di 50 caratteri

3.2 Analisi dei vincoli

3.2.1 Tabella dei vincoli

Requisito	Categoria	Impatto	Funzionalità
Facile navigabilità delle schermate (R21NF	Usabilità	Cercare di migliorare	Gestione vendite prevendita, Gestione entrata, Autenticazione
Protezione dei dati (R22NF, R23NF)	Sicurezza	Peggiorano tempo di risposta, migliorano la privacy dei dati	Gestione log, Scrittura log, Sblocco utente, Registrazione utente, Crea staff, Accedi staff, Cambia password, Visualizzazione statistiche personali, Gestione vendita prevendita, Gestione entrata, Gestione membri, Cambia codice accesso, Visualizzazione statistiche, Gestione eventi, Gestione tipologie prevendite, Autenticazione
Controllo accessi (R24NF)	Sicurezza	Peggiorano tempo di risposta e usabilità, migliorano la privacy dei dati	Gestione utente, Gestione amministrazione

3.3 Analisi delle interazioni

3.3.1 Tabella delle maschere

Maschera	Informazioni	Funzionalità
View Autenticazione	Username, password	Autenticazione
Home Gestione Amministrazione	Navigabilità delle funzioni amministratore di sistema	Gestione Amministrazione
Home Gestione log	Navigabilità delle funzioni gestione log: aggiungi log e leggi log	Gestione log
View Aggiungi log	Timestamp, descrizione, livello	Gestione log
View Leggi log	Timestamp, descrizione, livello	Gestione log
View Sblocco utente	Dati utente da sbloccare	Sblocco utente
View Registrazione utente	Nome, cognome, telefono, username, password	Registrazione utente
Home Gestione utente	Navigabilità delle funzioni utente, scelta staff	Gestione utente
View Crea staff	Nome, codice di accesso	Crea staff
View Accedi staff	Nome, codice di accesso	Accedi staff
View Cambia password	Vecchia password, nuova password	Cambia password
Home Gestione staff	Consente la navigabilità delle funzioni del membro, PR, cassiere e amministratore, scelta evento	Gestione amministrazione staff, Gestione entrata, Gestione vendita prevendita, Visualizza statistiche personali
View Visualizzazione statistiche personali	Entrate svolte, guadagni, filtri dei dati per evento, staff e totale	Visualizzazione statistiche personali
Home Gestione vendita prevendita	Navigabilità tra le funzioni del PR	Gestione vendita prevendita
View Aggiungi prevendita	Informazioni cliente, evento prevendita, tipologia prenvendita, documento finale	Gestione vendita prevendita
View Lista prevendite	Informazioni cliente, evento e tipologia, filtri in base al cliente e all'evento	Gestione vendita prevendita
View Annulla prevendita	Informazioni prevendita per l'annullamento	Gestionve vendita prevendita
Home Gestione entrata	Fornisce la navigabilità tra le funzioni del Cassiere	Gestione entrata
View Timbra entrata	Documento digitale	Gestione entrata
View Lista entrate	Entrata, filtri per evento, staff	Gestione entrata
Home gestione membri	Fornisce la navigabilità tra le funzioni di gestione dei membri da parte dell'amministratore	Gestione membri
View lista membri	Informazioni sui membri	Gestione membri
View Modifica ruoli membro	Informazioni del membro da modficare, nuovi ruoli	Gestione membri
View Rimuovi membro	Informazioni del membro da eliminare	Gestione membri
View Cambia codice di accesso	Nome staff, vecchio codice, nuovo codice	Cambia codice accesso
View Visualizzazione statistiche	Informazioni membro, entrate effettuate, guadagni, informazioni evento, entrate evento, guadagni evento, filtro per membro o per evento	Visualizzazione statistiche

Maschera	Informazioni	Funzionalità
Home Gestione eventi	Fornisce la navigabilità tra le funzioni di gestione eventi da parte dell'amministratore	Gestione eventi
View Aggiungi evento	Nome, descrizione, periodo temporale e luogo di svolgimento	Gestione eventi
View Lista eventi	Nome, descrizione, periodo temporale e luogo di svolgimento	Gestione eventi
View Modifica evento	Informazioni dell'evento da modificare, modifiche da effettuare	Gestione eventi
View Annulla evento	Informazioni dell'evento da annullare	Gestione eventi
Home Gestione tipologie prevendite	Fornisce la navigabilità tra le funzioni di gestione tipologie prevendite da parte dell'amministratore	Gestione tipologie prevendite
View Aggiungi tipologia prevendita	Nome, descrizione, periodo temporale di vendita e prezzo	Gestione tipologie prevendite
View Lista tipologie prevendite	Nome, descrizione, periodo temporale di vendita e prezzo	Gestione tipologie prevendite
View Modifica tipologia prevendita	Informazioni della tipologia prevendita da modificare, modifiche da effettuare	Gestione tipologie prevendite
View Elimina tipologia prevendita	Informazioni della tipologia prevendita da eliminare	Gestione tipologie prevendite

3.4 Analisi dei ruoli e responsabilità

3.4.1 Tabella dei ruoli

Ruolo	Responsabilità	Maschere	Riservatezza	Numerosità
Utente	Può creare uno staff, e accedere ad altri staff.	View Autenticazione, Home Gestione utente, View Crea staff, View Accedi staff, View Cambia password	Alto grado di riservatezza	Il numero può essere gestito da un amministratore di sistema, al più è limitato dalle risorse del sistema.
Amministratore di sistema	Gestisce il log, lo sblocco di utenti bloccati e la registrazione di un nuovo utente.	View Autenticazione, Home Gestione Amministrazione, Home gestione log, View Aggiungi log, View Leggi Log, View Sblocco utente, View Registrazione utente	Massimo grado di riservatezza	Per ogni installazione software è sufficiente un amministratore di sistema.
Membro di uno staff	Può ricavare informazioni sullo staff, può leggere le statistiche ad esso attribuite.	Home gestione staff, View Visualizzazione statistiche personali	Alto grado di riservatezza	Numero massimo limitato dalle risorse del sistema.
Cassiere di uno staff	Può effettuare tutte le operazioni di un membro e in più gestisce le entrate di un evento.	Home gestione staff, View Visualizzazione statistiche personali, Home Gestione entrata, View Timbra entrata, View lista entrate	Alto grado di riservatezza	Numero massimo limitato dalle risorse del sistema.
PR di uno staff	Può effettuare tutte le operazioni di un membro e in più gestisce la vendita di prevendite.	Home gestione staff, View Visualizzazione statistiche personali, Home Gestione vendita prevendita, View Aggiungi prevendita, View Lista prevendite, View Annulla prevendita	Alto grado di riservatezza	Numero massimo limitato dalle risorse del sistema.
Amministratore di uno staff	Può effettuare tutte le operazioni di un membro e in più gestisce lo staff.	Home gestione staff, View Visualizzazione statistiche personali, Home Gestione membri, View lista membri, View Modifica ruoli membro, View Rimuovi membro, View Cambia codice di accesso, View Visualizzazione statistiche, Home Gestione eventi, View Aggiungi evento, View lista eventi, View Modifica evento, View Annulla evento, Home gestione tipologie prevendite, View Aggiungi tipologia prevendita, View lista tipologie prevendite, View modifica tipologia prevendita, View elimina tipologia prevendita	Alto grado di riservatezza	Numero massimo limitato dalle risorse del sistema.

3.4.2 Tabelle Ruolo-Informazioni

Utente: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Password personale	Scrittura
Staff	Lettura/Scrittura

Amministratore di sistema: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Password personale	Scrittura
Utente	Lettura/Scrittura
Log	Lettura/Scrittura

Membro di uno staff: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Informazioni sullo staff	Lettura
Statistiche personali	Lettura

Cassiere staff: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Informazioni sullo staff	Lettura
Statistiche personali	Lettura
Prevendite	Lettura
Entrate	Lettura/Scrittura
Tipologie prevendita	Lettura
Evento	Lettura

PR di uno staff: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Informazioni sullo staff	Lettura
Statistiche personali	Lettura
Prevendite	Lettura/Scrittura
Tipologie prevendita	Lettura
Evento	Lettura

Amministratore di uno staff: Tabella Ruolo-Informazioni

Informazione	Tipo di accesso
Dati personali	Lettura
Informazioni sullo staff	Lettura
Statistiche personali	Lettura
Evento	Lettura/Scrittura
Tipologie Prevendite	Lettura/Scrittura
Statistiche	Lettura
Codice di accesso	Scrittura
Membri dello staff	Lettura/Scrittura
Ruoli di un membro	Lettura/Scrittura

3.5 Scomposizione del problema

Funzionalità	Scomposizione
Gestione Amministrazione	Gestione log, Sblocco utente, Registrazione utente
Gestione log	Leggi log, Aggiungi log
Gestione utente	Crea staff, Accedi staff, Cambia password
Gestione Amministrazione staff	Gestione membri, Cambia codice accesso, Visualizzazione statistiche, Gestione eventi, Gestione tipologie prevendita
Gestione vendita prevendita	Aggiungi prevendita, Lista prevendite, Annulla prevendita
Gestione entrata	Timbra entrata, Lista entrate
Gestione membri	Lista membri, Modifica ruoli membro, Rimuovi membro
Gestione eventi	Aggiungi evento, Lista eventi, Modifica evento, Annulla evento
Gestione tipologie prevendita	Aggiungi tipologia prevendita, Lista tipologie prevendita, Modifica tipologia prevendita, Rimuovi tipologia prevendita

Gestione Amministrazione: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Registrazione utente	Sblocco utente	Si può sbloccare un utente solo se è registrato	Username

Gestione log: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Aggiungi log	Leggi log	Si può leggere un record solo se è stato aggiunto	Record

Gestione utente: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Crea staff	Accedi staff	Si può accedere ad uno staff solo se è stato creato	Nome staff

Gestione amministrazione staff: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Gestione tipologie prevendita	Gestione eventi	Non posso creare una tipologia prevendita con un periodo di vendita durante o dopo l'evento (R9NF)	Evento

Gestione vendita prevendita: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Aggiungi prevendita	Lista prevendite	La lista delle prevendite è condizionata dalle prevendite aggiunte	Prevendita
Aggiungi prevendita	Annulla prevendita	Si può annullare una prevendita solo se è stata aggiunta	Prevendita

Gestione entrata: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Timbra Entrata	Lista entrate	La lista delle entrate è condizionata dalle entrate aggiunte	Entrata

Gestione membri: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Lista membri	Rimuovi membro	Posso eliminare un membro solo se fa parte della lista, cioè dello staff	Membro

Gestione eventi: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Aggiungi evento	Lista eventi	La lista delle prevendite è condizionata dalle prevendite aggiunte	Prevendita
Aggiungi prevendita	Annulla prevendita	Si può annullare una prevendita solo se è stata aggiunta	Prevendita

Gestione tipologie prevendita: Tabella Sotto-Funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
Aggiungi tipologia prevendita	Lista tipologie prevendita	La lista delle tipologie prevendita è condizionata dalle tipologie prevendita aggiunte	Tipologia prevendita
Aggiungi tipologie prevendita	Annulla tipologie prevendita	Si può annullare una tipologie prevendita solo se è stata aggiunta	Tipologia prevendita

3.6 Creazione del modello del dominio

Analizzando il vocabolario si evidenzia la differenza tra utente normale e amministratore di sistema, e i vari ruoli di un membro, che possono essere cassiere, pr o amministratore.
Dall'analisi del vocabolario, rivisto dopo l'analisi di sicurezza, si evidenzia anche il log, con i livelli di importanza: basso, medio, alto.
Qui sotto il modello del dominio:

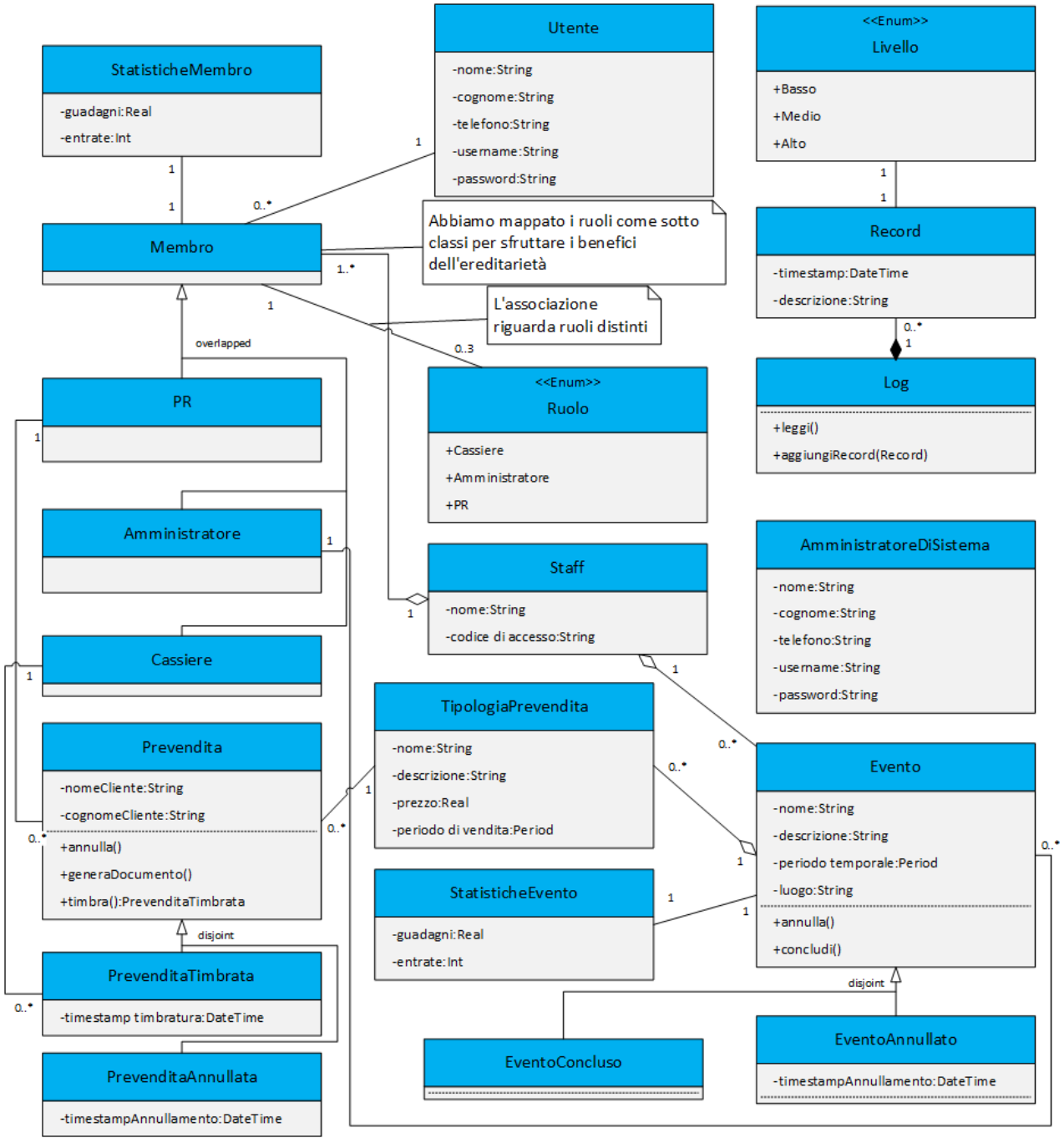

Figura 4: modello_dominio.vsdx

Le credenziali dovranno essere gestite in maniera sicura, nella fase di progettazione. L'aggiunta del **telefono** alle entità Utente e AmministratoreDiSistema garantisce flessibilità per una futura implementazione **OTP**.

3.7 Architettura logica

3.7.1 Diagramma dei package

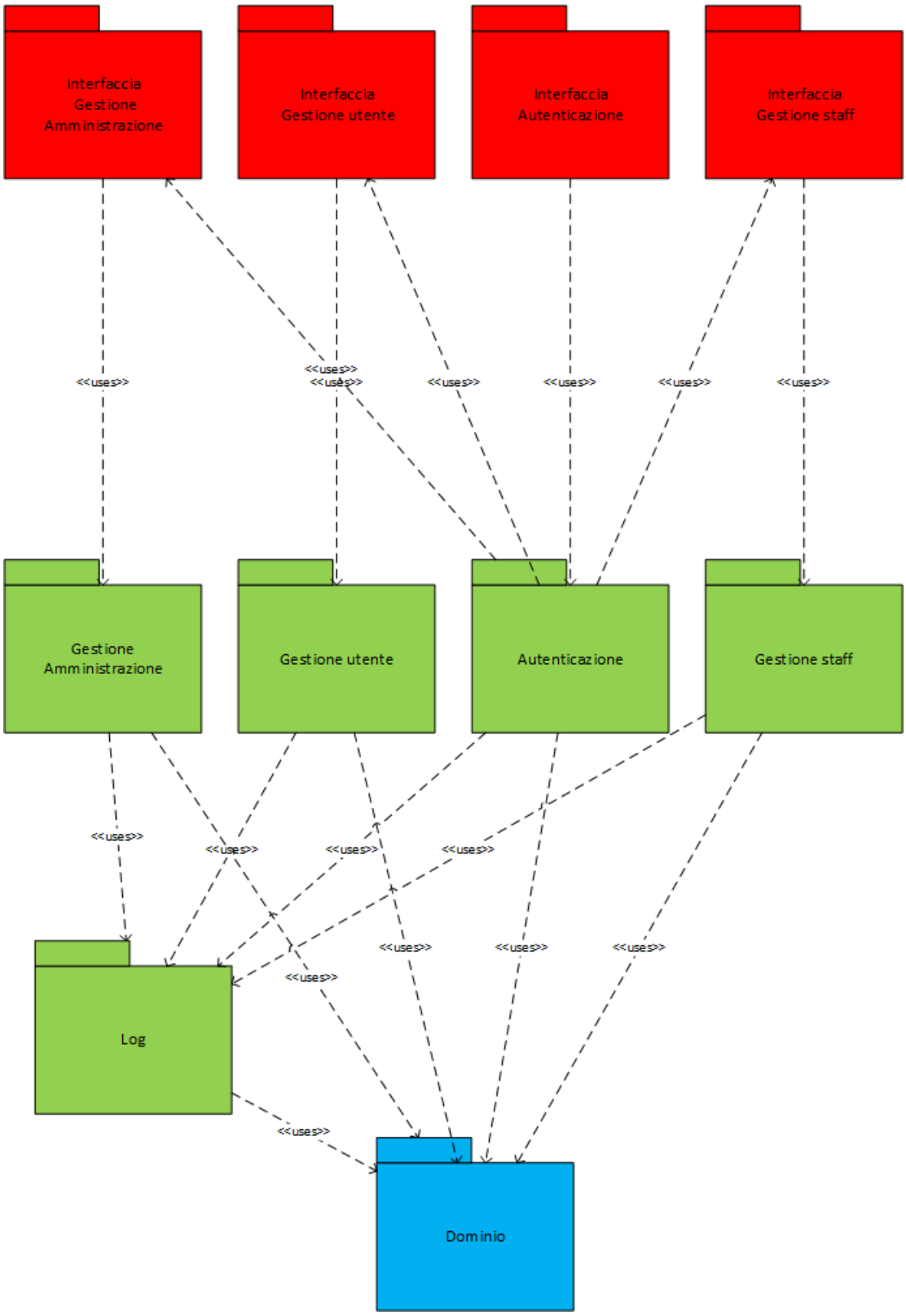

Figura 5: diagramma_package.vsdx

3.7.2 Diagramma delle classi

Diagramma delle classi: Gestione Amministrazione

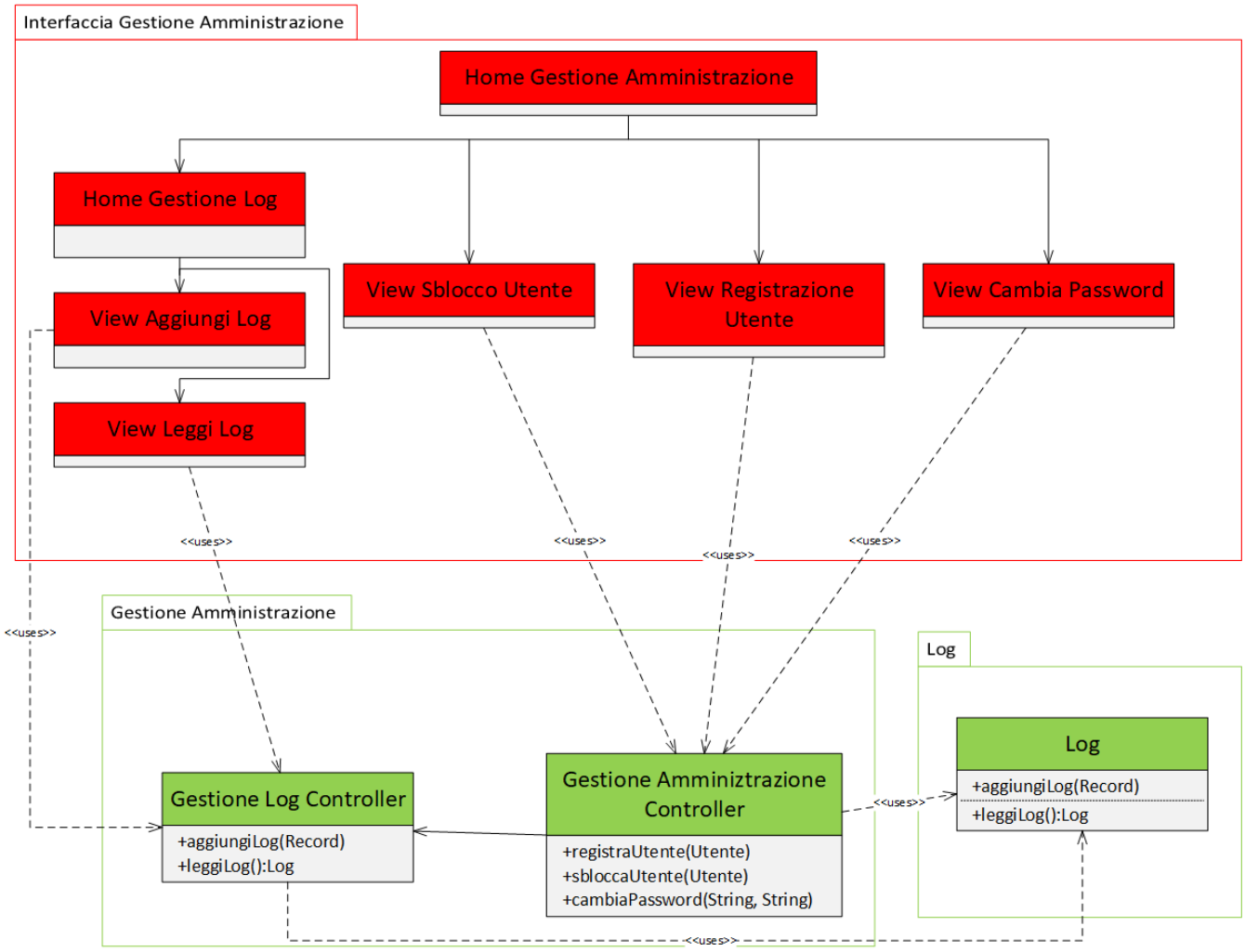

Figura 6: diagramma_classi_gestione_amministrazione.vsdx

Gestione Amministrazione Controller si occupa di gestire le operazioni associate all'amministratore di sistema. Abbiamo ritenuto utile separare **Gestione Amministrazione Controller** e **Gestione Log Controller** per rispettare la scomposizione del problema vista precedentemente.

Diagramma delle classi: Gestione Utente

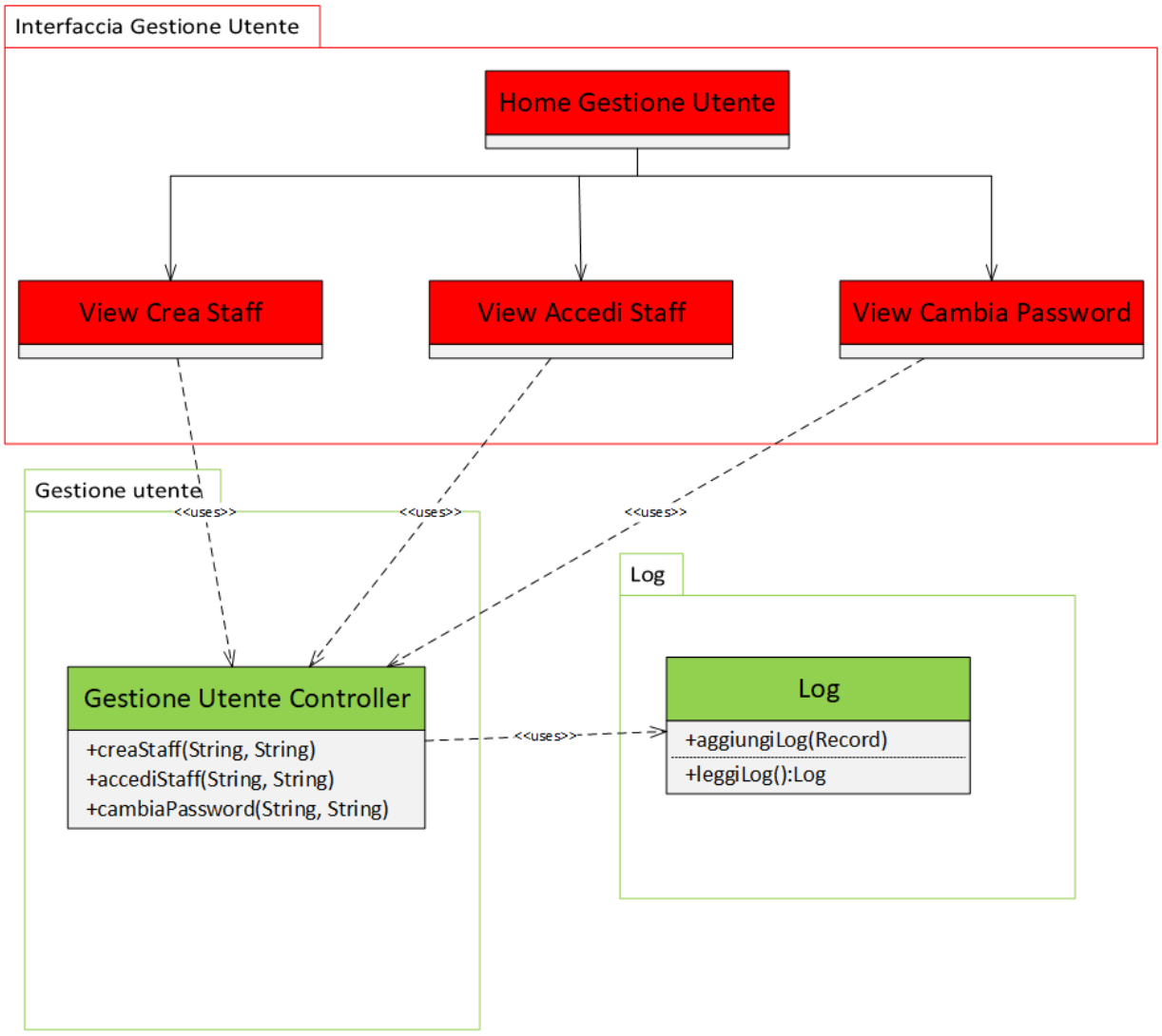

Figura 7: diagramma_classi_gestione_utente.vsdx

Gestione Utente Controller si occupa di gestire le operazioni associate agli utenti.

Diagramma delle classi: Gestione Staff

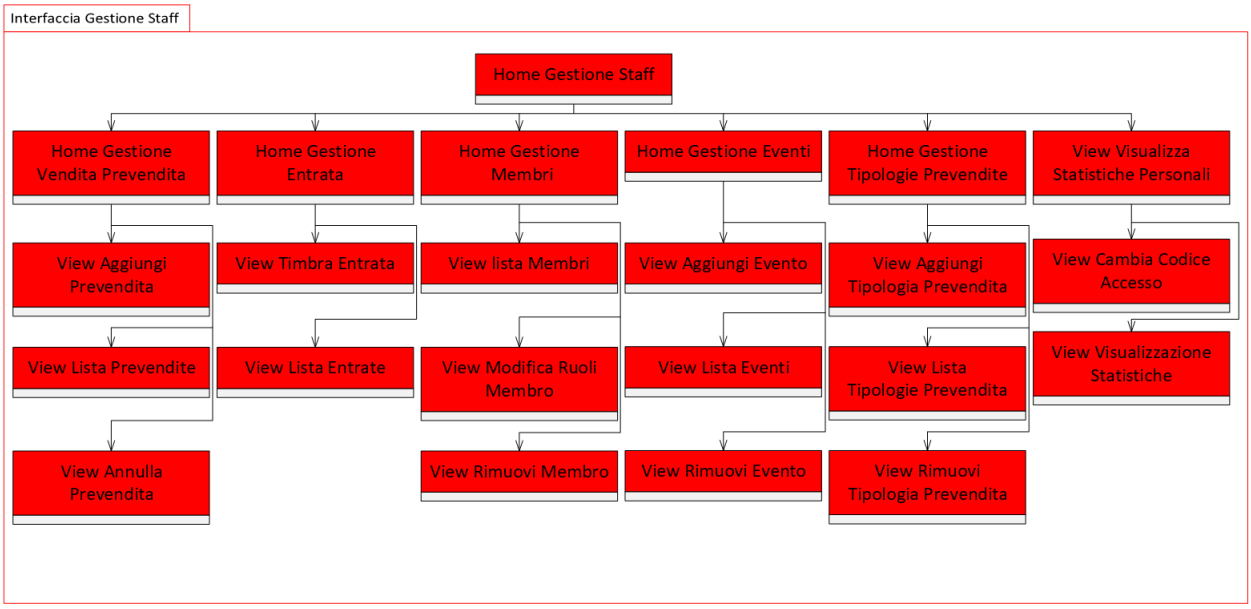

Figura 8: diagramma_classi_gestione_staff_interfacce.vsdx

Le interfacce sono molte, visto le operazioni effettuabili nel sistema gestionale. Abbiamo separato le interfacce dai controller per evitare di confondere la lettura. Le interfacce sono comunque collegate ai relativi controller.

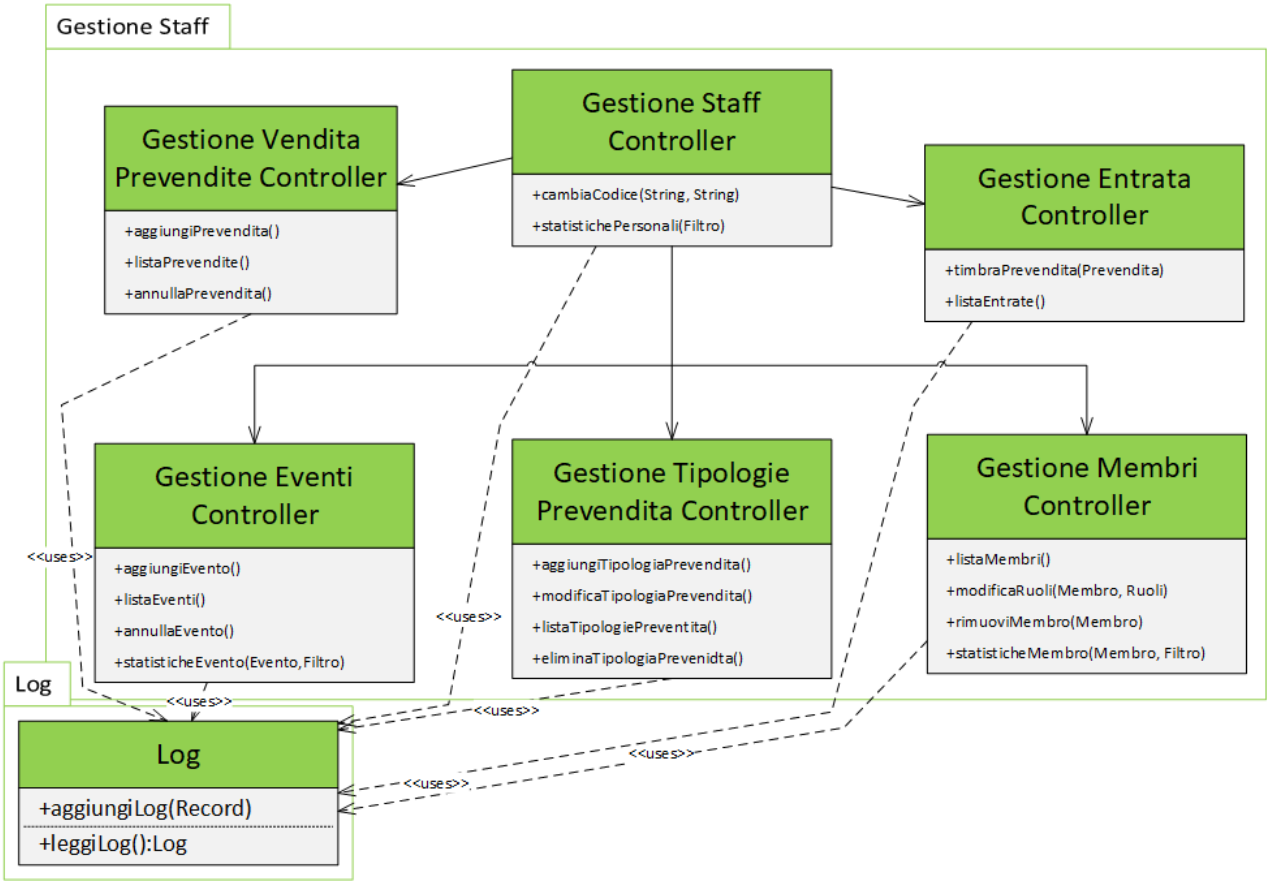

Figura 9: diagramma_classi_gestione_staff.vsdx

Gestione Staff Controller si occupa della parte di gestione dello staff. Usato dai membri di uno staff per le varie operazioni, è stato diviso in base alle interfacce.

Diagramma delle classi: Autenticazione

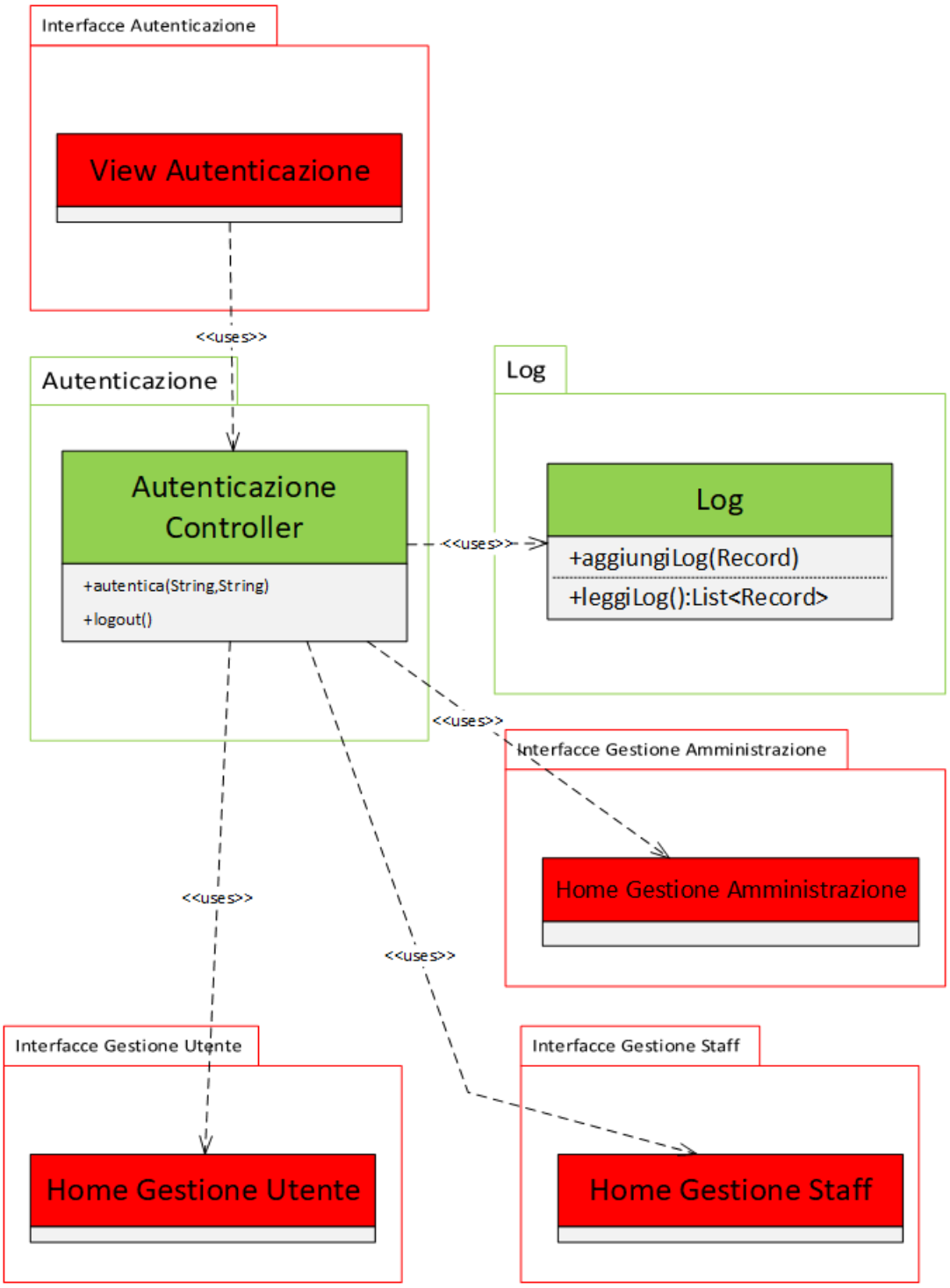

Figura 10: diagramma_classi_autenticazione.vsdx

Autenticazione Controller gestisce la sessione di un utente; permette di tenere traccia anche del contesto attuale dell'utente. Per esempio, nella fase progettuale si potrebbe sfruttare per tenere traccia dello staff e dell'evento in uso dall'utente, semplificando la presentazione.

3.7.3 Interazione

Saranno riportati solo i diagrammi di sequenza ritenuti principali per il sistema di gestione prevendite.

Diagramma di sequenza: Autenticazione con successo

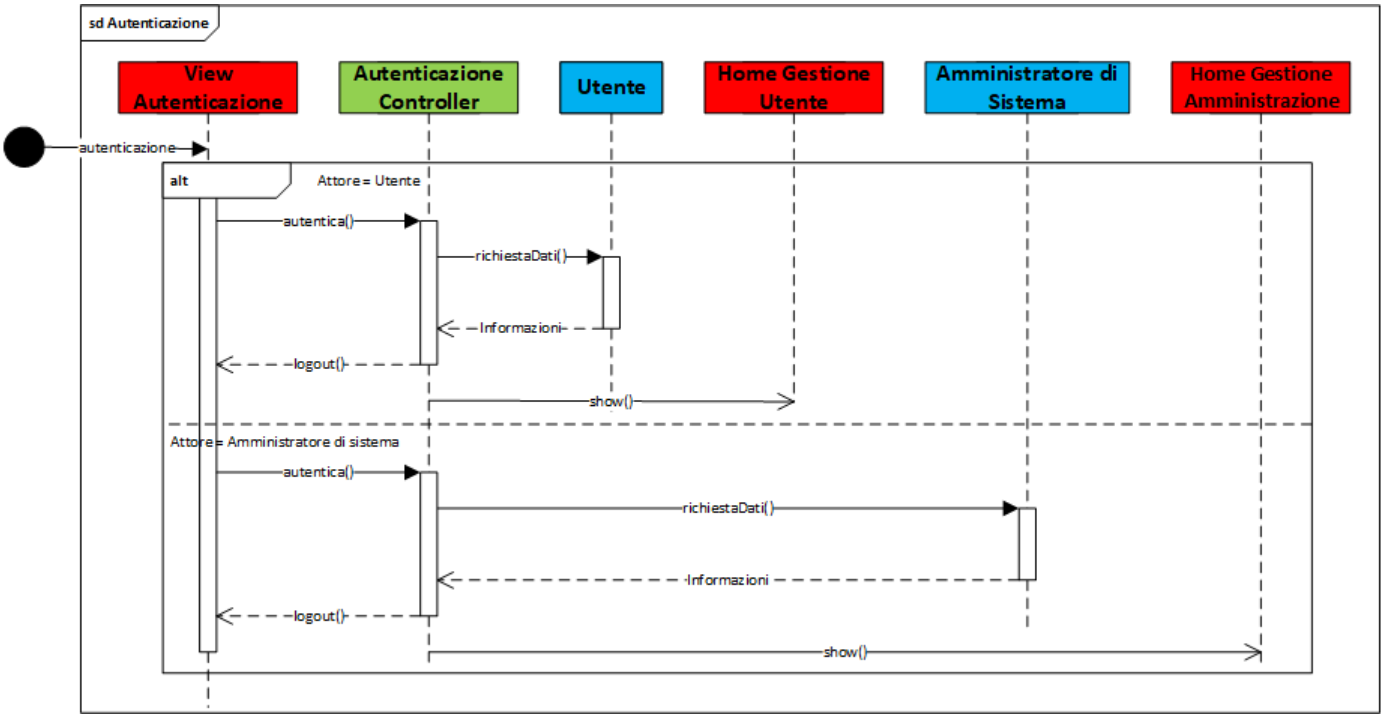

Figura 11: diagramma_sequenza_autenticazione.vsdx

Una volta effettuato il logout si richiede di nuovo l'autenticazione tramite **View Autenticazione**. Effettuato l'accesso, si passa alla schermata di **Home Gestione Utente** o **Home Gestione Amministrazione** in caso di amminstratore di sistema.

Diagramma di sequenza: Registrazione utente

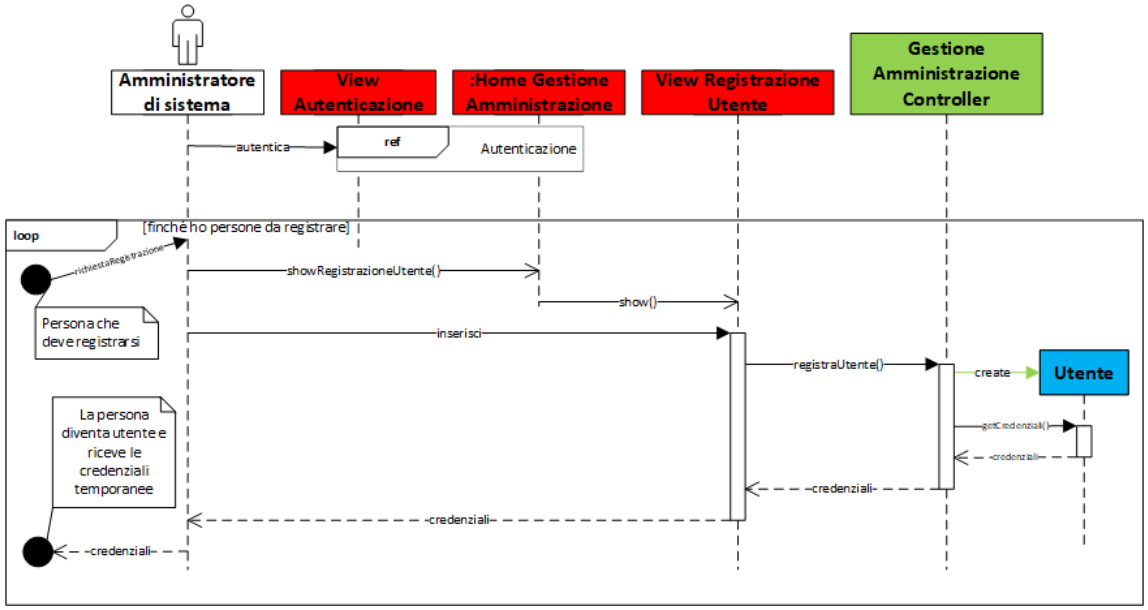

Figura 12: diagramma_sequenza_registrazione_utente.vsdx

Prima della registrazione la persona non è nota al sistema e non può interagire con esso direttamente. Abbiamo inserito un **loop** perché probabilmente la registrazione avviene per staff, evitando l'autenticazione ripetuta.

Diagramma di sequenza: Creazione evento

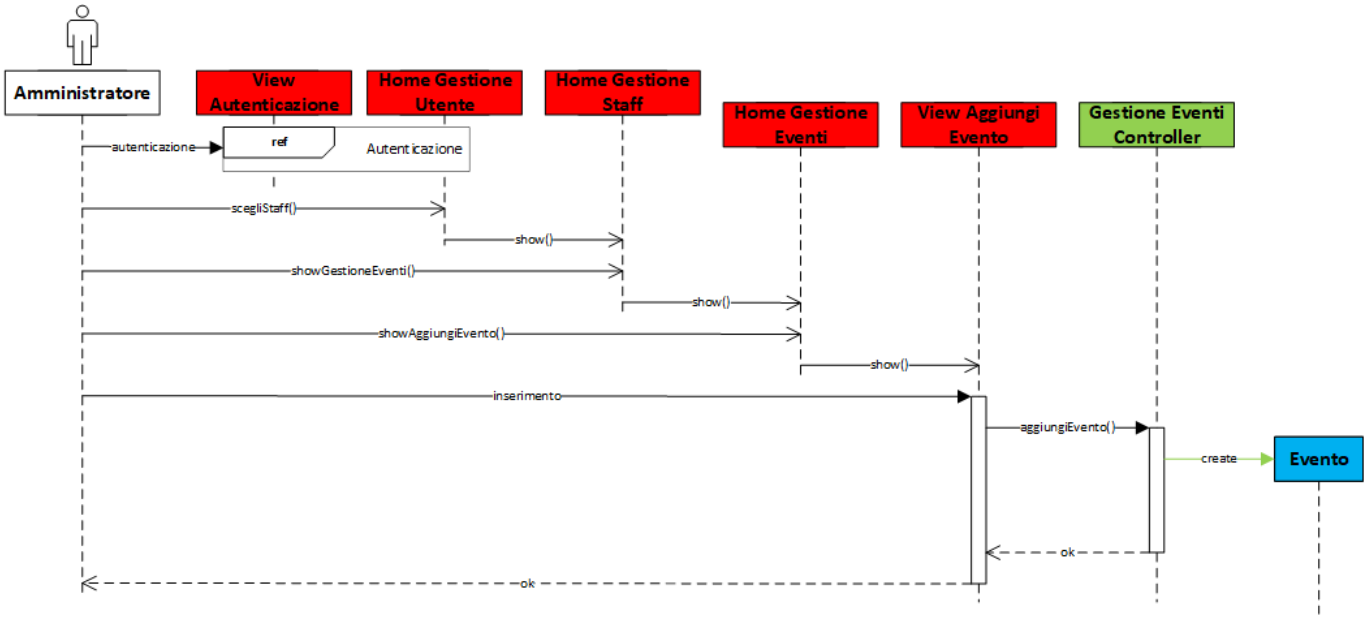

Figura 13: diagramma_sequenza_creazione_evento.vsdx

Diagramma di sequenza: Vendita prevendita

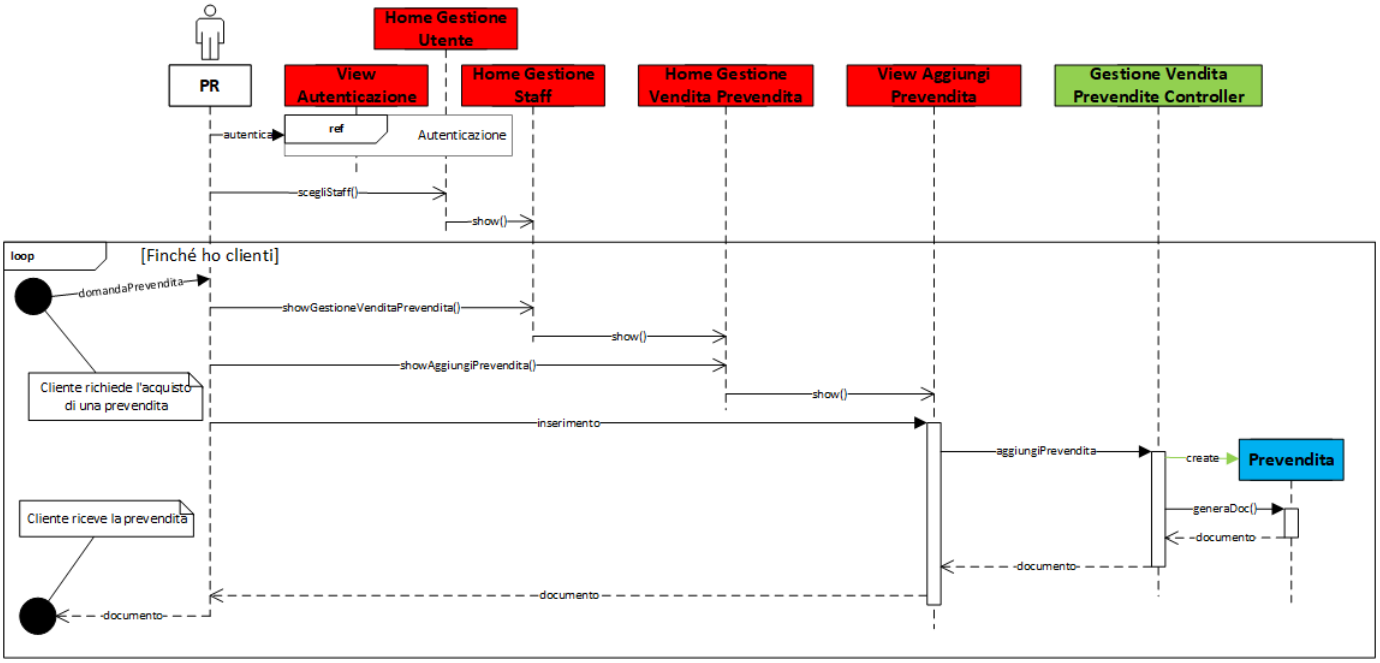

Figura 14: diagramma_sequenza_vendita_prevendita.vsdx

Il cliente non può interfacciarsi direttamente al sistema, infatti è richiesto il PR per la gestione della vendita della prevendita. Quando il PR ha interagito con il sistema, consegna al cliente il **documento digitale** che rappresenta la prevendita. Abbiamo deciso di aggiungere un **loop** perché è probabile che il PR venda ad un gruppo di persone alla volta, evitando di effettuare l'autenticazione ogni volta.

Diagramma di sequenza: Ingresso evento

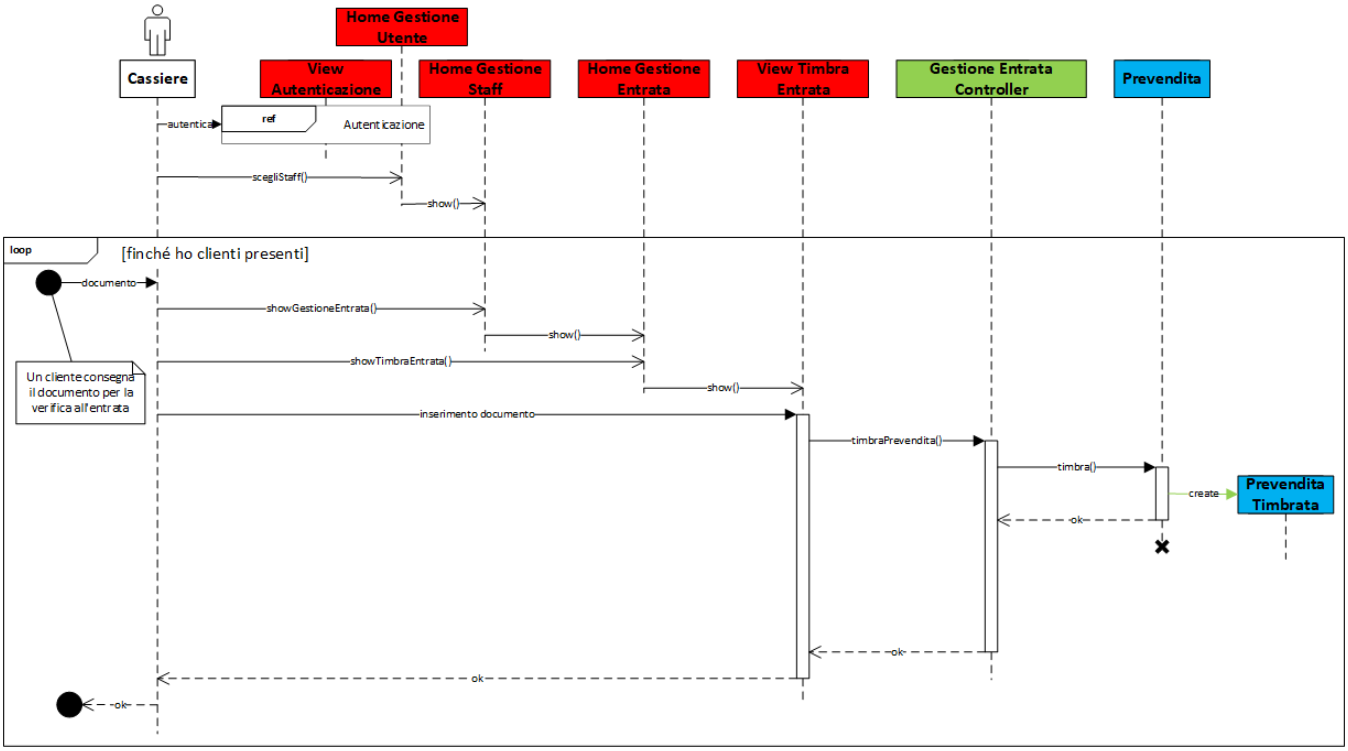

Figura 15: diagramma_sequenza_ingresso_evento.vsdx

Il cassiere **ripete** l'operazione di timbratura delle prevendite per tutta la durata necessaria durante l'evento. Qui il **loop** è d'obbligo data la natura del cassiere. Il documento permette di timbrare la prevendita, la quale si trasforma in **Prevendita Timbrata** come nel modello del dominio.

3.7.4 Comportamento

Dopo un'attenta analisi, abbiamo deciso di mostrare i diagrammi di stato riguardo un **evento** ed una **prevendita**, sempre basandoci sui requisiti.

Diagramma di Stato: Evento

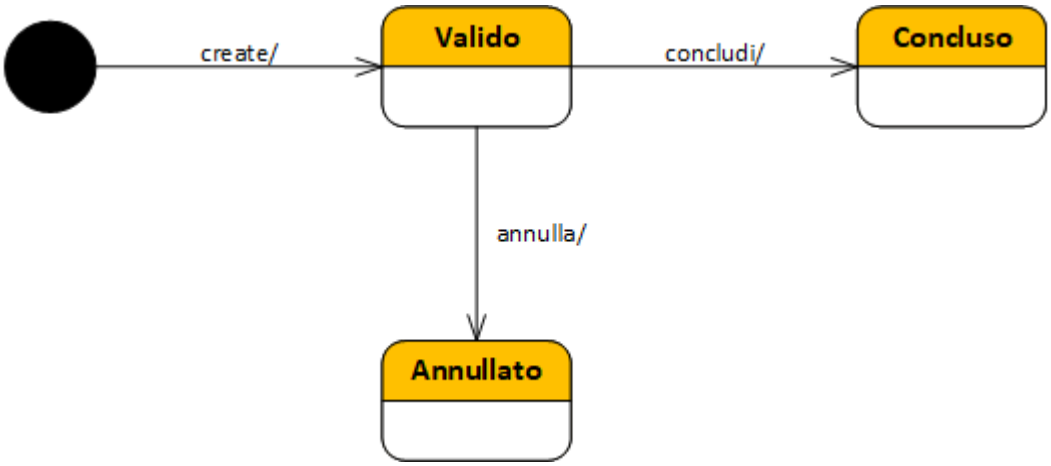

Figura 16: diagramma_stato_evento.vsdx

Secondo le specifiche R19F e R11NF, un evento può essere **annullato**, mentre lo stato **Concluso** si deduce dal requisito R18F in quanto un evento ha un periodo di svolgimento.

Diagramma di Stato: Prevendita

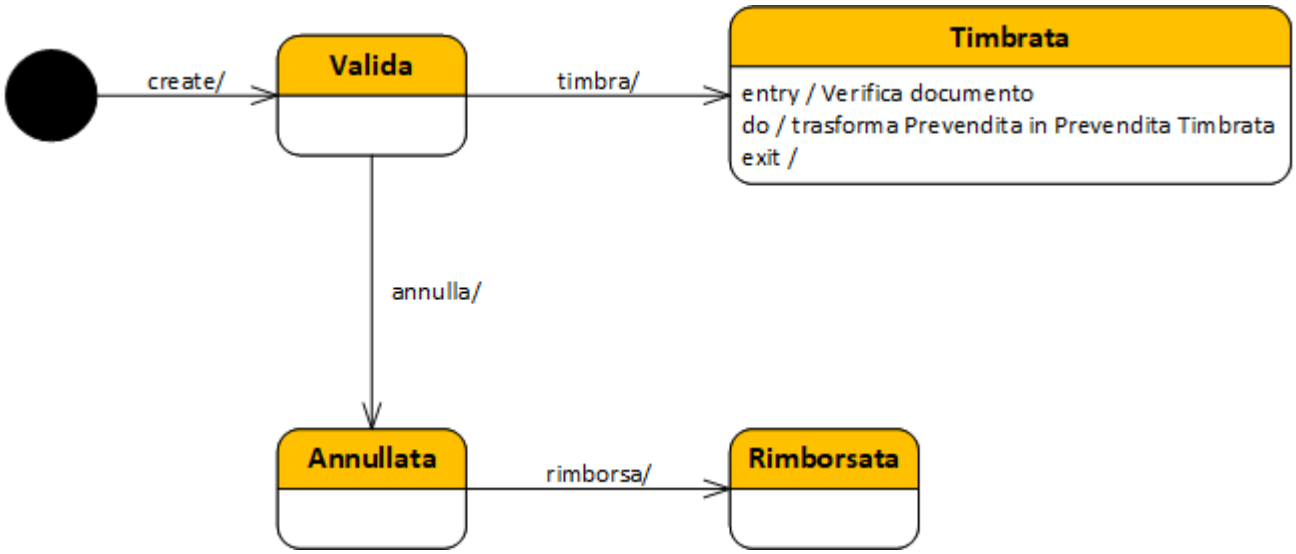

Figura 17: diagramma_stato_prevendita.vsdx

Secondo le specifiche R21F R10NF una prevendita può essere **annullata** e **rimborsata**, mentre dai requisiti R12F R20NF si deduce lo stato **Timbrata**, in accordo con il diagramma di sequenza di ingresso evento.

3.8 Piano del lavoro

Il lavoro di gruppo è stato diviso in base alle competenze dei membri del team. Si prevede un team per lo sviluppo del database e uno per una possibile applicazione middleware da inserire tra database e applicativo finale. **Tutti i dettagli nella tabelle sottostanti**:

Nome Team	**Composizione**
Team progettazione	Bartolomei, di Nuzzo, Veronesi
Team sviluppo DB	Bartolomei, Veronesi
Team sviluppo Middleware	Bartolomei, di Nuzzo
Team sviluppo front-end	di Nuzzo, Veronesi
Team sicurezza	Bartolomei, di Nuzzo, Veronesi

Package	**Progetto**	**Sviluppo**
Dominio	Team progettazione	Team sviluppo DB, Team sicurezza
GestioneAmministrazione	Team progettazione	Team sviluppo Middleware
GestioneUtente	Team progettazione	Team sviluppo Middleware
GestioneStaff	Team progettazione	Team sviluppo Middleware
Autenticazione	Team progettazione	Team sviluppo DB, Team sviluppo Middleware, Team sicurezza
Log	Team progettazione	Team sviluppo DB, Team sviluppo Middleware, Team sicurezza
InterfacciaGestioneAmministrazione	Team progettazione	Team sviluppo front-end
InterfacciaGestioneUtente	Team progettazione	Team sviluppo front-end
InterfacciaGestioneStaff	Team progettazione	Team sviluppo front-end
InterfacciaAutenticazione	Team progettazione	Team sviluppo front-end, Team sicurezza

Dopo una dovuta valutazione, i tempi di rilascio previsti sono i seguenti:

- **Progettazione**: entro 14 giorni dalla data odierna.
- **Sviluppo moduli con test unitari**: entro 21 giorni dalla fine di progetto odierna.
- **Integrazione e testing**: Entro 7 giorni dalla fine dello sviluppo dei moduli.

3.8.1 Sviluppi futuri

Il committente ha richiesto una flessibilità nell'applicazione per quanto riguarda **l'utilizzo di nuove tipologie dei documenti digitali**, in modo da andare incontro a nuove tecnologie applicabili. Per esempio una valida proposta, già discussa con il committente, è l'utilizzo di QR code come documento digitale. Non è da escludere che in futuro si possa usare una tecnologia alternativa per migliorare la qualità del servizio. L'introduzione di un numero di telefono per gli utenti e per l'amministratore di sistema, potrebbe permettere l'introduzione di un'**autenticazione a due fattori** con password **OTP**.

3.9 Piano di collaudo

Per garantire il funzionamento del sistema sono necessari una gamma di test unitari e di integrazione che permettono di verificare la correttezza delle singole parti.

```java
package test.dominio;

import org.junit.Before;
import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class TestEvento {

    private Evento evento;

    //La classe rappresenta un periodo temporale.
    //Nella implementazione potrebbe variare
    private PeriodoTemporale periodoTemporale;

    //Classi aggiunte che sono associate a Evento, anche indirettamente
    private Staff staff;
    private Staff staff2;

    private PeriodoTemporale periodoTemporale2;

    /**
     * Inizializzazione di una prova di Evento.
     */
    @Before
    public void utenteSetup(){
        staff = new Staff("Staff1", "12345");
        periodoTemporale = new PeriodoTemporale("03/05/2020 16:00", "03/05/2020 18:00")
            ;
        evento = new Evento("Evento1", "Desc", periodoTemporale, "Bologna", staff);

        staff2 = new Staff("Staff2", "123456");
        periodoTemporale2 = new PeriodoTemporale("03/05/2020 15:00", "03/05/2020 15:30"
            );
    }


    /**
     * Test dei getter di Evento.
     */
    @Test
    public void gettersTest(){
        assertEquals("Evento1", evento.getNome());
        assertEquals("Desc", evento.getDescrizione());
        assertEquals(periodoTemporale, evento.getPeriodoTemporale());
        assertEquals("Bologna", evento.getLuogo());

        //Questo Equals va testato con classe di test a parte.

        assertEquals(staff, evento.getStaffAssociato());

    }

    /**
     * Test dei setter di Evento.
     * Se nella progettazione viene descritta come classe immutabile, ignorare questo
         test.
     */
    @Test
    public void settersTest(){
        evento.setNome("Evento2");
        evento.setDescrizione("Desc2");
        evento.setPeriodoTemporale(periodoTemporale2);
```

```
61          evento.setLuogo("Milano");
62          evento.setStaff(staff2);
63
64          assertEquals("Evento2", evento.getNome());
65          assertEquals("Desc2", evento.getDescrizione());
66          assertEquals(periodoTemporale2, evento.getPeriodoTemporale());
67          assertEquals("Milano", evento.getLuogo());
68
69          //Questo Equals va testato con classe di test a parte.
70
71          assertEquals(staff2, evento.getStaffAssociato());
72
73      }
74
75
76  }
```

```
1   package test.dominio;
2
3   import org.junit.Before;
4   import org.junit.Test;
5
6   import static org.junit.Assert.assertEquals;
7
8   public class TestPrevendita {
9
10      private Prevendita prevendita;
11
12      //Classi aggiunte che sono associate a Prevendita, anche indirettamente
13      private Utente utente;
14
15      private Staff staff;
16      private PR pr;
17
18      private Evento evento;
19      private TipologiaPrevendita tipologiaPrevendita;
20
21      //Sono usati nei setters.
22      private Utente utente2;
23      private PR pr2;
24
25      private Evento evento2;
26      private TipologiaPrevendita tipologiaPrevendita2;
27
28
29      /**
30       * Inizializzazione di una prova di Prevendita.
31       * Sono necessari molte classi accessorie per creare una istanza di Prevendita.
32       */
33      @Before
34      public void utenteSetup(){
35          utente = new Utente("Marcello", "Rossi", "+39333222154", "pippo", "123456");
36          staff = new Staff("Staff1", "12345");
37          pr = new PR(utente, staff, new Ruolo[]{Ruolo.PR});
38
39          utente2 = new Utente("Mario", "Rossi", "+39333212154", "pippo2", "1234567");
40          pr2 = new PR(utente2, staff, new Ruolo[]{Ruolo.PR});
41
42          evento = new Evento("Evento1", "Desc",
43                  new PeriodoTemporale("03/05/2020 16:00", "03/05/2020 18:00"), "
                        Bologna", staff);
44
45          tipologiaPrevendita = new TipologiaPrevendita("Tip1", "Desc2", 10.0,
46                  new PeriodoTemporale("03/05/2020 13:00", "03/05/2020 15:00"),
                        evento);
47
48          evento2 = new Evento("Evento2", "Desc",
```

```
49                      new PeriodoTemporale("03/05/2020 16:00", "03/05/2020 18:00"), "
                            Bologna", staff);
50          tipologiaPrevendita2 = new TipologiaPrevendita("Tip2", "Desc3", 10.0,
51                      new PeriodoTemporale("03/05/2020 13:00", "03/05/2020 15:00"),
                            evento2);
52
53          prevendita = new Prevendita("Luciano", "Rossi", pr, tipologiaPrevendita);
54      }
55
56
57      /**
58       * Test dei getter di Prevendita.
59       */
60      @Test
61      public void gettersTest(){
62          //Equals Nome Cliente
63          assertEquals("Luciano", prevendita.getNomeCliente());
64          //Equals Cognome Cliente
65          assertEquals("Rossi", prevendita.getCognomeCliente());
66
67          //Questi equals sono da testare nelle classi di test apposite.
68
69          //Equals PR
70          assertEquals(pr, prevendita.getPRAssociato());
71          //Equals Tipologia Prevendita
72          assertEquals(tipologiaPrevendita, prevendita.getTipologiaPrevenditaAssociata())
                ;
73      }
74
75      /**
76       * Test dei setter di Prevendita.
77       * Se nella progettazione viene descritta come classe immutabile, ignorare questo
             test.
78       */
79      @Test
80      public void settersTest(){
81
82          prevendita.setNome("Franco");
83          prevendita.setCognome("Franco");
84          prevendita.setPR(pr2);
85          prevendita.setTipologiaPrevendita(tipologiaPrevendita2);
86
87          //Equals Nome Cliente
88          assertEquals("Franco", prevendita.getNomeCliente());
89          //Equals Cognome Cliente
90          assertEquals("Franco", prevendita.getCognomeCliente());
91
92          //Questi equals sono da testare nelle classi di test apposite.
93
94          //Equals PR
95          assertEquals(pr2, prevendita.getPRAssociato());
96          //Equals Tipologia Prevendita
97          assertEquals(tipologiaPrevendita2, prevendita.getTipologiaPrevenditaAssociata()
                );
98      }
99
100
101 }
```

4 Progettazione

4.1 Progettazione architetturale

4.1.1 Requisiti non funzionali

Dall'analisi dei requisiti sono emersi alcuni vincoli non funzionali, tra cui:

- Integrità dati
- Sicurezza dati, anche nella comunicazione
- Controllo accessi
- Controllo autorizzazioni
- Disponibilità del servizio
- Replicabilità del sistema
- **basso costo del prodotto software**
- Facilità d'utilizzo

L'integrità e la sicurezza dei dati, soprattutto per quanto riguarda la comunicazione, sono di fondamentale importanza, soprattutto in vista di una architettura 3-tier. Purtroppo, dato il basso costo di sviluppo software, è opportuno scegliere soluzioni standard esterne al sistema, come l'uso di protocolli di cifratura come il TLS, il quale viene utilizzato da protocolli più di alto livello come HTTPS, che abbiamo deciso di utilizzare data la facilità d'utilizzo e il vasto supporto. L'avvento di tecnologie di trasmissione di dati moderne rende più che abbordabile l'utilizzo di tecnologia di cifratura. Inoltre l'utilizzo di uno standard già implementato e testato da terze parti riduce il rischio di sicurezza dovuto a difetti all'interno del nostro prodotto software.
Il controllo degli accessi rappresenta una opportuna soluzione per la garantire il sistema a fronte di malintenzionati. L'uso di un controllo delle autorizzazioni favorisce a separare i ruoli e a migliorare la sicurezza, probabilmente perdendo in facilità d'utilizzo.
La Disponibilità del servizio e la Replicabilità del sistema servono a garantire i requisiti R25NF R26NF, per contenere un attacco DoS: questi vincoli potrebbero essere in conflitto con la strutturazione di un'architettura client/server, si cerca quindi un trade-off tra disponibilità e architettura.

4.2 Scelta dell'architettura

Abbiamo optato per una architettura client/server a tre livelli: un database, un software middleware e un applicativo front-end. L'applicazione middleware fa da intermediario tra il database e l'applicativo lato cliente. Questa scelta rende facile l'implementazione di applicativi front-end multipiattaforma e specializzati: si potranno avere quindi più applicativi front-end adatti al lavoro dello specifico utilizzatore finale.

4.2.1 Livello di Persistenza

Abbiamo deciso di adottare un database relazionale indipendente per la persistenza dei dati. Inoltre abbiamo capito che il software di gestione ha dei momenti di picco: per esempio durante l'entrata ad un evento si generano molte richieste. Vista la natura del software gestionale, l'utilizzo di modelli di persistenza integrata dell'applicazione middleware potrebbe scaturirsi in basse prestazioni. Inoltre, l'utilizzo di un database indipendente permette anche una scalabilità orizzontale del parco middleware, favorendo la disponiblità del servizio (R25NF), ma limitando la replicabilità del sistema in locale (R26NF) nel caso in cui il database non sia accessible all'esterno. Il problema si può affrontare creando uno "scatto" del database, prima dell'inizio dell'evento, risolvendo localmente così il problema riscontrato nell'analisi del rischio.
La persistenza dei log viene gestita attraverso file.

4.2.2 Livello Middleware

Le funzionalità di questo livello sono pensate per facilitare lo sviluppo multipiattaforma centralizzato, sfruttando una comunicazione con la controparte front-end. Si tratta dell'unico livello che può interagire con il livello di persistenza, e nel caso ci siano vincoli non direttamente implementabili nel progetto del database, saranno implentati qui, senza il rischio che una terza applicazione che utilizza il database, non implementi tali vincoli, rendendo intrinsecamente più semplice la struttura del front-end.

4.2.3 Livello Front-end

Si tratta dell'ultimo livello, quello a contatto con gli utilizzatori finali, specializzato nel rappresentare le informazioni reperite dallo strato server intermedio. Il livello è pensato per essere altamente specializzato e multipiattaforma: un'implementazione può sfruttare la piattaforma sottostante e reperire le informazioni in maniera più rapida. Inoltre si potrebbero introdurre diverse implementazioni, a seconda dell'utilizzatore finale.

4.2.4 Patterns & Design Principle

Abbiamo adottato il pattern **Broker**, che rende facile la scalabilità orizzontale dello strato intermedio, facilitando anche il compito di replicazione locale in caso di necessità. Tale scelta, inoltre, permette di applicare **The Dependency Inversion Principle** disaccoppiando i client dallo specifico middleware, dato che viene definito a priori il protocollo di comunicazione.
Abbiamo deciso di adottare il pattern architetturale **MVP** per l'applicativo middleware, mentre per la parte front-end si è utilizzato il **MVVM**. Il software middleware avrà una parte di view minima, **passiva**, volta solo alla comunicazione lato front-end, mentre il software front-end sarà decisamente più orientato alla presentazione e avente meno business logic.

4.2.5 Diagramma a componenti

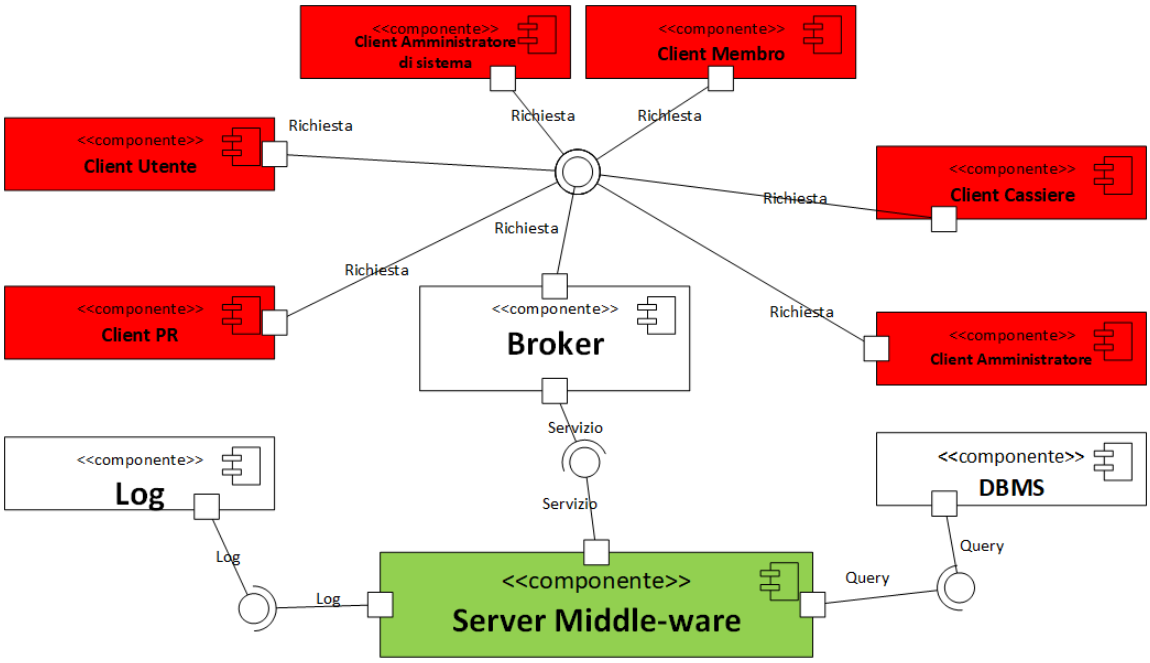

Figura 18: diagramma_architettura_logica_componenti.vsdx

4.2.6 Scelte tecnologiche

Abbiamo deciso di utilizzare come protocollo di sicurezza e trasmissione dei dati il protocollo **HTTPS**: la sicurezza è garantita dal layer TLS, che assicura una comunicazione sicura end-to-end, inoltre l'ampio bacino di utilizzatori del protocollo permette la produzione di un prodotto software versatile: un possibile servizio esterno potrebbe interfacciarsi con l'applicativo middleware, senza riscontrare troppe difficoltà.
Per il formato dei dati abbiamo scelto **JSON**, dato il grande supporto e la sua facilità di utilizzo.

4.3 Progettazione di dettaglio

4.3.1 Struttura

I cambiamenti del dominio seguente rispetto a quello dell'analisi del problema sono dovuti alla decisione di **dividere il software** in strato middleware e strato front-end. Inoltre, i cambiamenti sono imposti dalla scelta di utilizzare i Design Pattern **MVP** e **MVVM**.

4.3.2 Struttura middleware

Come detto in precedenza, la struttura è basata su **MVP**.

- **Model**: Si tratta della parte che gestisce il database.
  Utilizzando il pattern **DAO**, si crea il mapping delle relazioni con gli oggetti.
  Non viene utilizzato il pattern **Proxy** per implementare il **lazy-load**: dato che le richieste al middleware sono definite a priori, si conosce esattamente quali dati reperire, potendo usufruire del **JOIN** fornito dal DBMS.
  Per l'interazione con i log si è preferito un accesso diretto a file, sempre con un accesso tramite pattern **DAO**.
- **View**: Gestisce la risposta al client, formattando i dati nel formato JSON, si tratta di una vista **passiva**.
- **Presenter**: Collante tra Model e View, con tutta la business logic di supporto.

Package it.unibo.prevenditaelettronica.middleware.model.dao

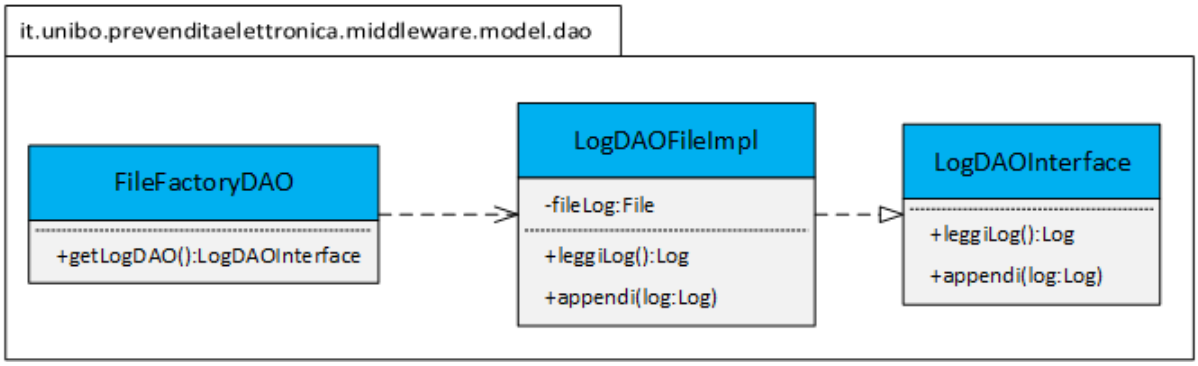

Figura 19: dao.vsdx

Pattern **DAO** applicato alla persistenza dei log. Dato che il log è stato pensato come file, il pattern DAO viene applicato al filesystem, anche se sarebbe possibile creare un implementazione per un data source differente in una futura versione, grazie alla versatilità del pattern DAO.

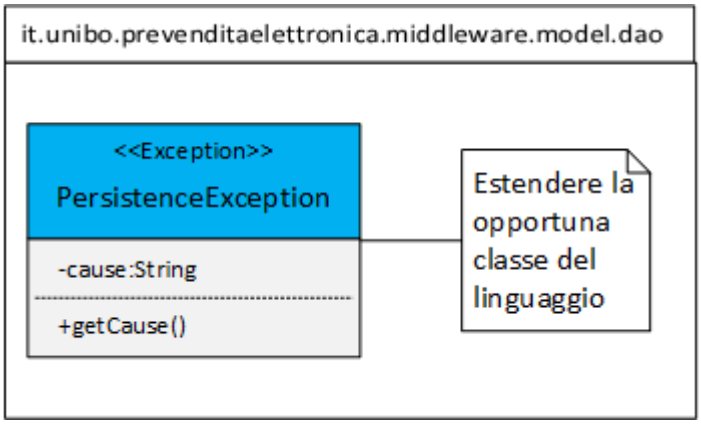

Figura 20: dao.vsdx

Naturalmente gli oggetti **DAO** possono lanciare eccezioni, qui si incapsulano tali eccezioni, per migliorare la sicurezza.

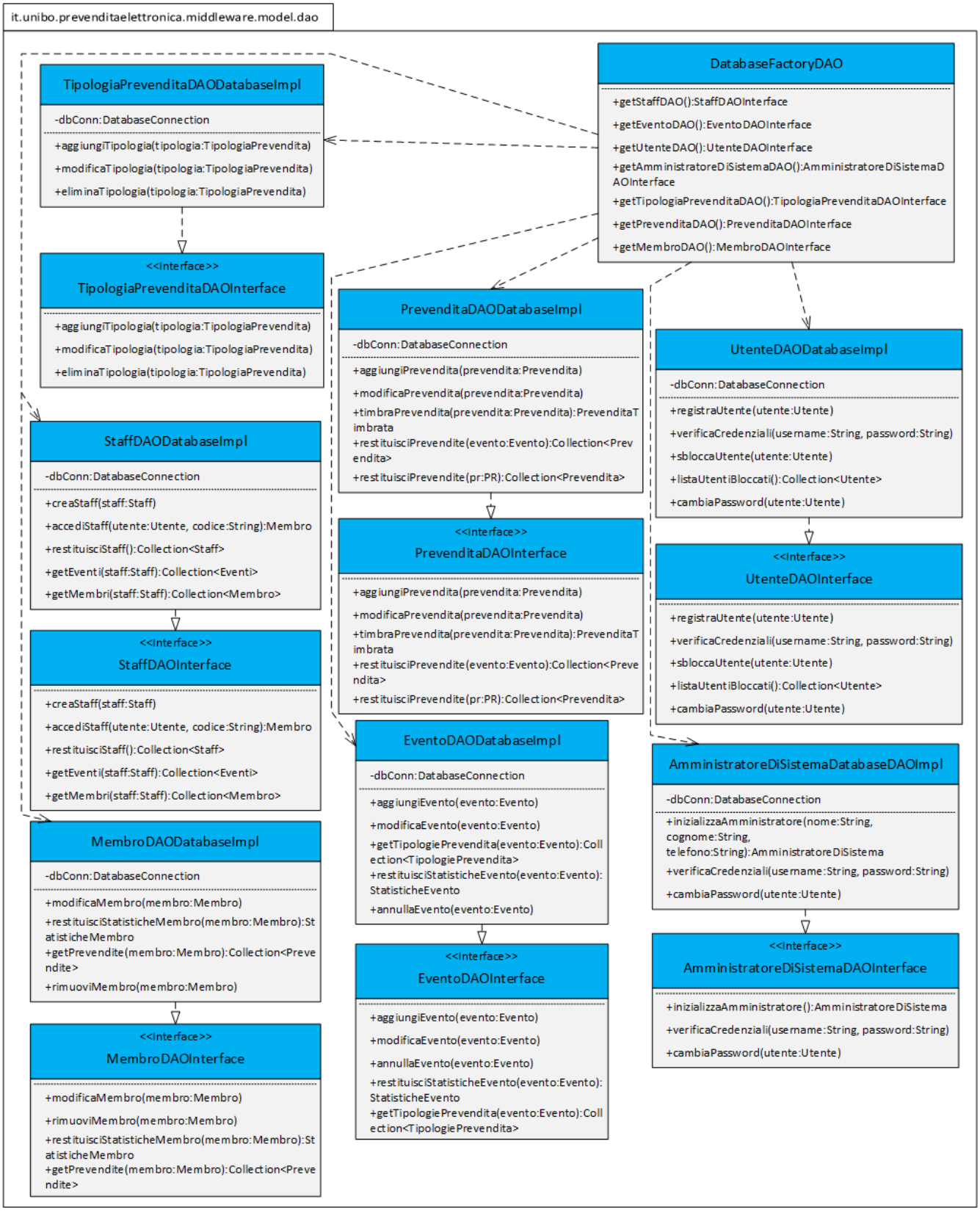

Figura 21: dao.vsdx

Pattern **DAO** applicato alla persistenza del modello di dominio, eccetto il log, salvato su file. L'accesso al database dipende dall'implementazione che si sceglie, infatti **DatabaseConnection** è un oggetto dipendente dalla piattaforma utilizzata. L'operazione **inizializzaAmministratore** inizializza l'amministratore di sistema ai valori di default. L'operazione può essere fatta ovviamente una volta e l'amministratore dovrà poi cambiare la password di default. Le **transazioni** sono gestite all'interno delle implementazioni, a seconda dei casi d'uso. Se necessario aggiungere nell'implementazione i metodi CRUD.

Package it.unibo.prevenditaelettronica.middleware.model

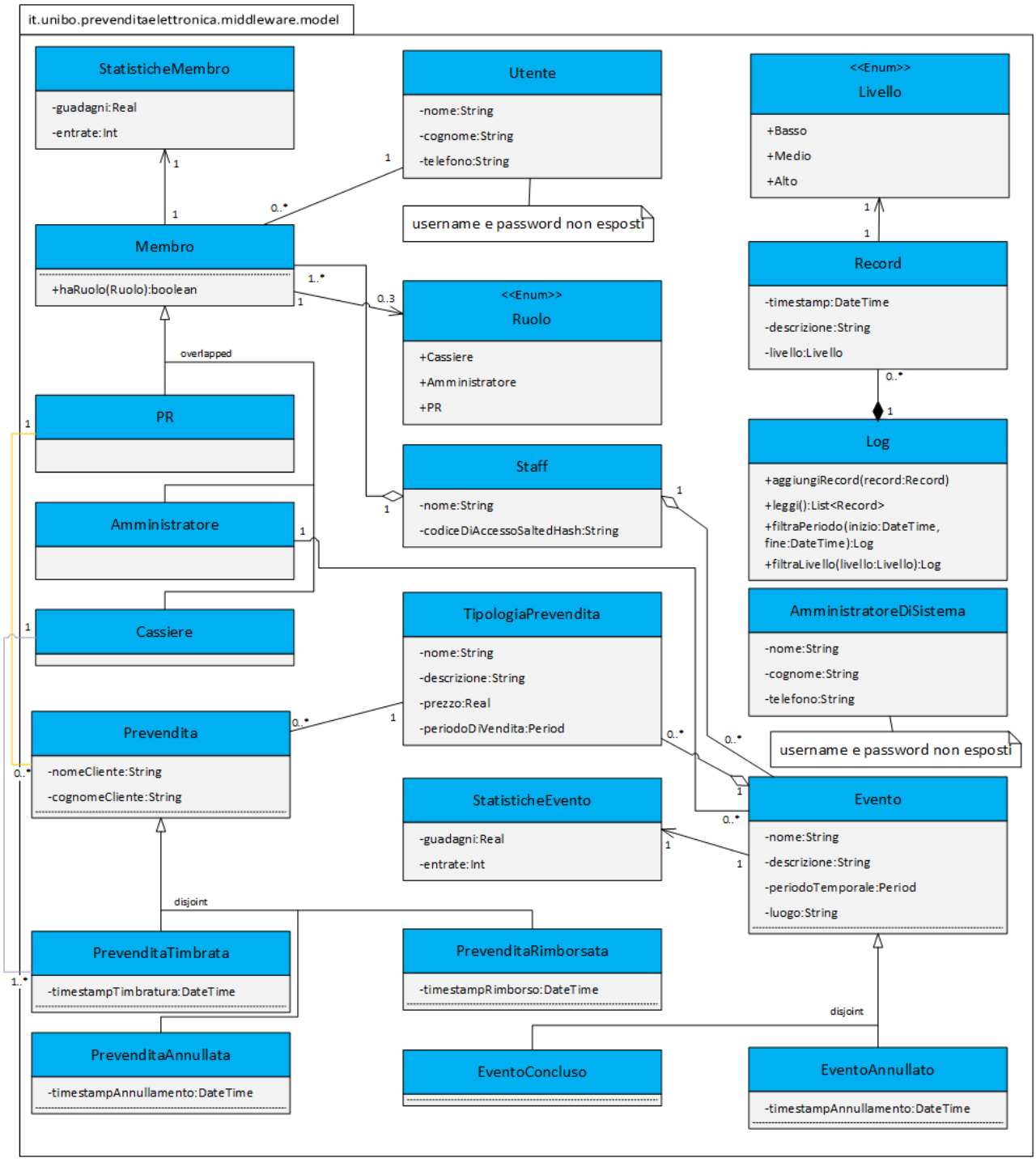

Figura 22: model.vsdx

Si tratta del modello di dominio visto in analisi. La maggior parte delle operazioni previste in analisi sono spostate in altri package. La gestione delle credenziali viene fatta nel package DAO e quindi il modello di dominio non contiene informazioni sulle credenziali. La generazione del documento viene fatta dal componente front-end. Rappresentano anche le classi di **DTO** per referenziare gli oggetti persistenti tramite chiavi surrogate (le chiavi da aggiungere sono viste nella progettazione della persistenza). Gli oggetti del modello di dominio vengono usati anche per trasmettere i dati ai client.

Package it.unibo.prevenditaelettronica.middleware.view

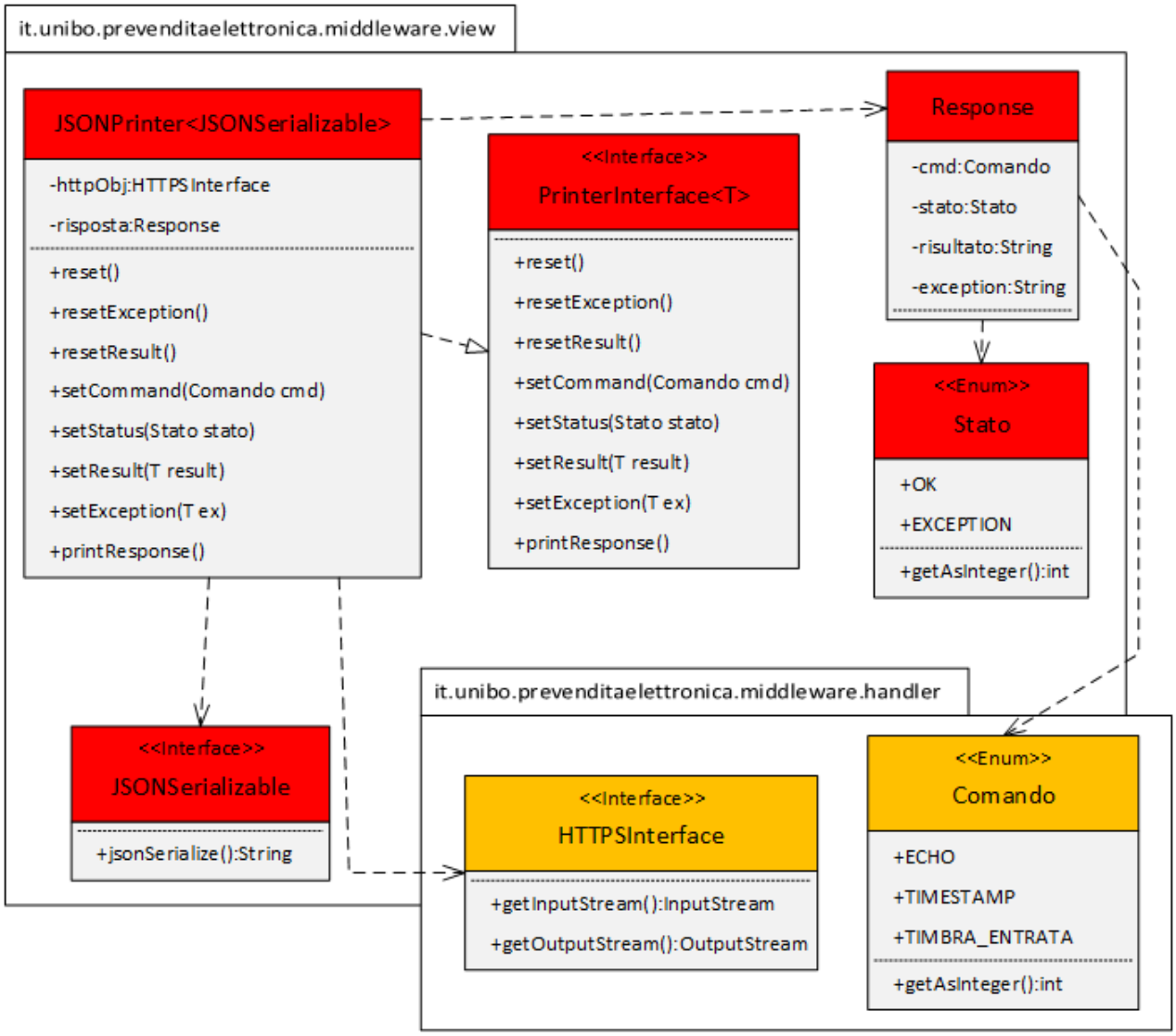

Figura 23: view.vsdx

Il package it.unibo.prevenditaelettronica.middleware.view si occupa della rappresentazione dei dati verso il front-end. Data la scelta tecnologica di usare JSON come formato dati, abbiamo implementato una sola classe Printer, ma l'interfaccia rende flessibile l'utilizzo di un altro formato dati.
L'interfaccia **JSONSerializable** non è obbligatoria, dipende dalle librerie usate in fase di implementazione, per esempio GSON non ne prevede. Nel caso in cui la serializzazione venga fatta senza aiuti esterni, l'interfaccia va implementata nelle classi di dominio, rispettando i vincoli di sicurezza.
L'interfaccia **HTTPSInterface** serve per ricevere la richiesta e inviare la risposta. Può essere sostituita in implementazione dalla relativa classe di supporto della piattaforma scelta.
L'enumeratore **Stato** indica lo stato della risposta, mentre l'enumeratore **Comando** contiene tutti i comandi eseguibili sul middleware. La lista non è completa e viene riportata nel package **handler** in seguito.
La classe **Response** rappresenta le informazioni scambiate con il client in fase di risposta: viene trasferito il **comando** per cui si è fatta la richiesta, lo stato della risposta, l'eventuale risultato o eccezione. Il risultato o l'eccezione è già serializzato, mentre l'oggetto Response verrà serializzato successivamente.

Package it.unibo.prevenditaelettronica.middleware.controller

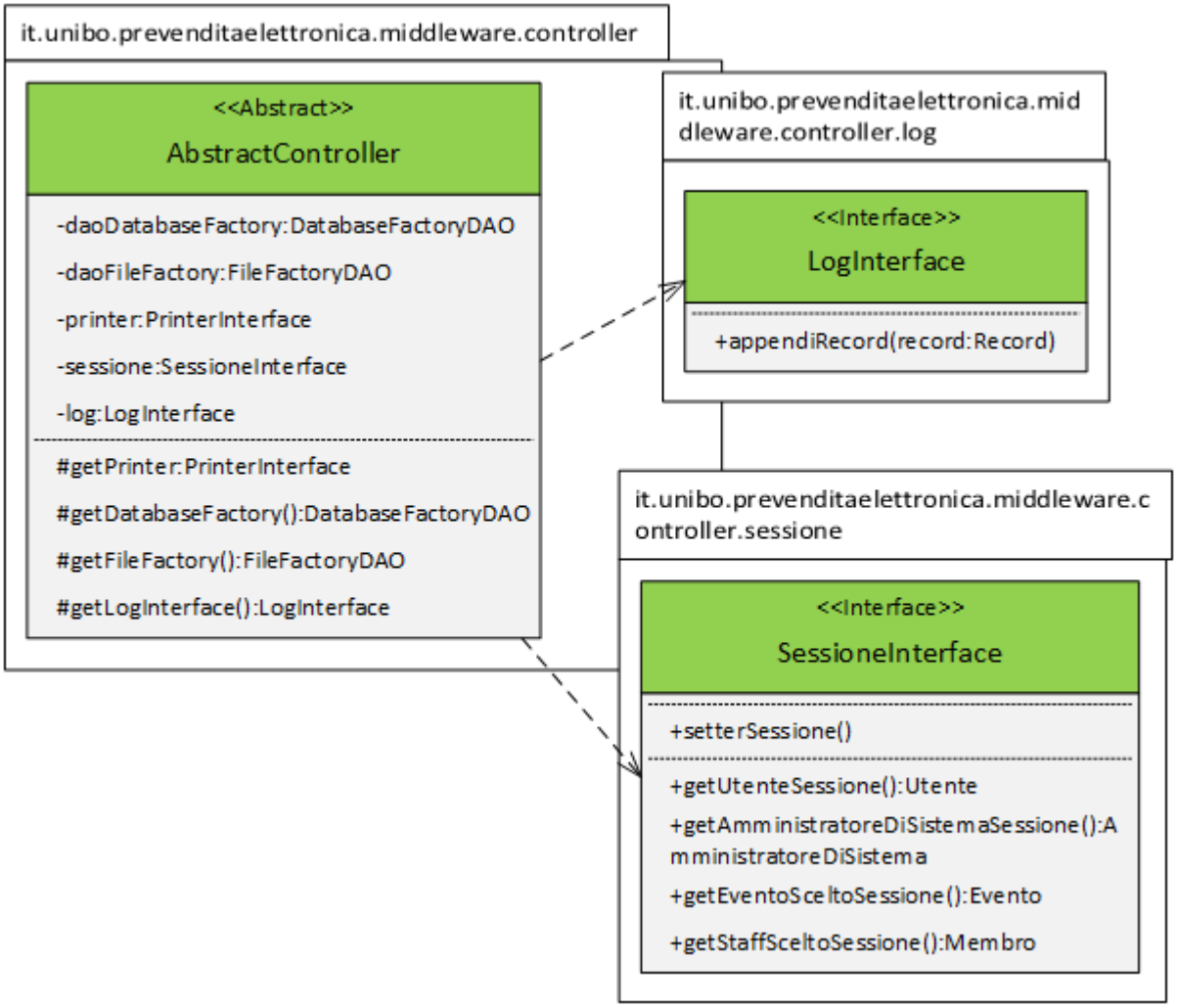

Figura 24: controller.vsdx

Si tratta di un controller astratto, usato dai controller concreti mediante lo stereotipo <<Controller>>. Utilizza l'interfaccia **LogInterface** per l'append del log nel caso l'operazione concreta lo preveda. Utilizza l'interfaccia **SessioneInterface** nel caso serva accedere alla sessione corrente.

Package it.unibo.prevenditaelettronica.middleware.controller.sessione

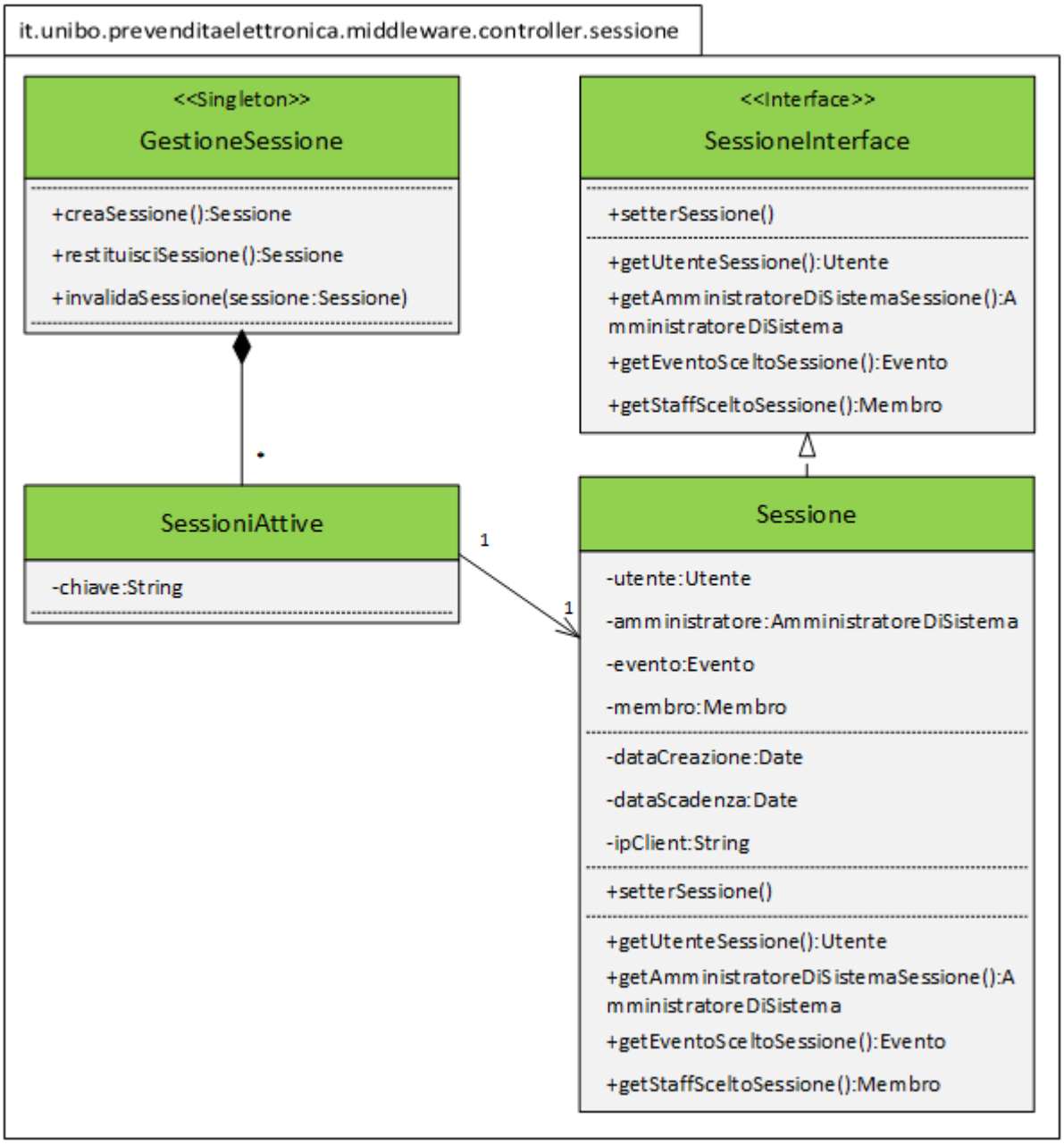

Figura 25: controller.vsdx

La sessione viene gestita tramite pattern **Singleton**, nella classe **GestioneSessione**. Viene mostrata agli altri controller tramite l'interfaccia **SessioneInterface**. Vengono salvate le informazioni relative al modello del dominio, come l'utente o amministratore di sistema che ha effettuato il login e l'eventuale staff o evento scelto. Staff restituisce Membro, dato che abbiamo bisogno di identificare i ruoli del membro nello staff.

Package it.unibo.prevenditaelettronica.middleware.controller.autenticazione

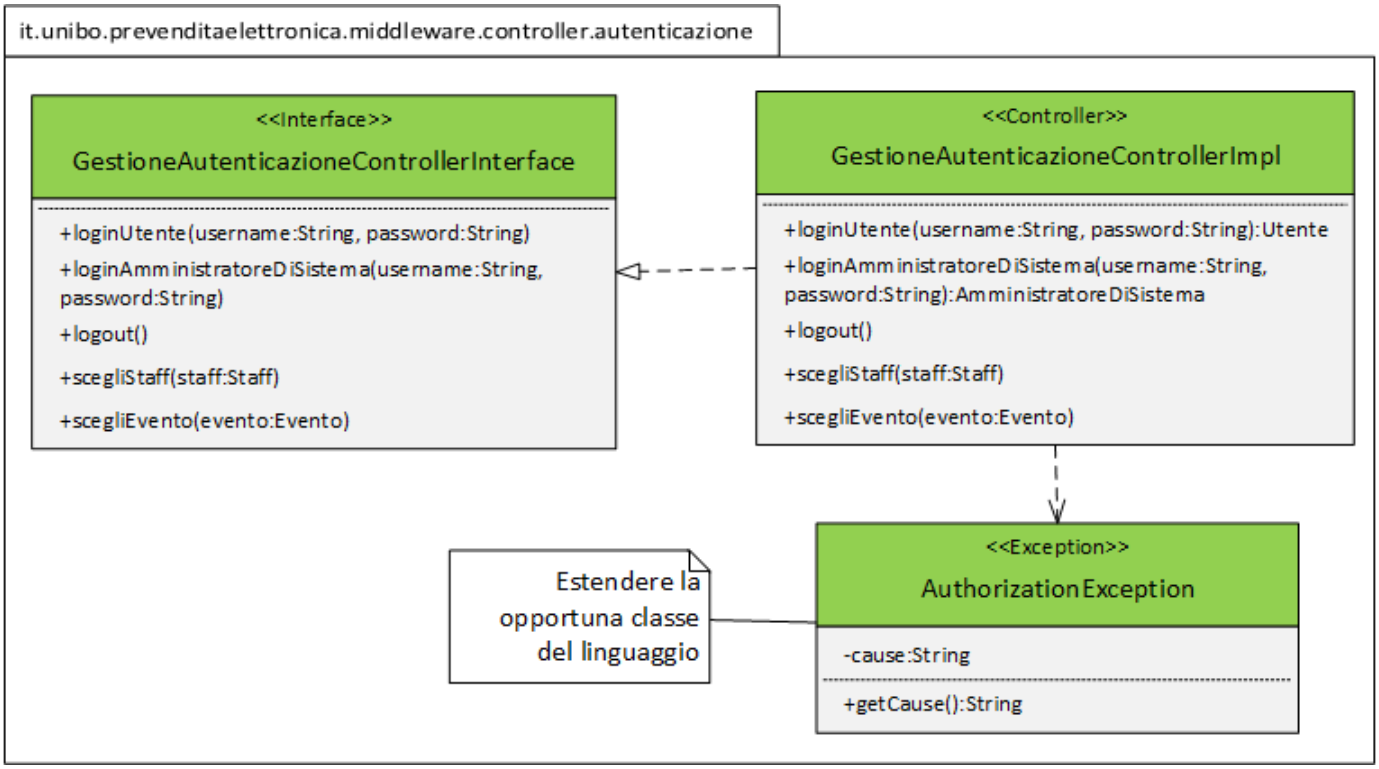

Figura 26: controller.vsdx

In figura la parte di controller con la responsabilità di gestire la parte di autenticazione.
L'eccezione **AuthorizationException** viene lanciata nei più svariati casi di accesso negato: credenziali errate, utente bloccato, ruolo non concesso, eccetera, e viene utilizzato anche da altri controller nel caso sia necessario.

Package it.unibo.prevenditaelettronica.middleware.controller.log

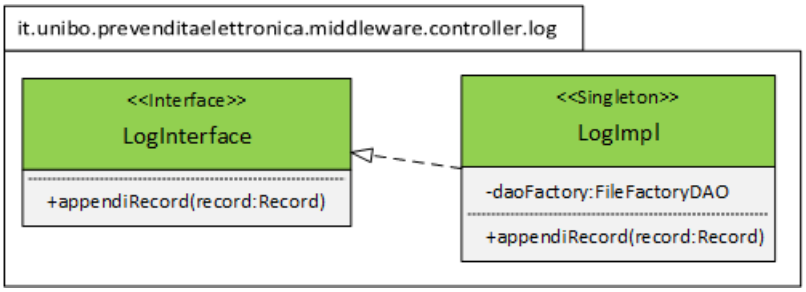

Figura 27: controller.vsdx

In figura viene mostrato anche la parte di log interna al sistema, utilizzata dagli altri controller per aggiungere righe al log. **LogImpl** ha una DAO specializzata per l'accesso al filesystem, dato che i file di log si trovano su di esso.

Package it.unibo.prevenditaelettronica.middleware.controller.amministrazione

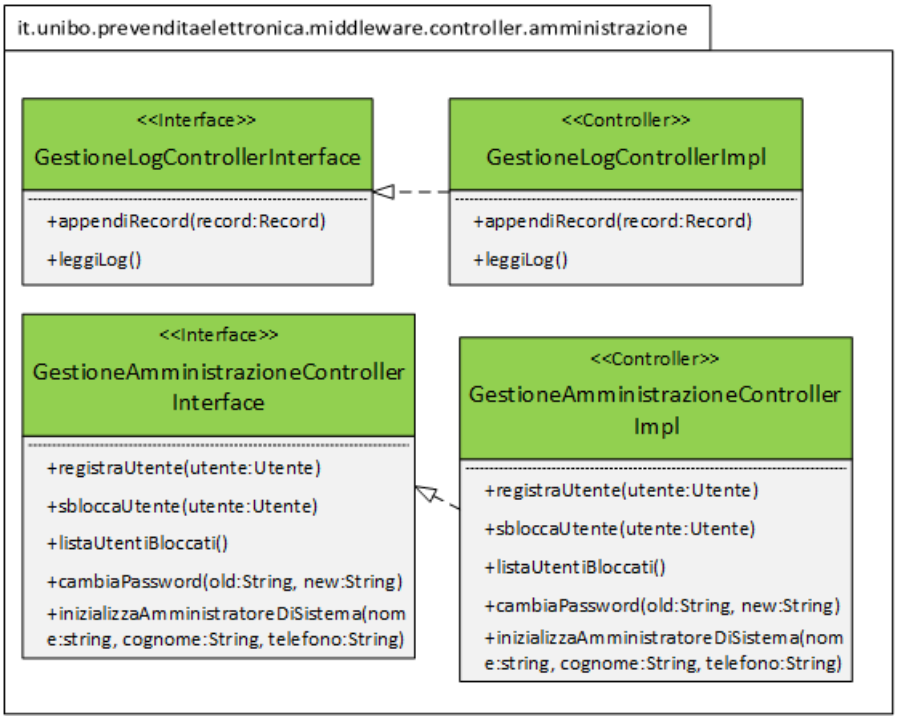

Figura 28: controller.vsdx

In figura la parte di controller con la responsabilità di gestire la parte di amministrazione e gestione log. **GestioneLogControllerImpl** utilizza la DAO per l'accesso ai file, dato che i log sono salvati sul file system. L'operazione **inizializzaAmministratoreDiSistema** serve durante il **deployment** del sistema: fornisce un account di amministrazione con paramentri a **default**. Non è necessario inizializzare altri dati all'interno del database, perché sarà compito dell'amministratore di sistema registrare nuovi utenti, i quali aggiungeranno tutto il necessario per il funzionamento del software gestionale.

Package it.unibo.prevenditaelettronica.middleware.controller.utente

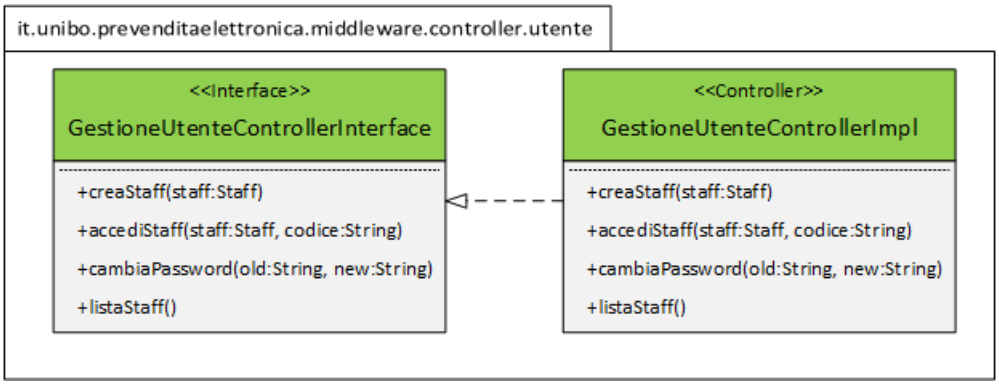

Figura 29: controller.vsdx

In figura la parte di controller con la responsabilità di gestire la parte di utente.

Package it.unibo.prevenditaelettronica.middleware.controller.staff
Abbiamo suddiviso il package in due diagrammi per migliorarne la lettura.

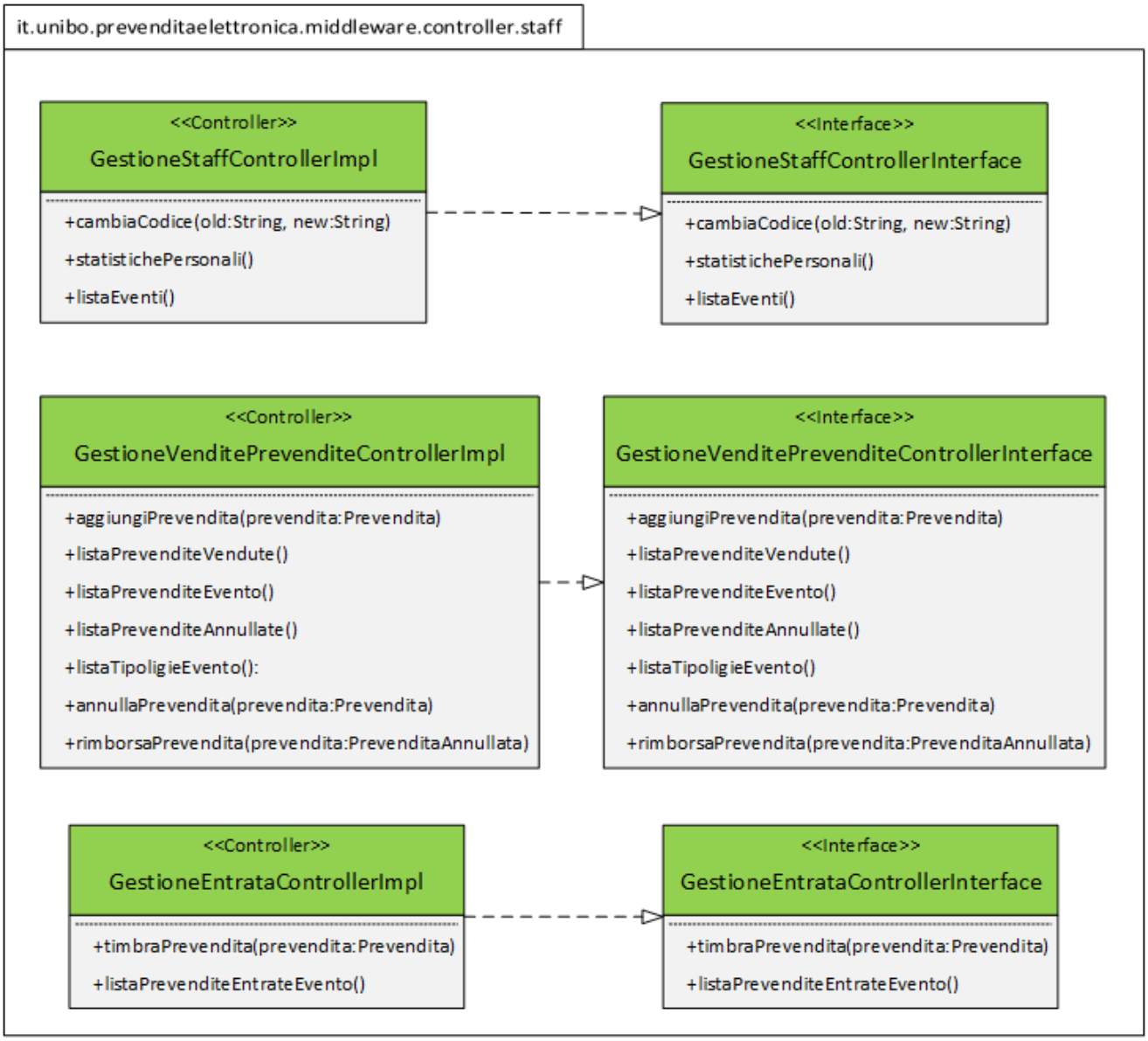

Figura 30: controller.vsdx

In figura la parte di controller con la responsabilità di gestire la parte di staff. Qui viene mostrata la parte contenente **GestioneStaffController**, **GestioneVenditePrevendite** e **Gestione Entrata**. Si tratta della parte utilizzata dagli attori **PR** e **Cassiere**.

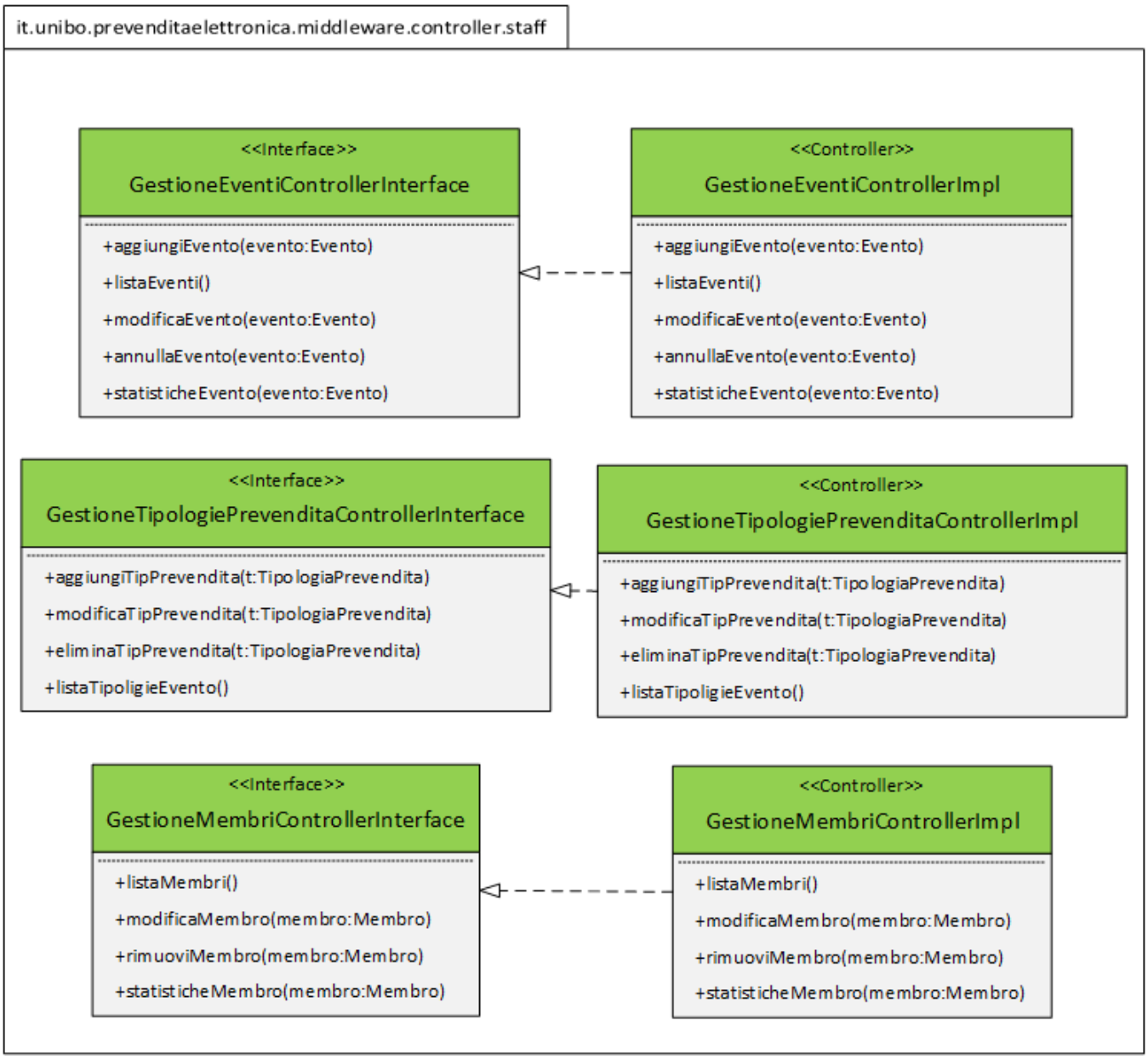

Figura 31: controller.vsdx

In figura la parte di controller con la responsabilità di gestire la parte di staff. Qui viene mostrata la parte contenente **GestioneEventi**, **GestioneMembri** e **Gestione Tipologia Prevendita**. Si tratta della parte utilizzata dall'attore **Amministratore**.

Package it.unibo.prevenditaelettronica.middleware.handler

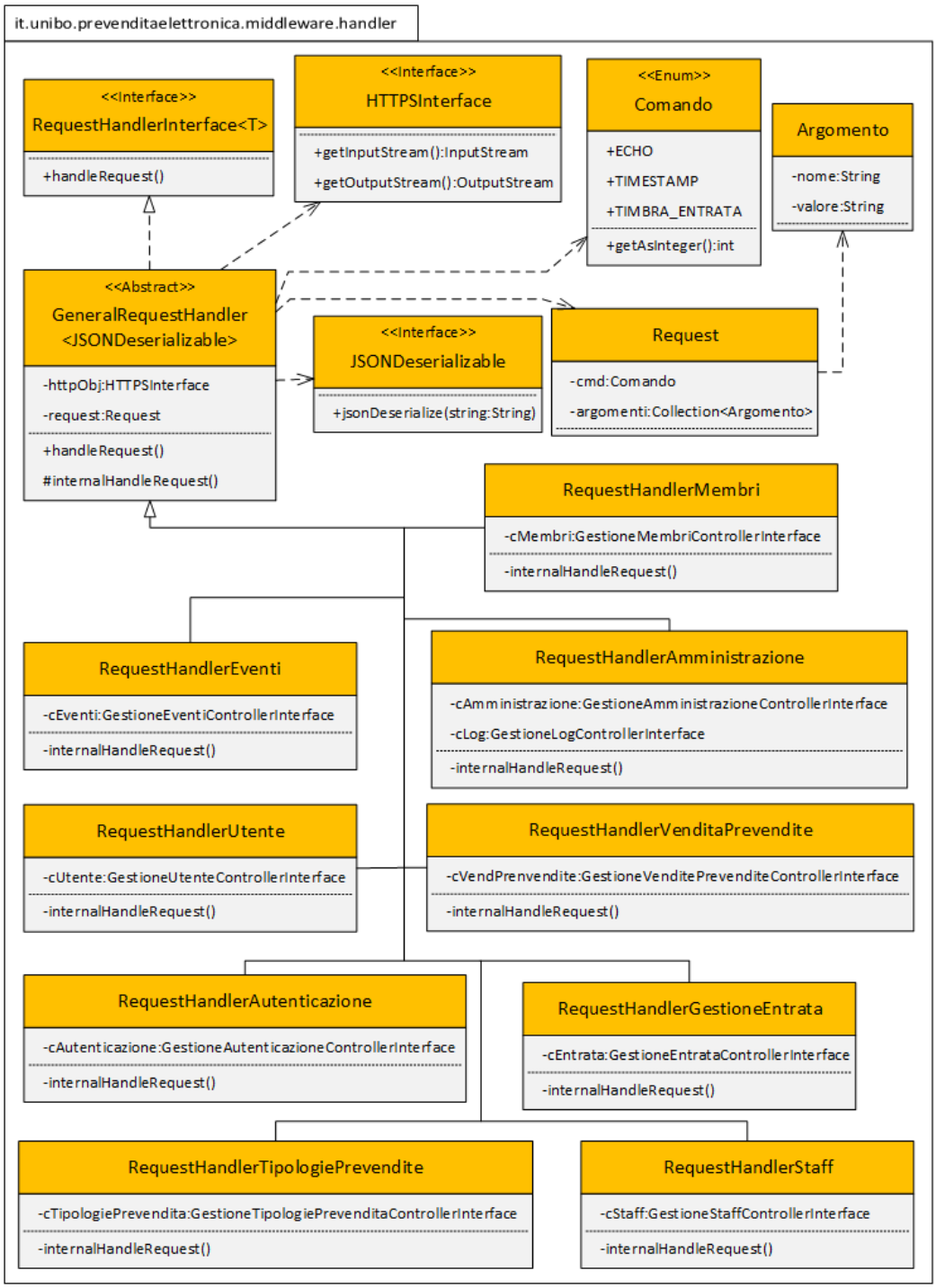

Figura 32: handler.vsdx

In figura viene mostrato il package **handler**, che si occupa di raccogliere le richieste HTTPS ed elaborarle, inoltrando la richiesta al controller opportuno. Si utilizza il pattern **State** per differenziare la presa in carico della richiesta, a seconda del comando utilizzato. L'interfaccia **HTTPSInterface** serve per ricevere la richiesta e inviare la risposta. Può essere sostituita in implementazione dalla relativa classe di supporto della piattaforma scelta. Il metodo **handleRequest** viene chiamato quando si deve gestire una richiesta, utilizzando un thread separato per ogni richiesta. La classe astratta **AbstractRequestHandler** potrebbe subire dei cambiamenti, a seconda della piattaforma sottostante. L'enumeratore **Comando** contiene tutti i comandi disponibili presso il livello middleware. L'interfaccia **JSONDeserializable** serve per convertire la richiesta JSON in oggetti utilizzabili. Può essere sostituita da un'eventuale libreria della piattaforma. La classe **Request** rappresenta le informazioni in ingresso dal client: contiene il **comando** da eseguire ed eventuali **argomenti** per la corretta esecuzione del comando, il tutto già serializzato, mentre l'oggetto Request verrà serializzato successivamente.

Di seguito una lista dei comandi:

- REGISTRA_UTENTE
- LOGIN
- LOGOUT
- INIZIALIZZA_SISTEMA
- LOGIN_SYSTEM_ADMIN
- CAMBIA_PASSWORD_ADMIN
- CREA_STAFF
- ACCEDI_STAFF
- CAMBIA_PASSWORD
- LISTA_STAFF
- LISTA_MEMBRI
- LISTA_EVENTI
- SCEGLI_EVENTO
- SCEGLI_STAFF
- LISTA_TIPOLOGIE_PREVENDITA
- AGGIUNGI_PREVENDITA
- ANNULLA_PREVENDITA
- RIMBORSA_PREVENDITA
- LISTA_PREVENDITE
- STATISTICHE_MEMBRO
- STATISTICHE_EVENTO
- LISTA_PREVENDITE_EVENTO
- TIMBRA_PREVENDITA
- CAMBIA_CODICE_ACCESSO
- STATISTICHE_PERSONALI
- AGGIUNGI_EVENTO
- ANNULLA_EVENTO
- AGGIUNGI_TIP_PREVENDITA
- MODIFICA_TIP_PREVENDITA
- ELIMINA_TIP_PREVENDITA
- MODIFICA_MEMBRO
- RIMUOVI_MEMBRO

4.3.3 Struttura front-end

Come detto in precedenza, la struttura è basata su **MVVM**.

- **Model**: Deve elaborare i dati per la ricezione/invio verso il middleware.
  L'accesso al middleware è dettato dal pattern **Broker**.
- **View**: Componente principale del client front-end, utilizziamo il pattern **Observer** per il binding con il ViewModel. Dovrebbe essere ridotta alla sola logica di presentazione.
- **ViewModel**: Collante tra Model e View, è il responsabile della logica della vista.

La modularità del **MVVM** permette di combinare facilmente le varie viste, per ottenere client veramente **personalizzati**.

Package it.unibo.prevenditaelettronica.frontend.model.net.broker

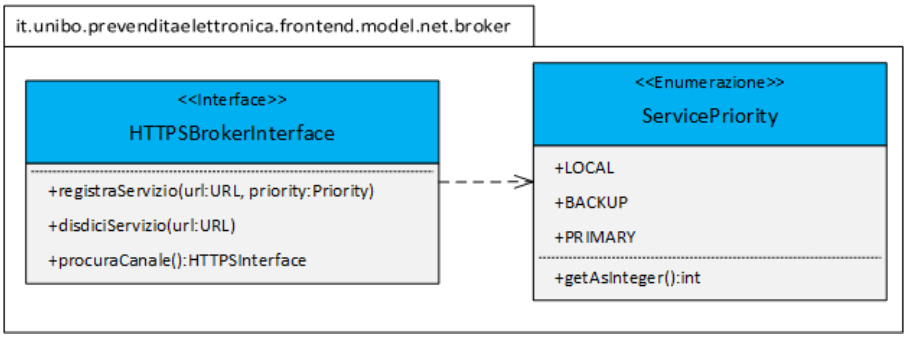

Figura 33: broker.vsdx

Il componente **HTTPSBrokerInterface** rappresenta un'interfaccia in grado di registrare servizi middleware, secondo una certa **priorità**. Utilizzando il metodo **procuraCanale**, il broker procura un canale per il client, in particolare andando a cercare il servizio remoto, seguendo l'ordine di priorità. L'enumeratore **ServicePriority** serve per classificare i servizi in base ad una priorità: **Primary** rappresenta il servizio principale, quello chiamato la maggior parte delle volte, **Backup** rappresenta un possibile livello middleware secondario, sempre che si riesca a scalare orizzontalmente, mentre la priorità **Local** rappresenta il backup locale effettuato dall'amministratore di sistema su richiesta, per limitare gli attacchi DoS.

Package it.unibo.prevenditaelettronica.frontend.model

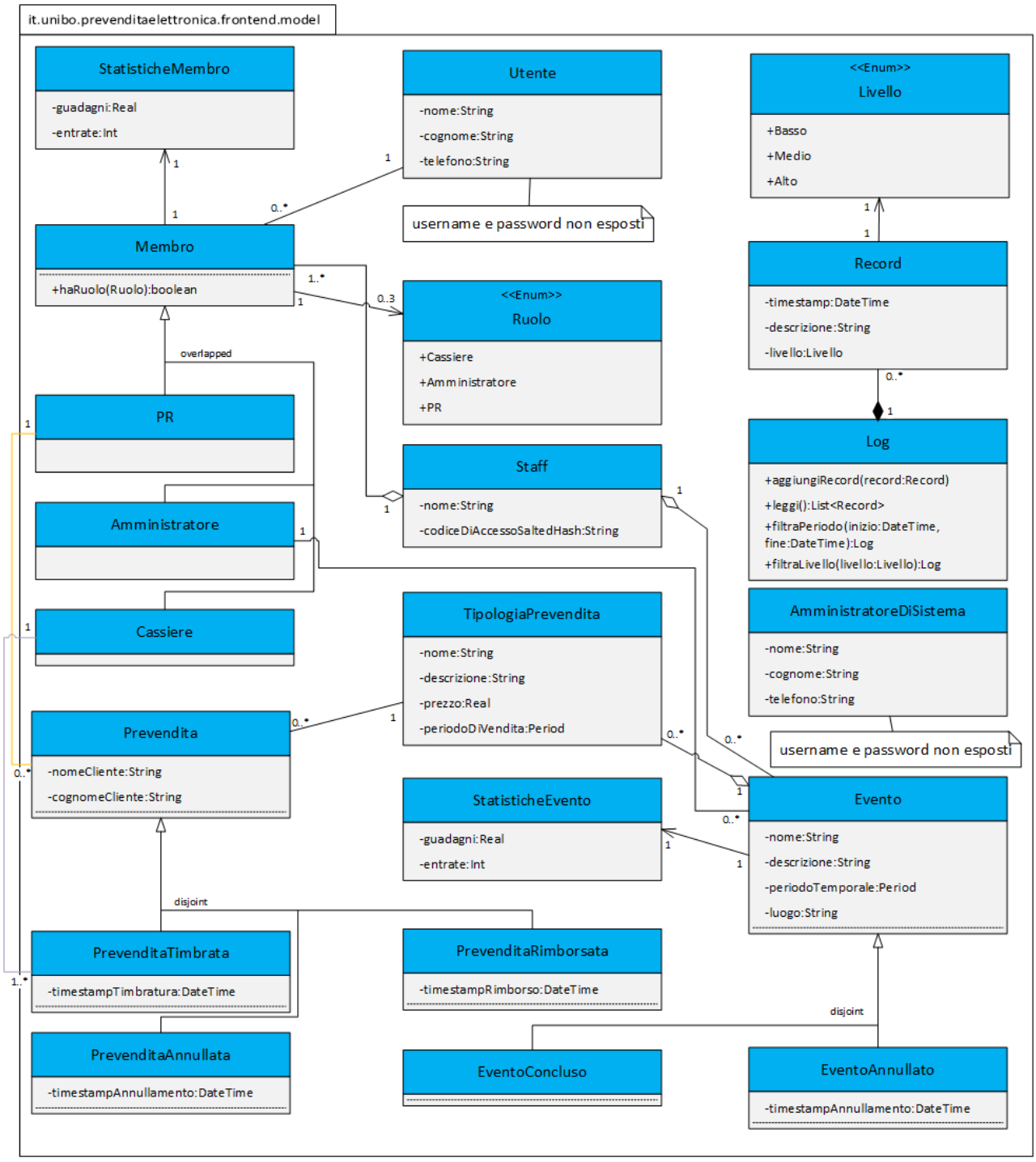

Figura 34: model.vsdx

Il modello di dominio è uguale a quello del midde-ware, ma cambia il metodo di reperire le informazioni: nel middleware le informazioni vengono ottenute attraverso interrogazioni del database, mentre nel front-end, le informazioni vengono ottenute tramite interrogazioni del middleware. Serve quindi progettare il meccanismo di accesso al middleware. Ovviamente è possibile che le informazioni ottenute dal servizio **non siano complete**: ci si limita alle sole informazioni richieste dal comando inviato. Inoltre i vari client implementeranno solo la parte di modello **necessaria**.

Package it.unibo.prevenditaelettronica.frontend.model.net

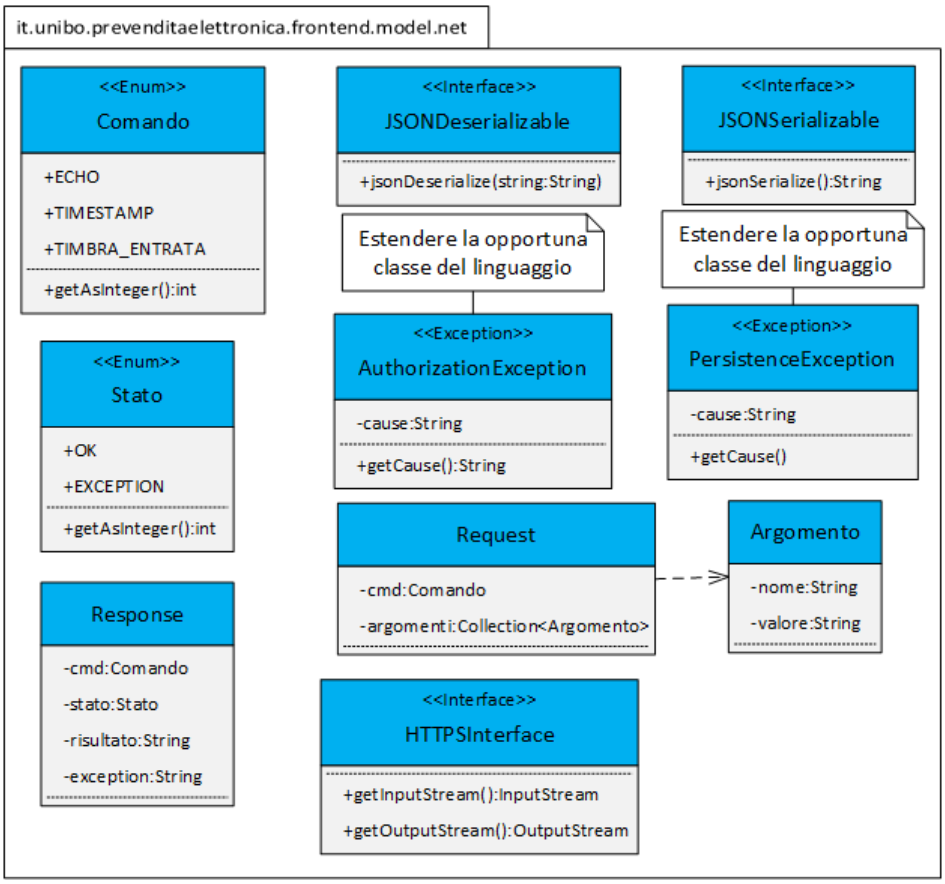

Figura 35: net.vsdx

Molte entità sono in comune con il layer middleware. Le interfacce **JSONSerializable** e **JSONDeserializable** servono per convertire la richieste e le risposte nel formato JSON. Può essere sostituita da una eventuale libreria della piattaforma. Le interfacce **Request** e **Response** contengono le informazioni per la comunicazione. All'interno le informazioni sono già serializzate, mentre gli oggetti stessi verranno serializzati successivamente.

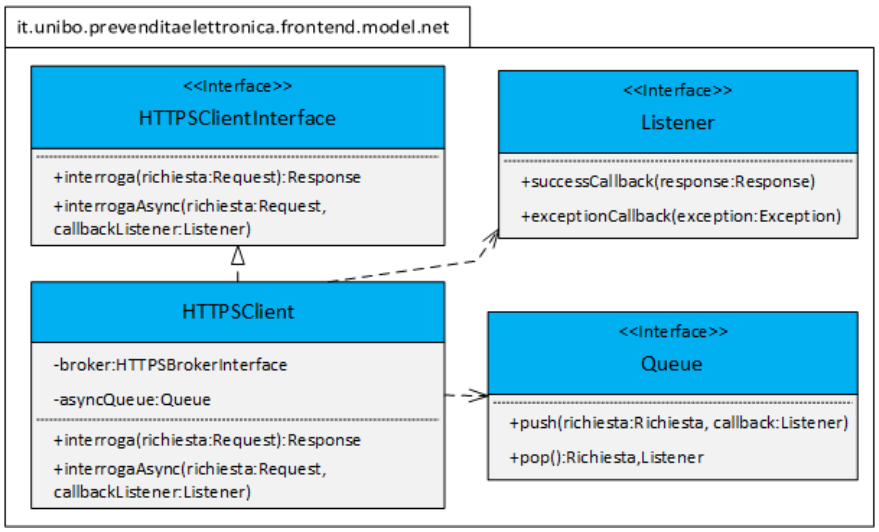

Figura 36: net.vsdx

In figura si rappresenta la modalità di comunicazione con il middleware: si è introdotto **HTTPSClientInterface** che rappresenta il client che effettua le richieste al middleware. Si ha la possibilità di effettuare richieste in modo sincrono o asincrono, a seconda delle esigenze. Il canale di comunicazione **HTTPSInterface** viene fornito da **HTTPSBrokerInterface**, il quale si pone tra client e middleware.

Package it.unibo.prevenditaelettronica.frontend.viewmodel

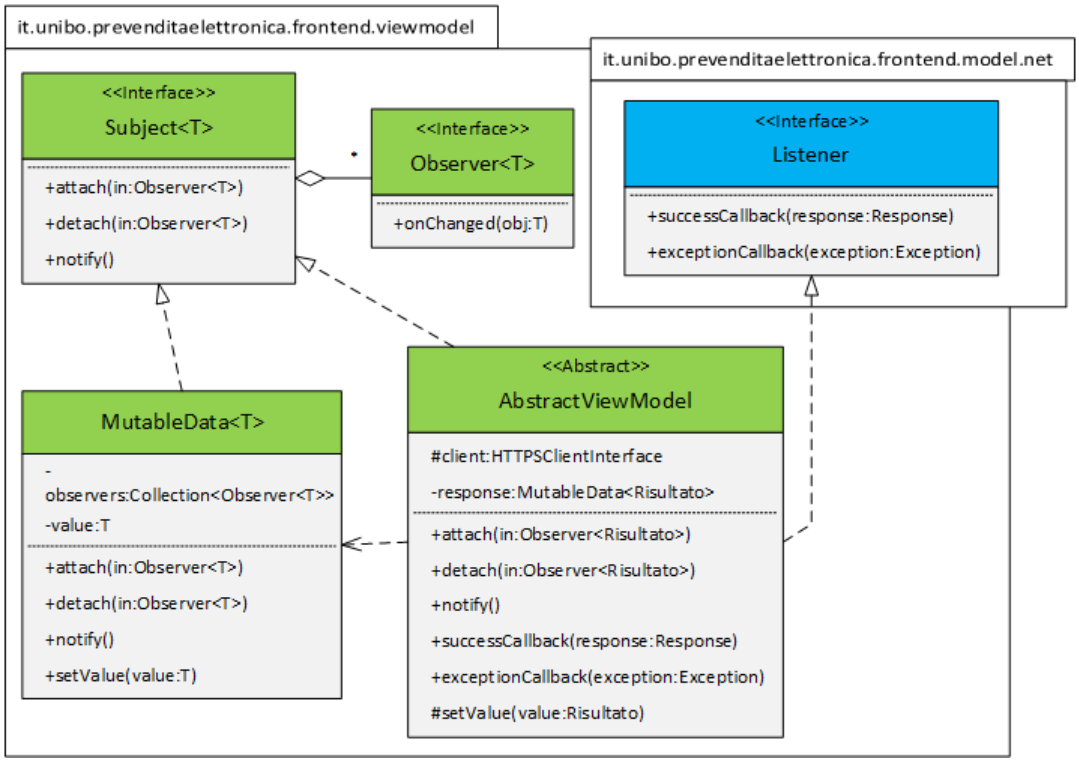

Figura 37: viewmodel.vsdx

Come detto in precedenza, si utilizza il pattern **Observer** per il binding tra **View** e **ViewModel**. Si è aggiunta una classe **MutableData** per la gestione di un oggetto con osservatori esterni. La classe astratta **AbstractViewModel** rappresenta un ViewModel generale, dentro alla quale si trovano il client **HTTPSClientInterface** per effettuare le richieste e un **MutableData** per la risposta, la quale sarà osservata dalla view e si aggiornerà di conseguenza. **AbstractViewModel** implementa anche l'interfaccia **Listener**, nel caso sia neccessario effettuare delle richieste asincrone. I ViewModel **concreti** hanno lo stereotipo <<**ViewModel**>>, indicando che estendono **AbstractViewModel**.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.autenticazione

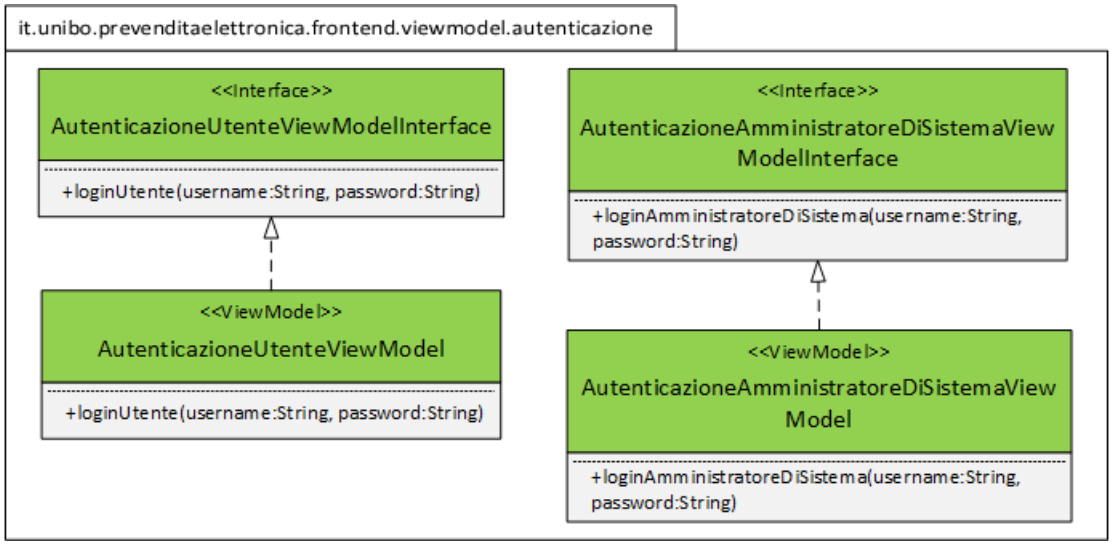

Figura 38: viewmodel.vsdx

Vengono rappresentate in figura i ViewModel per le viste di autenticazione. Sono quindi previste le viste di login collegate a questi ViewModel. Quando la vista riceve l'evento di submit, essa invoca il suo ViewModel, il quale dopo una verifica dell'input, invia la richiesta al middleware. Si potrebbe migliorare il meccanismo di validazione dell'input in futuro, rendendolo più dinamico, utilizzando sempre il pattern **Observer** per effettuare un check di valutazione dinamico. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.utente

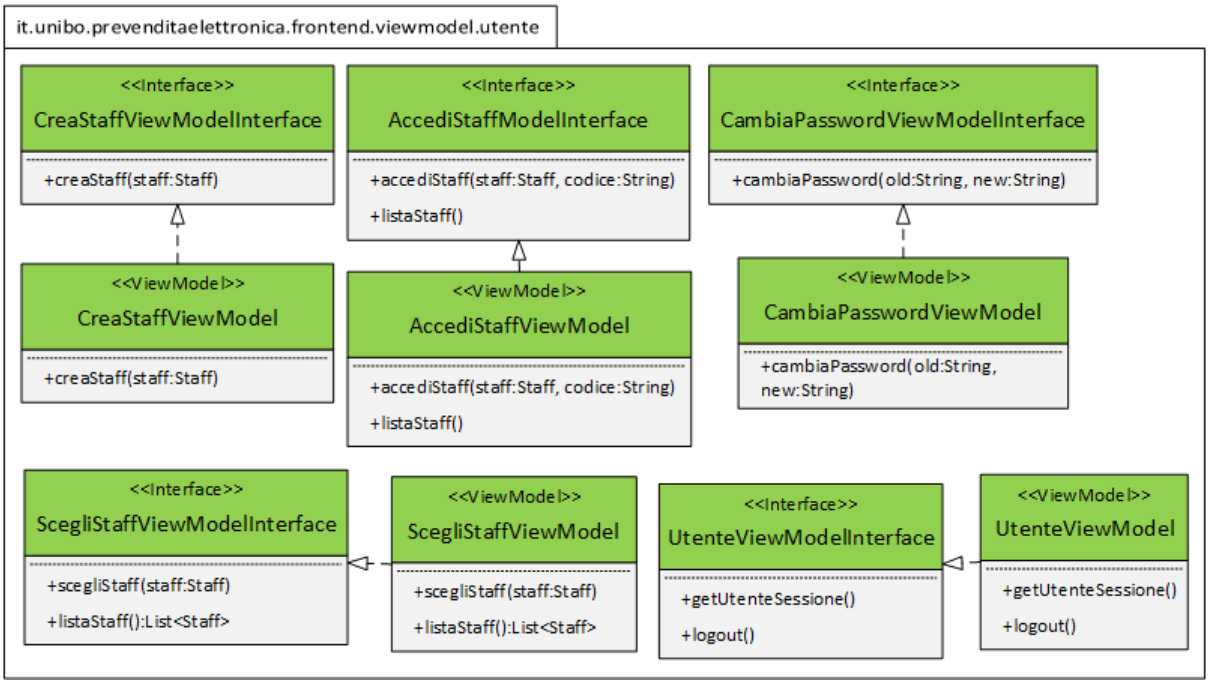

Figura 39: viewmodel.vsdx

Nella figura viene presentato la parte di ViewModel dell'**utente**. Oltre ai metodi principali viene inserito anche **ScegliStaff**, per scegliere lo staff durante la sessione. Esiste un ViewModel associato ad una vista per visionare l'utente loggato ed effettuare eventualmente il logout. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.amministrazione

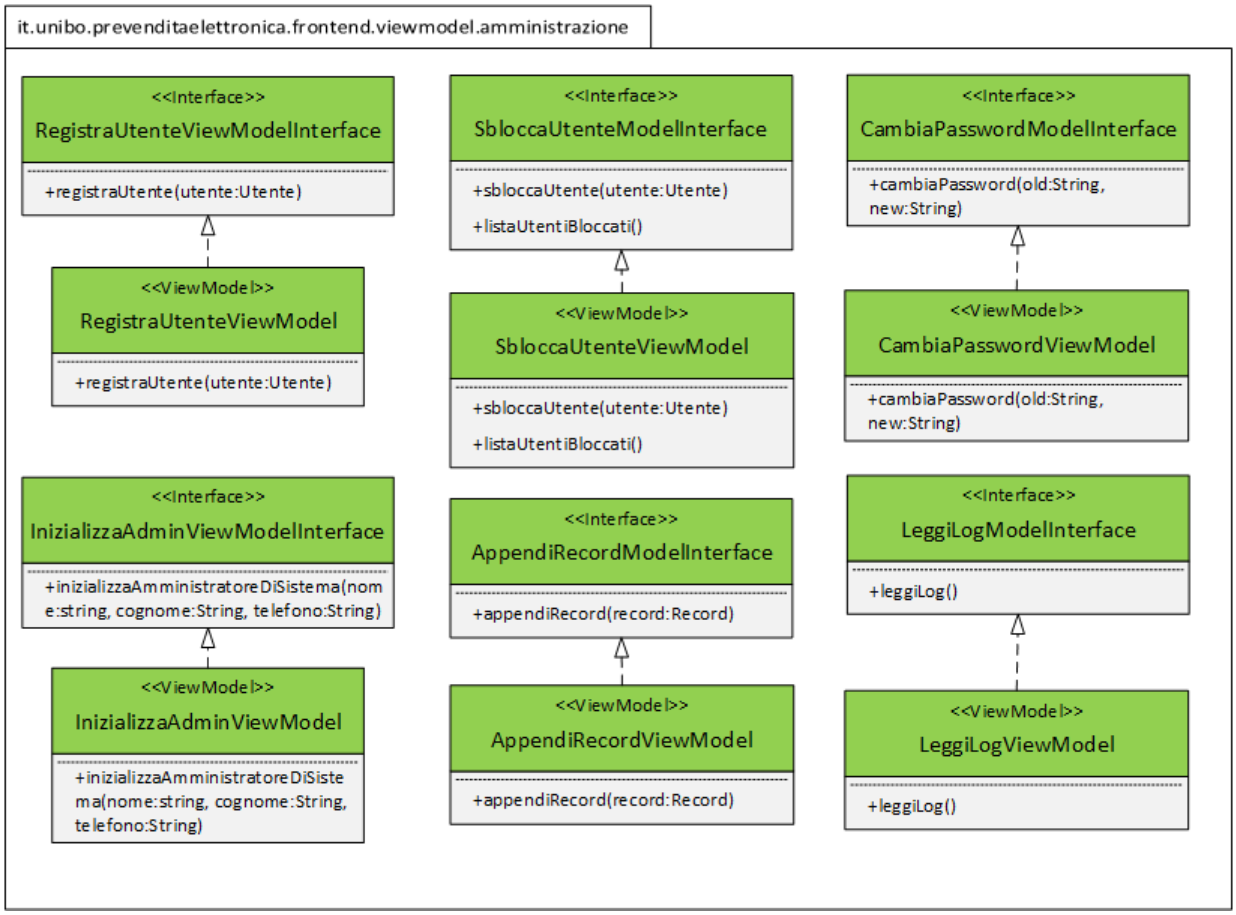

Figura 40: viewmodel.vsdx

Sono presenti tutti i ViewModel per gestire l'**amministrazione**: esiste la possibilità di registrare un utente, sbloccare un utente, cambiare la password dell'amministratore e gestire i log. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.membro

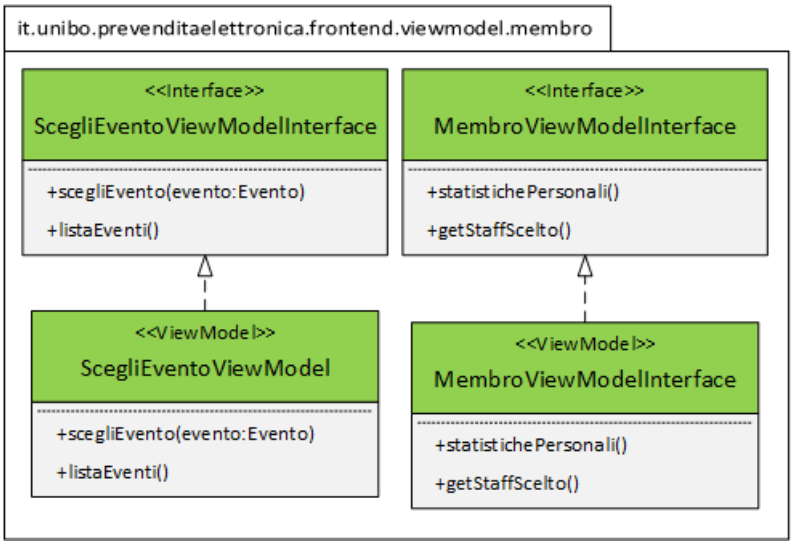

Figura 41: viewmodel.vsdx

Contiene semplici operazioni per la scelta dell'evento e per le statistiche del membro. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.pr

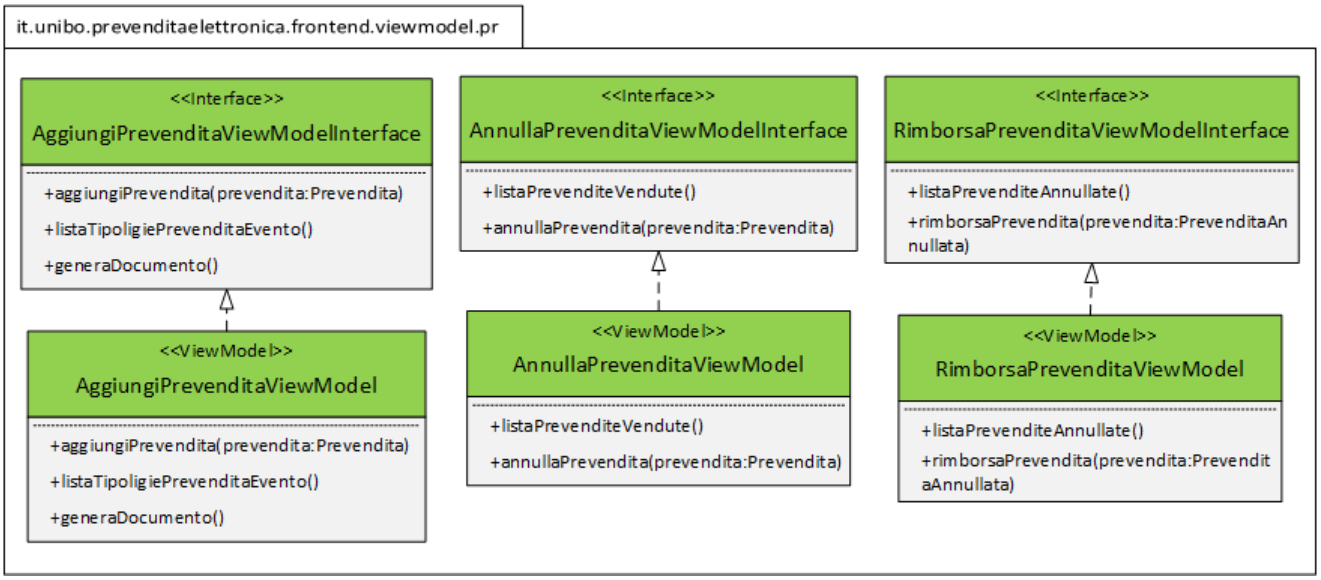

Figura 42: viewmodel.vsdx

Il PR ha alcune viste per la **vendita di prevendite**. La generazione del documento viene fatta tramite il metodo **generaDocumento** che permette alla vista di mostrare il documento generato, da consegnare al cliente. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.cassiere

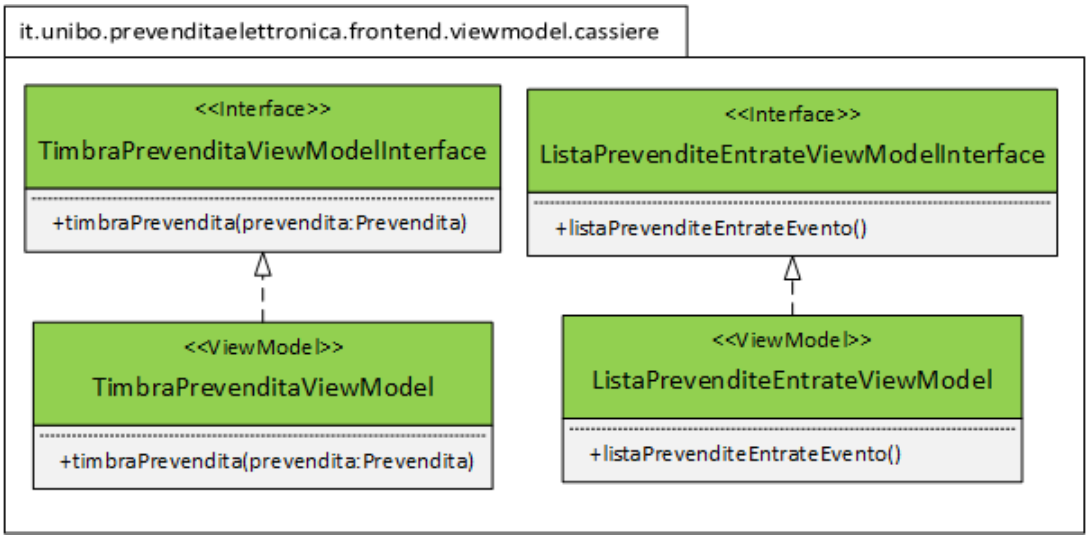

Figura 43: viewmodel.vsdx

Per il cassiere serve solo il ViewModel per la **timbratura** e per la lista di prevendite entrate. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.viewmodel.amministratore

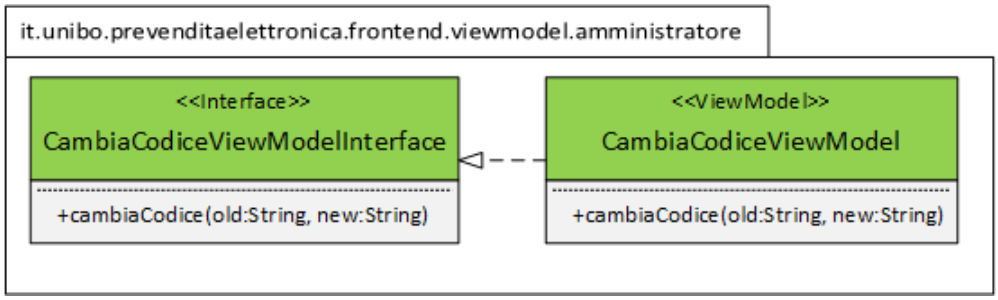

Figura 44: viewmodel.vsdx

In figura il ViewModel per cambiare il **codice di accesso** allo staff. L'interfaccia estende **Subject** per rendere possibile la registrazione della vista agli eventi.

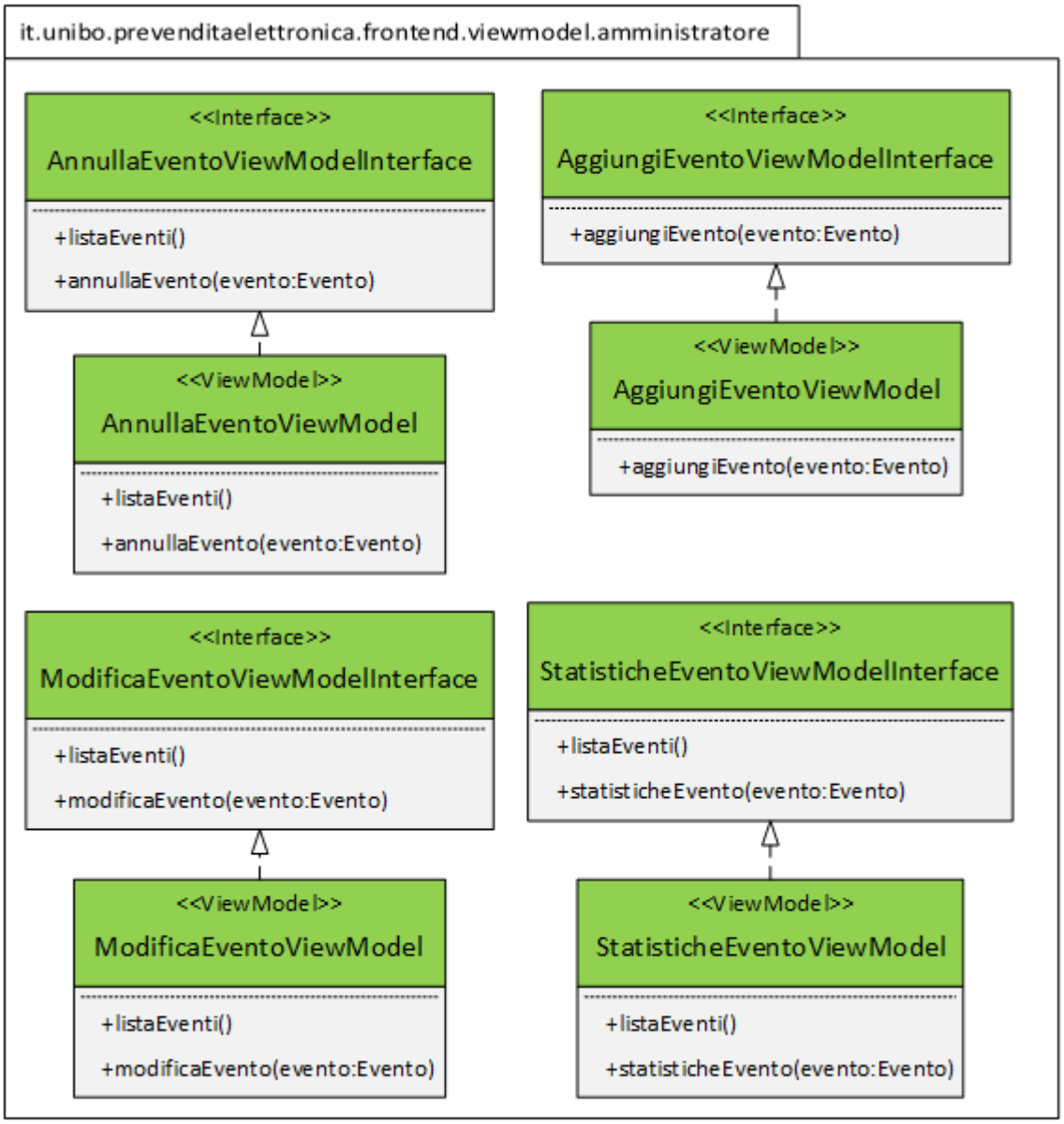

Figura 45: viewmodel.vsdx

In figura le ViewModel per la gestione di un **evento**. Alcune viste richiedono l'aggiunta di informazioni prima di eseguire l'operazione. Per esempio **AnnullaEventoViewModel** richiede la lista degli eventi tramite metodo **listaEventi**. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

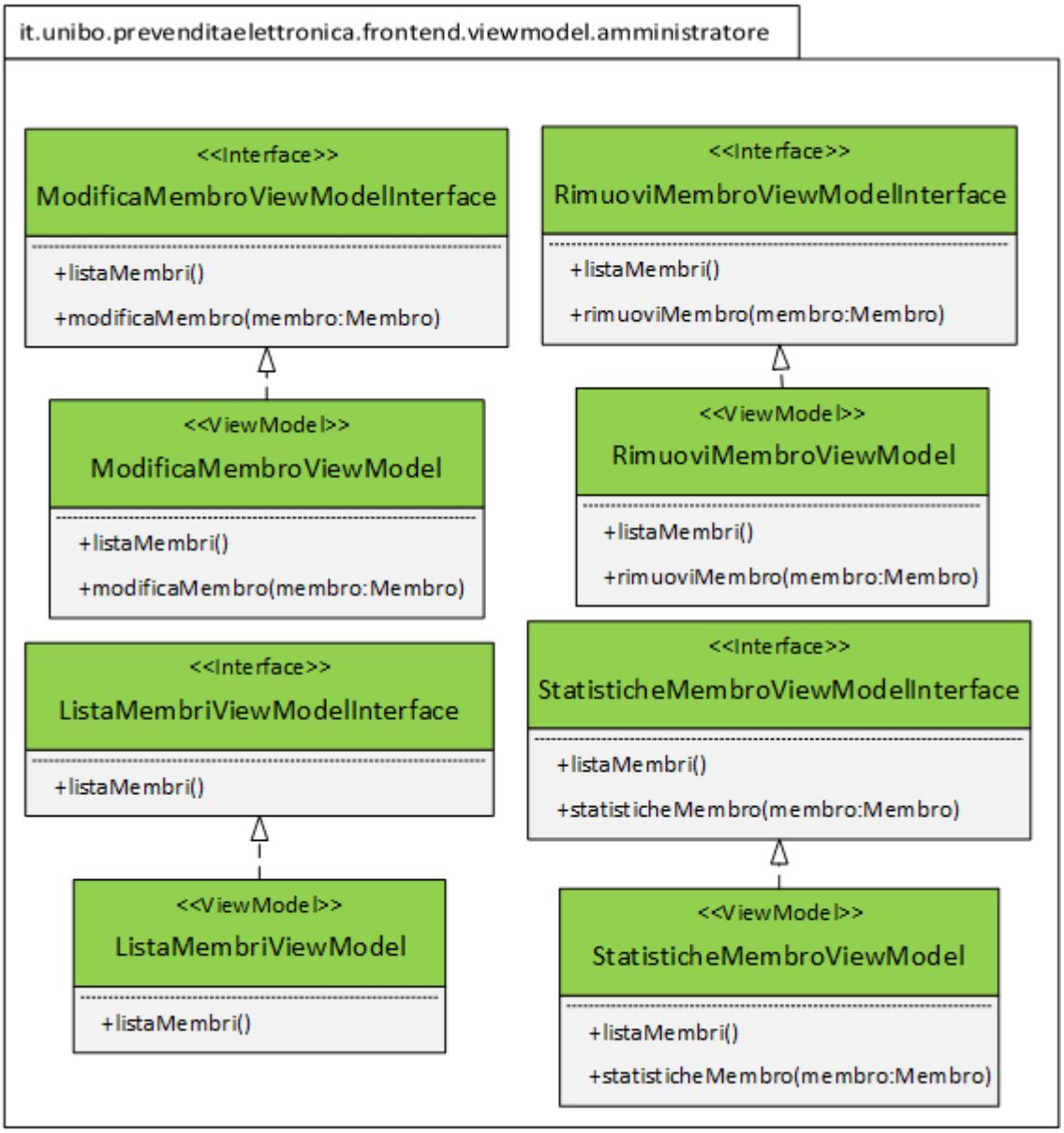

Figura 46: viewmodel.vsdx

In figura le ViewModel per la gestione dei **membri**. Alcune viste richiedono l'aggiunta di informazioni prima di eseguire l'operazione. Per esempio, prima di rimuovere un membro, ho bisogno di sapere la lista dei membri. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

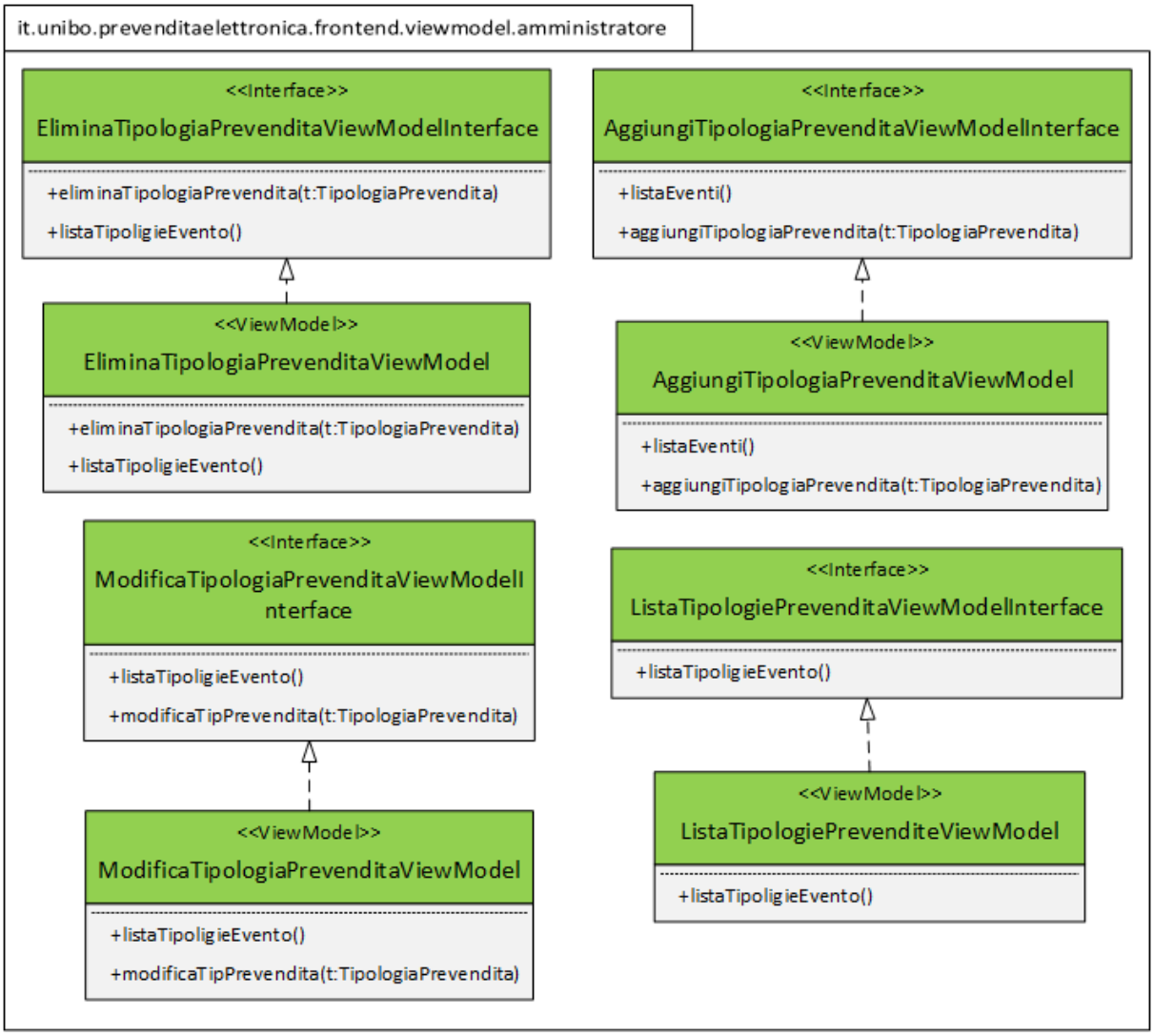

Figura 47: viewmodel.vsdx

In figura le ViewModel per la gestione delle **tipologie di prevendita**. Alcune viste richiedono l'aggiunta di informazioni prima di eseguire l'operazione. Per esempio, prima di rimuovere un membro, ho bisogno di sapere la lista dei membri. Le interfacce estendono **Subject** per rendere possibile la registrazione della vista agli eventi.

Package it.unibo.prevenditaelettronica.frontend.view

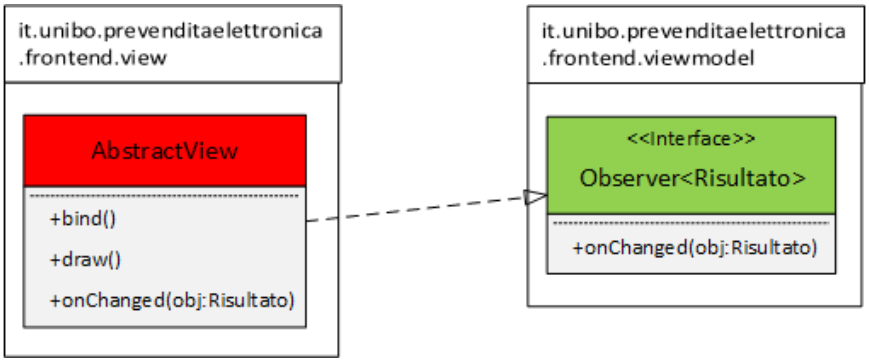

Figura 48: view.vsdx

L'**AbstractView** serve per aggiungere un comportamento comune a tutte le view: il **binding** tra View e ViewModel, tramite il metodo bind e il metodo per effettuare il draw su schermo della vista. Di conseguenza la classe abstract implementa **Observer**, per restare aggiornato sul risultato presente nel ViewModel. Tutte le viste utilizzano lo stereotipo <<View>> per indicare che estendono AbstractView.

Package it.unibo.prevenditaelettronica.frontend.view.home

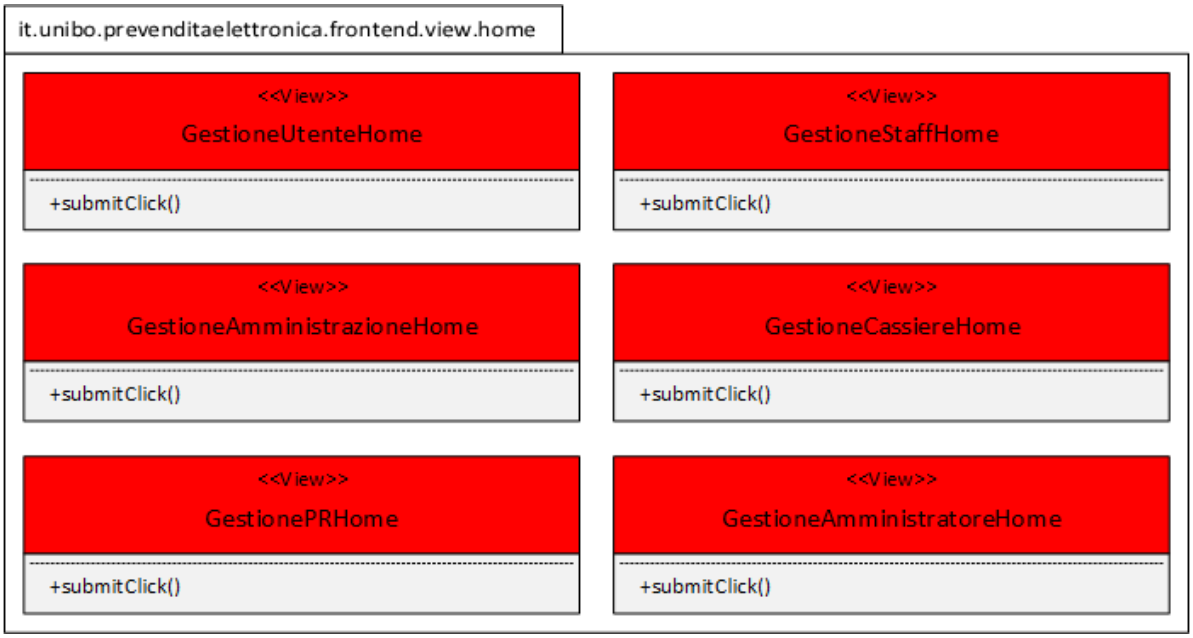

Figura 49: home.vsdx

In figura sono rappresentate le schermate home. Non necessitano di una ViewModel, in quanto non fanno altro che richiamare altre viste. Si espone solo il layout di **GestioneUtenteHome** per non appesantire troppo il documento, essendo già notevolmente ricco.

Figura 50: layout_home_utente.png

Il layout è composto da un titolo per identificare la home e una serie di pulsanti per agire sulla vista desiderata. Alcuni pulsanti potrebbero portare ad altre home invece che alla vista finale.

Package it.unibo.prevenditaelettronica.frontend.view.autenticazione

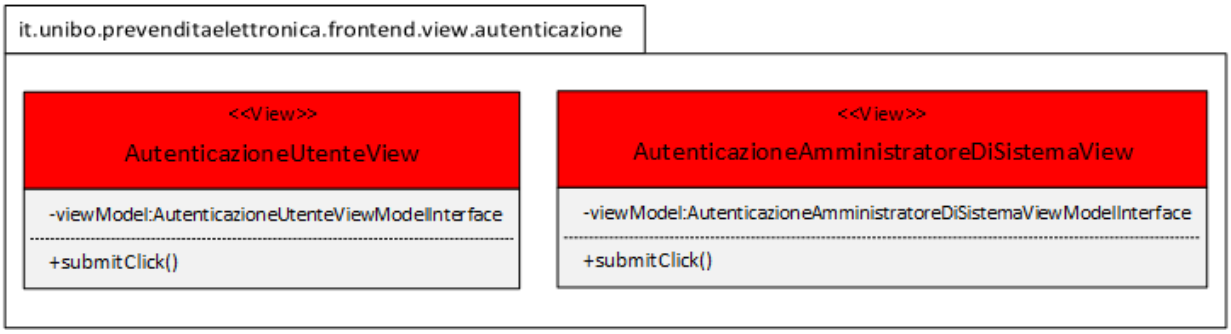

Figura 51: view.vsdx

Le viste di autenticazione offrono la schermata per il login. Il layout delle due viste è completamente interoperabile. Le altre funzionalità del package autenticazione sono state spostate in altri gruppi di viste, a seconda di come risultasse più facile per l'utente finale. Una volta che il login ha avuto successo si sceglie la home specifica: per un utente normale **GestioneUtenteHome**, mentre per un amministratore di sistema **GestioneAmministrazioneHome**.

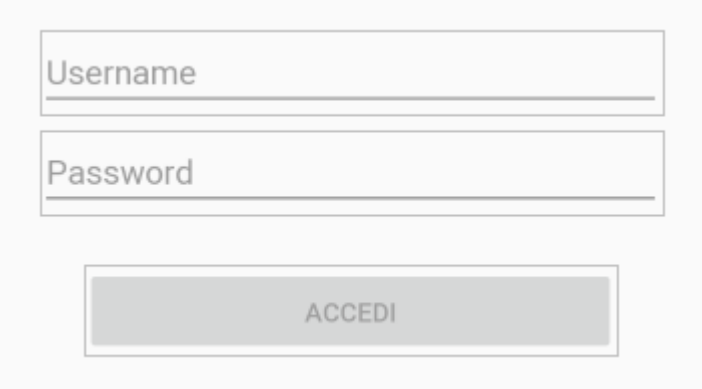

Figura 52: layout_autenticazione.png

Il layout è piuttosto semplicistico: composto da un form per la validazione delle credenziali e un pulsante di submit.

Package it.unibo.prevenditaelettronica.frontend.view.utente

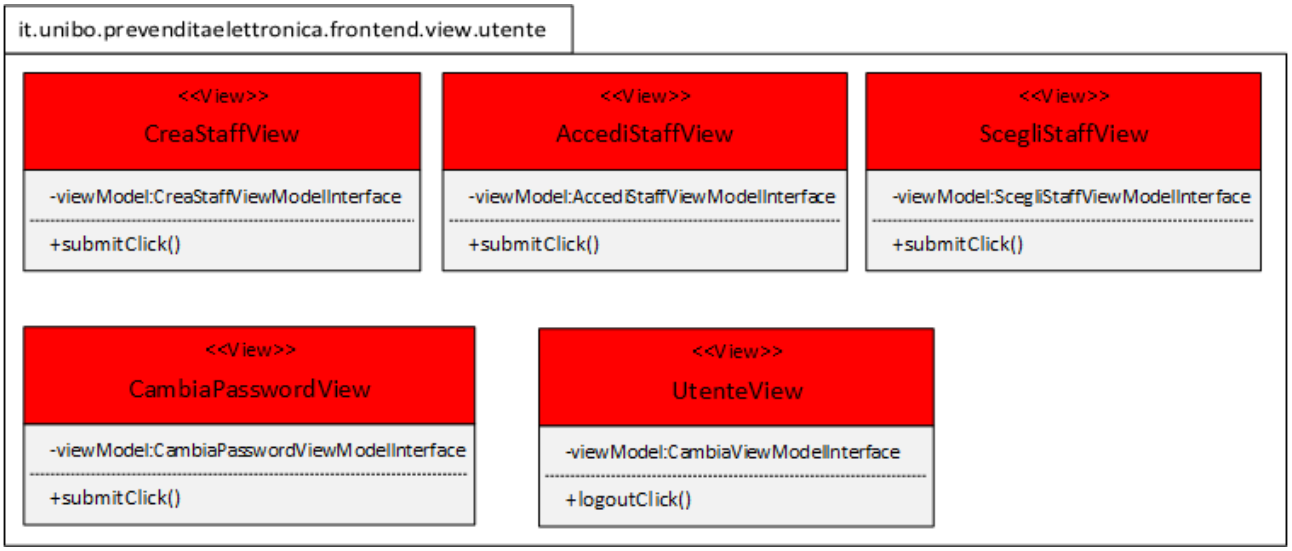

Figura 53: view.vsdx

Sono fornite le viste generali per un utente. Utente View mostra le informazioni sull'utente corrente e la possibilità di effettuare il logout. Viene mostrata solo quest'ultima vista per non appesantire il documento:

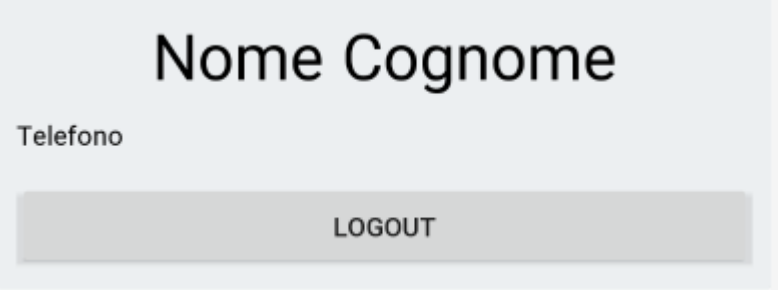

Figura 54: layout_utente.png

Viene mostrato il nome, il cognome e il telefono. Inoltre è presente il pulsante per effettuare il logout.

Package it.unibo.prevenditaelettronica.frontend.view.amministrazione

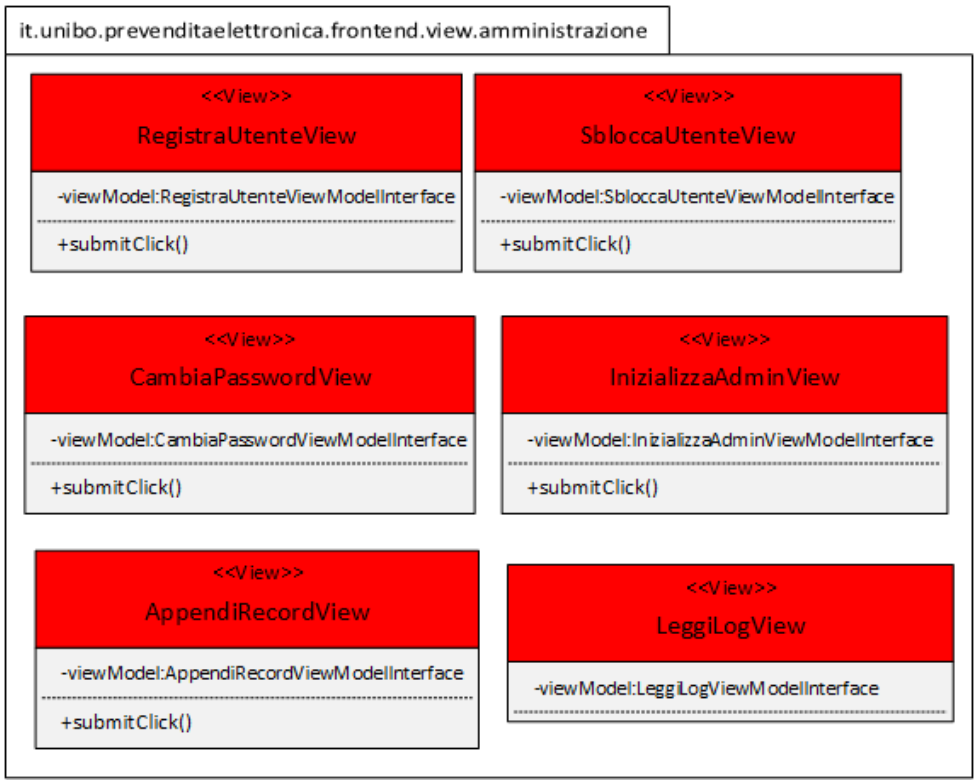

Figura 55: view.vsdx

Contiene le viste per la gestione di amministrazione di sistema. A queste viste ha accesso solo l'amministratore di sistema. Permette la lettura del log e un'aggiunta ad esso. In seguito viene mostrato il layout di aggiunta di un record di log:

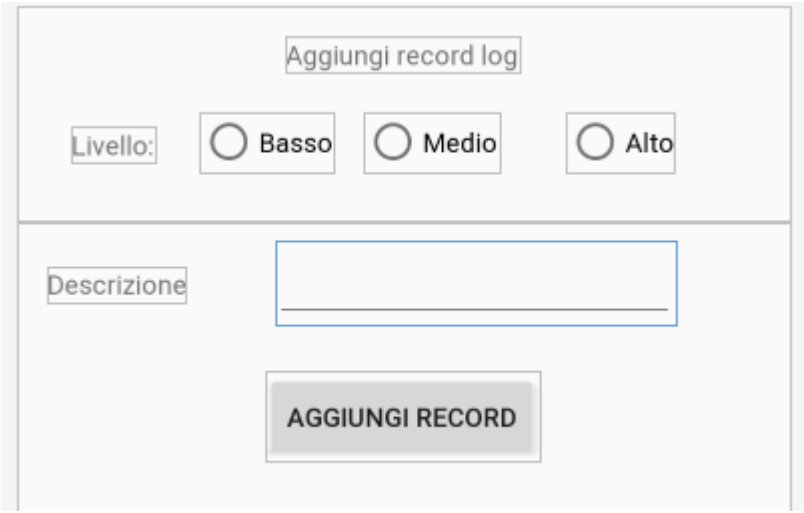

Figura 56: layout_aggiungi_record.png

Il layout mostra come è posssibile aggiungere la priorità al log e la descrizione. Il timestamp viene aggiunto automaticamente. Il pulsante aggiungi record permette l'aggiunta al log del sistema.

Package it.unibo.prevenditaelettronica.frontend.view.membro

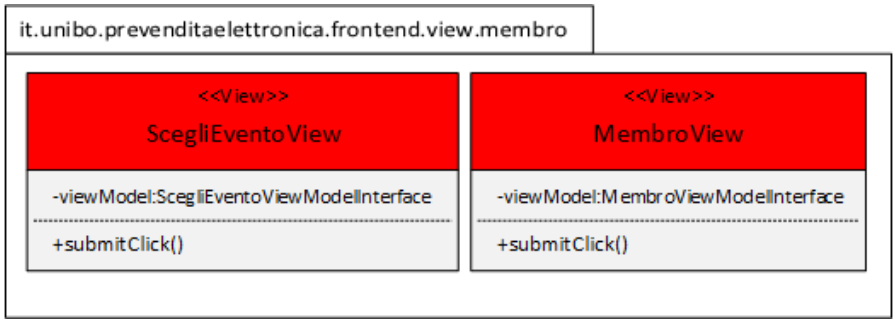

Figura 57: view.vsdx

Il package membro fornisce delle viste generali per scegliere l'evento della sessione e visualizzare statistiche sul membro. Viene mostrato il layout di **ScegliEventoView**:

Figura 58: layout_scegli_evento.png

Il layout rappresenta una lista di eventi che si possono scegliere, cliccandoci sopra. Il layout è applicabile anche a **ScegliStaffView**.

Package it.unibo.prevenditaelettronica.frontend.view.pr

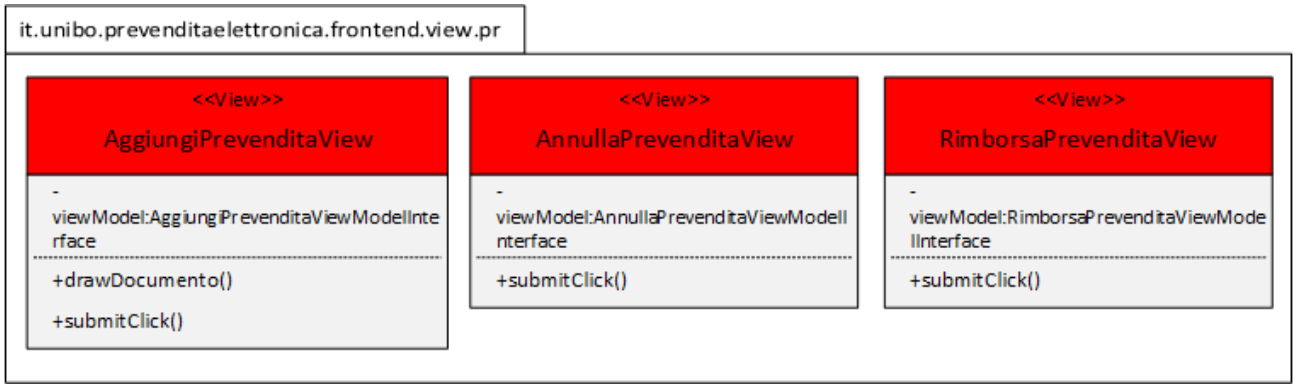

Figura 59: view.vsdx

Il package pr contiene le viste per la gestione vendita prevendita. Consente di aggiungere una prevendita, annullare una prevendita e rimborsare una prevendita. **AggiungiPrevenditaView** ha un metodo per il draw del documento digitale. Viene mostrato il layout AggiungiPrevenditaView e un esempio di documento digitale QR Code:

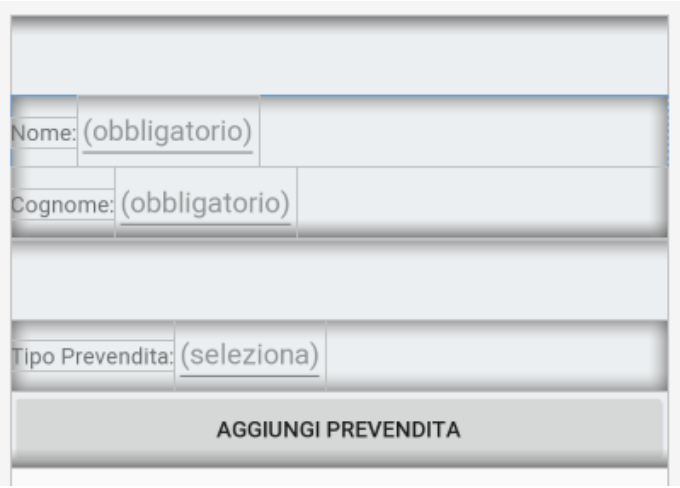

Figura 60: layout_aggiungi_prevendita.png

Il layout permette di inserire il nome e il cognome del cliente e di aggiungere la tipologia di prevendita. Il pulsante aggiungi prevendita permette di effettuare l'operazione.

Figura 61: layout_esempio_documento.png

Si tratta di un documento digitale formato da un QR Code e una descrizione. Il documento potrebbe essere implementato in un'altra forma, ma il codice QR è preferibile, in quanto permette di effettuare una scansione rapida della prevendita.

Package it.unibo.prevenditaelettronica.frontend.view.cassiere

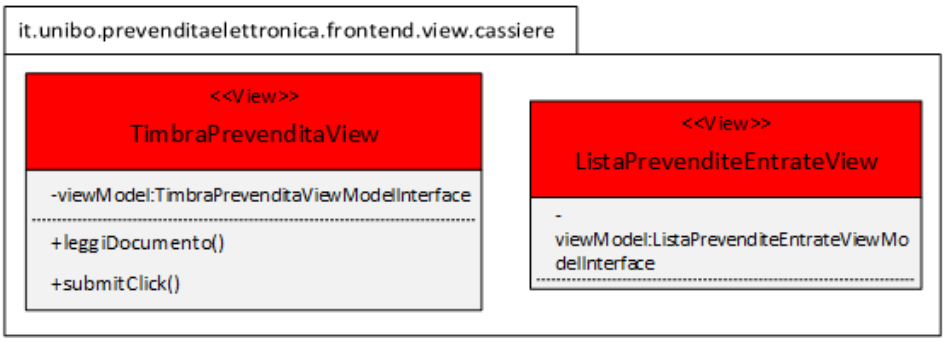

Figura 62: view.vsdx

Le viste del cassiere permettono la timbratura di una prevendita durante l'evento e la visione delle prevendite che sono entrate all'evento, per un controllo manuale. Contiene il metodo **leggiDocumento** per effetuare la scansione del documento. Viene mostrata la vista di timbratura della prevendita:

Figura 63: layout_timbra_prevendita.png

Si può vedere che nell'esempio la vista utilizza la camera per la lettura del QR Code visto nell'esempio precedente. Ovviamente il codice QR è a titolo di esempio, ma fortemente consigliato di applicarlo nell'implementazione.

Package it.unibo.prevenditaelettronica.frontend.view.amministratore

Abbiamo spezzato il package in più diagrammi vista la complessità. Anche qui vengono mostrate solo alcune viste.

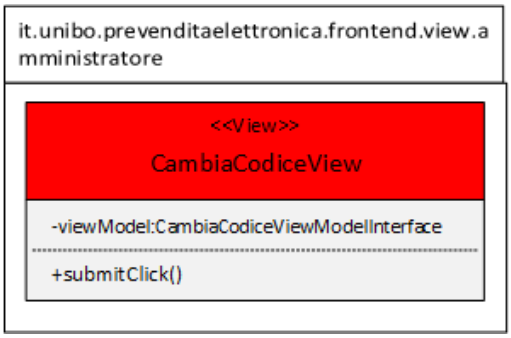

Figura 64: view.vsdx

La vista permette di cambiare codice di accesso allo staff. Qui il layout proposto per la vista:

Figura 65: layout_cambia_codice.png

Si ha un input per inserire i codici e un submit per effettuare l'operazione.

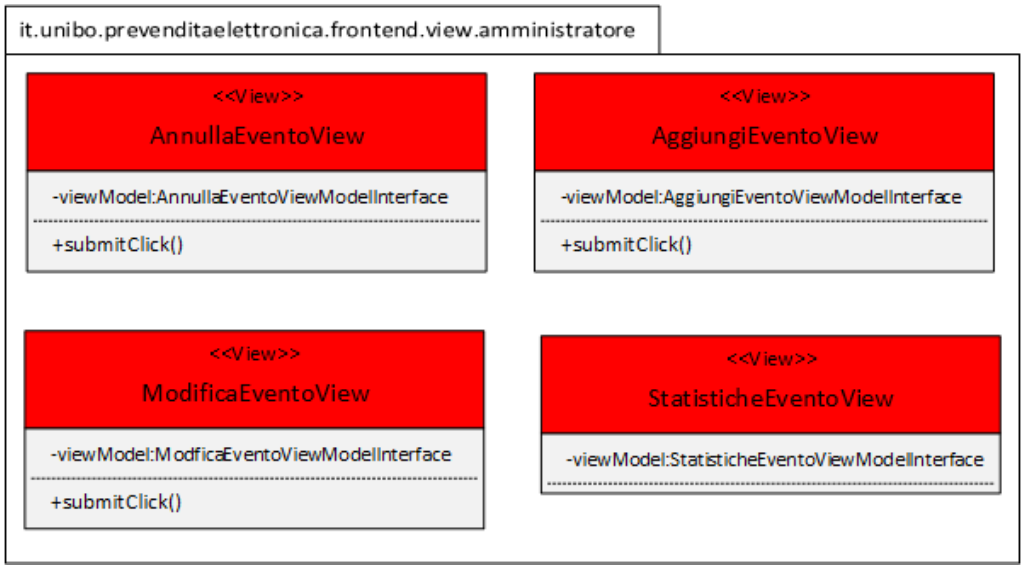

Figura 66: view.vsdx

Qui vengono mostrate le viste per la gestione degli eventi. Viene mostrata la vista di statistiche dell'evento:

Figura 67: layout_statistiche_evento.png

Viene mostrato un titolo, i quadagni e le entrate svolte per l'evento.

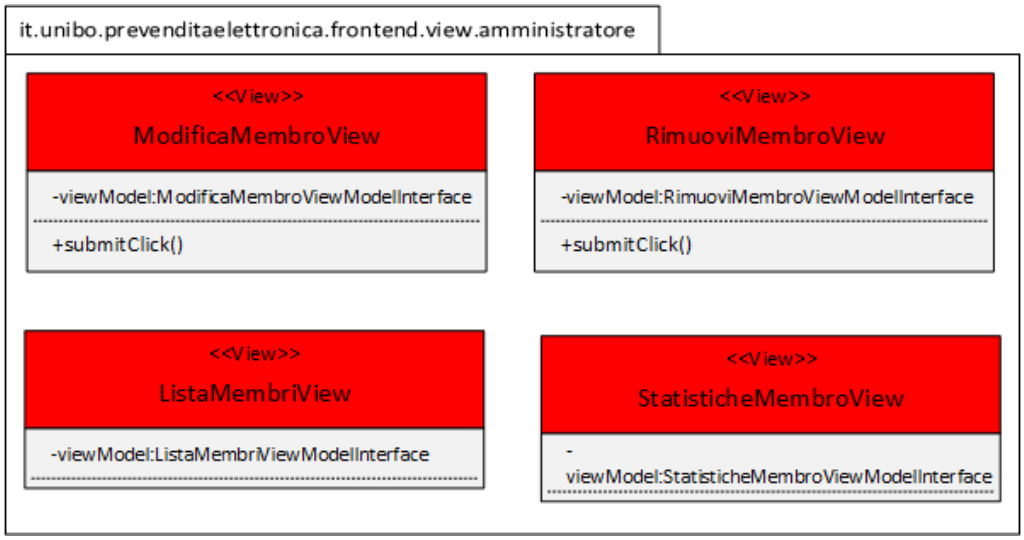

Figura 68: view.vsdx

In figura le viste per la gestione dei membri dello staff. Viene mostrato il layout della lista dei membri:

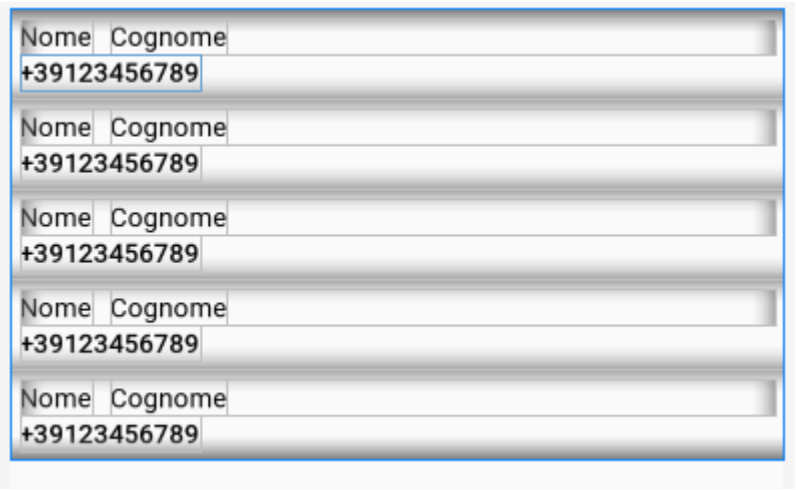

Figura 69: layout_lista_membri.png

Si tratta di una lista con le informazioni di ogni membro: nome, cognome e telefono.

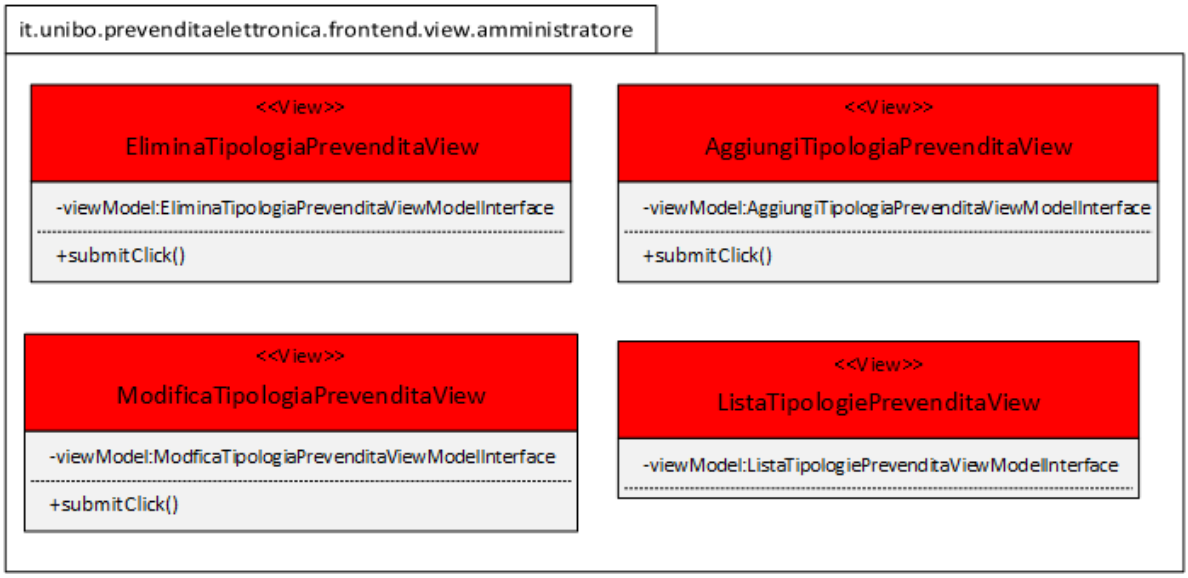

Figura 70: view.vsdx

In figura le viste per la gestione delle tipologie prevendita dell'evento scelto. Viene mostrato il layout della lista delle tipologie prevendita dell'evento scelto:

Figura 71: layout_lista_tipologie_prevendita.png

La lista prevede il nome della tipologia, il suo periodo di vendita e il prezzo di vendita.

4.3.4 Interazione

Autenticazione Utente Viene mostrato il diagramma di interazione tra utente, front-end, middleware e database per l'autenticazione dell'utente. Viene mostrato il funzionamento del pattern **Observer**, della richiesta **Asincrona** e della gestione della sessione a livello **middleware**.

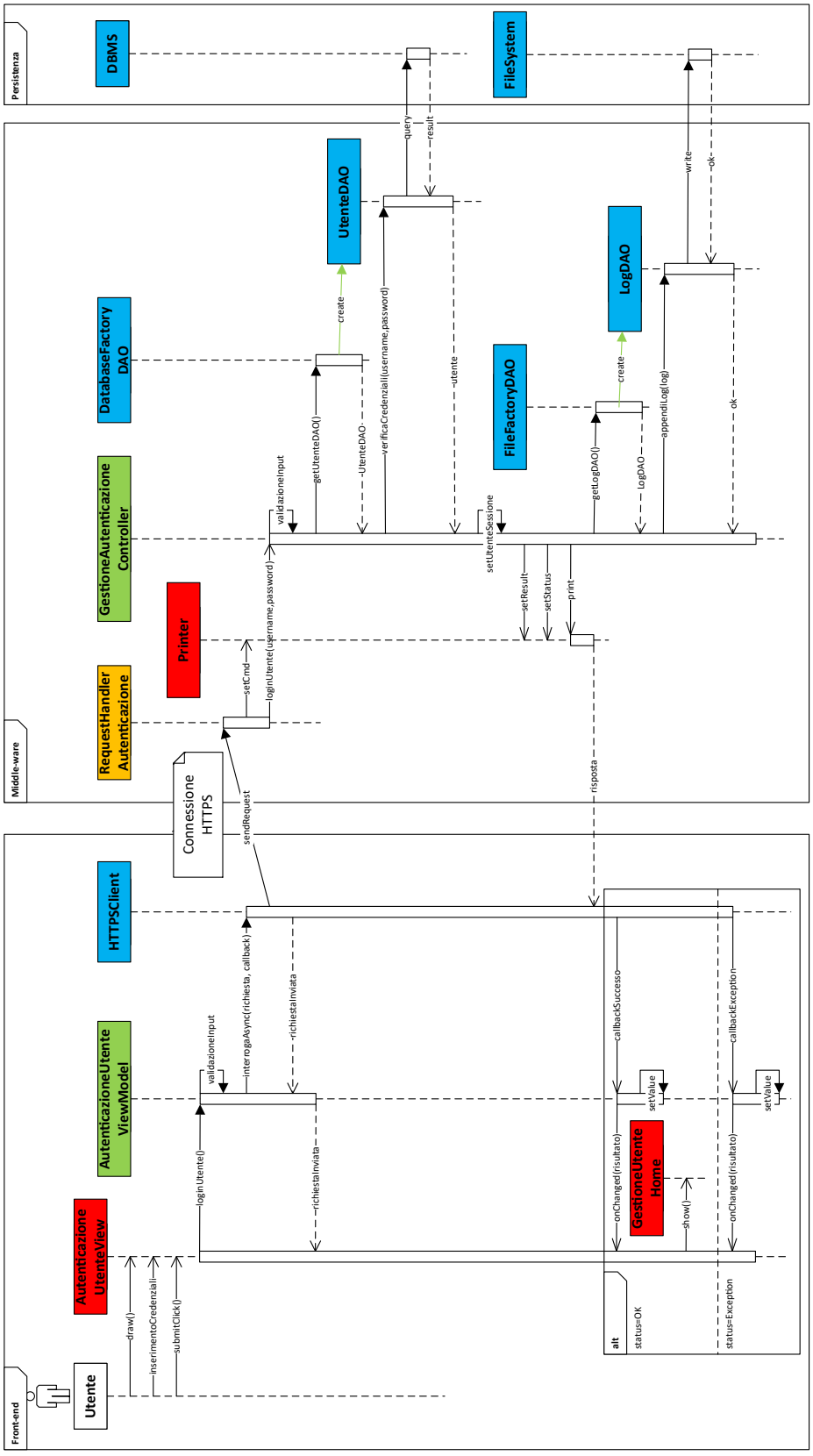

Figura 72: diagramma_sequenza_autenticazione_utente.vsdx

4.3.5 Comportamento

I diagrammi di stato principali sono presenti nella sezione Analisi. Qui si mostra l'algoritmo utilizzato da **HTTPSBrokerInterface** per scegliere il servizio middleware, tramite diagramma di flusso:

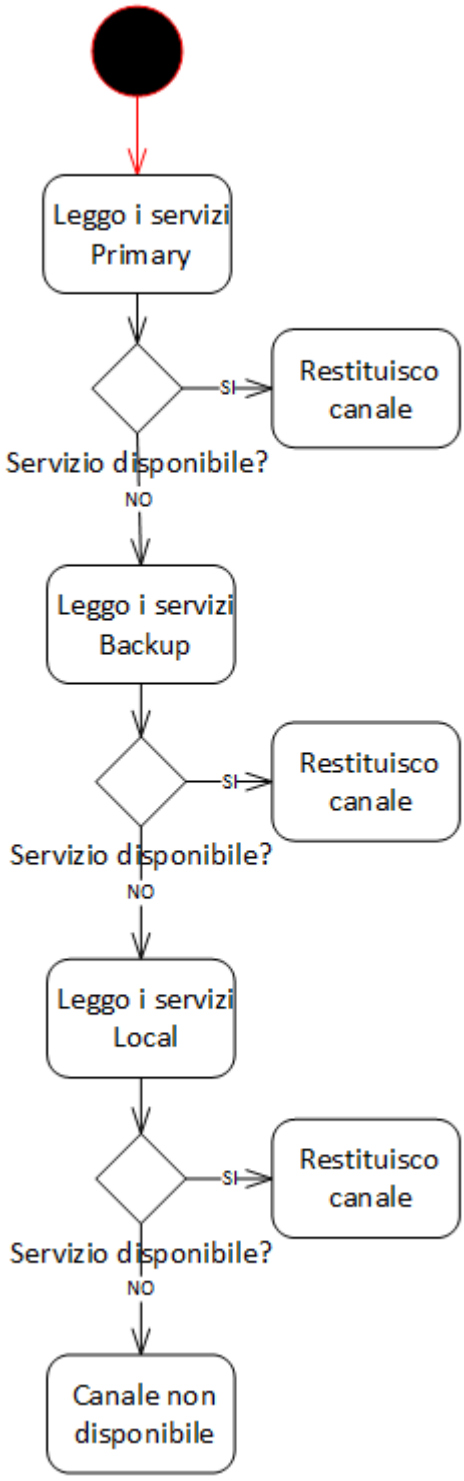

Figura 73: diagramma_algoritmo_broker.vsdx

Il diagramma di flusso mostra il broker che esegue la ricerca del canale per **priorità**. Quella più bassa è quella fornita dal servizio locale posto nell'evento in caso di attacco DoS in corso.

4.4 Persistenza

4.4.1 Schema ER

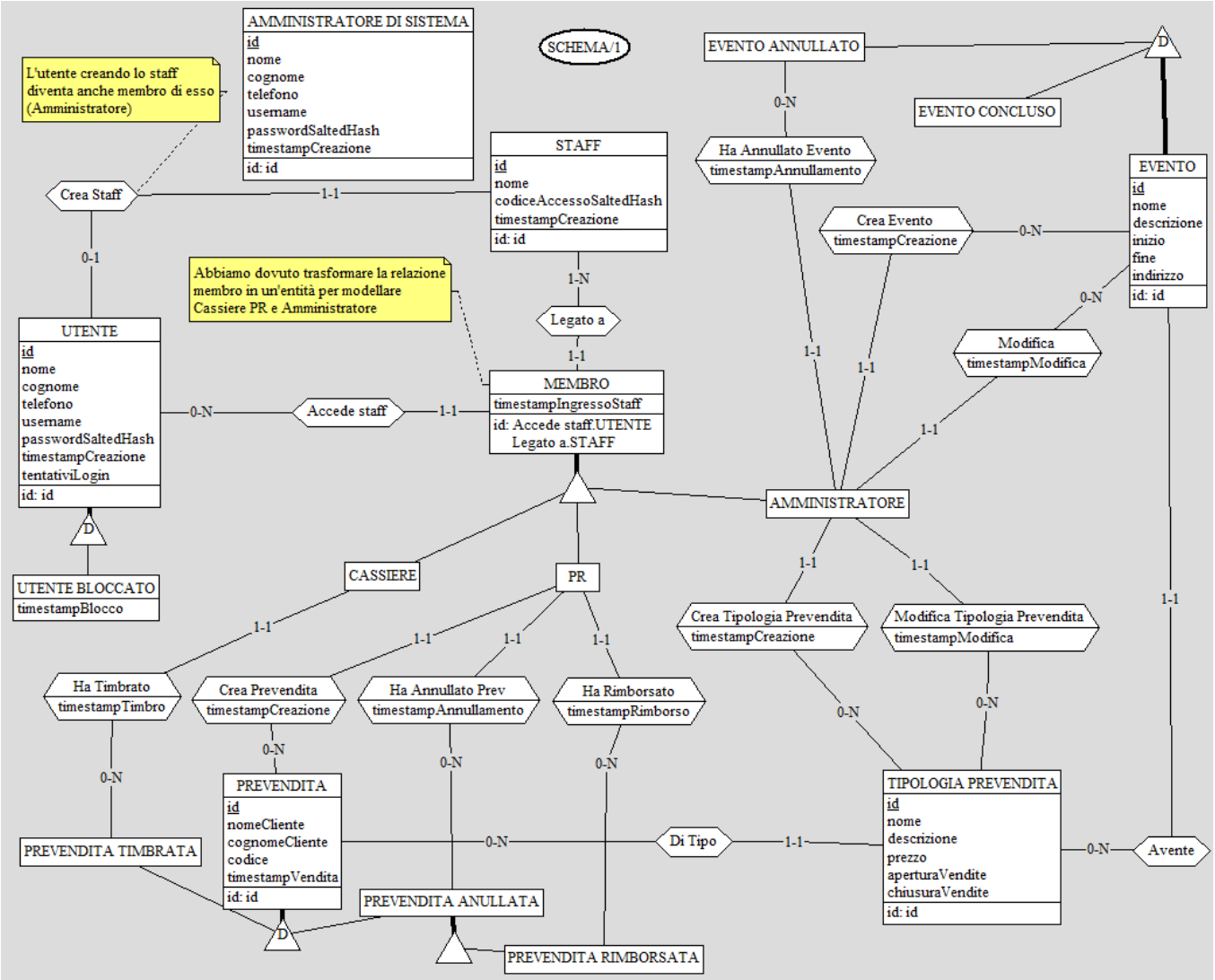

Figura 74: Progetto_Ing_Software_ER.lun

Entità Le chiavi primarie (**id**) sono di tipo intero ed autoincrement, per semplificare l'utilizzo del pattern DAO:

- **Utente**: rappresenta un utente nel sistema. **tentativiLogin** serve per rispettare il vincolo R28NF. L'attributo **username** deve essere univoco. La password viene salvata come salted hash, con meccanismo di hashing sicuro (SHA256 o superiore). La tecnica salting serve per contrastare la tecnica di attacco **Rainbow Tables**. Le stesse considerazioni vanno fatte per l'entità **Amministratore di sistema**.
- **Membro**: rappresenta un membro di uno staff. I ruoli sono modellati come sotto entità. Nel caso il membro perda il ruolo, le iterazioni del membro devono rimanere.
- **Prevendita**: gli stati della prevendita sono modellati come sotto-entità di prevendita. **codice** serve per rispettare il vincolo R27NF.
- **Evento**: gli stati del'evento sono modellati come sotto-entità di evento. Gli attributi **inizio** e **fine** rappresentano il periodo temporale in cui è in corso l'evento (R18F).
- **Amministratore di sistema**: entità a sé stante, l'**username** deve essere univoco.
- **Tipologia Prevendita**: gli attributi **aperturaVendite** e **chiusuraVendite** rappresentano il periodo temporale in cui è possibile vendere le prevendite (R20F). Questo periodo deve essere logicamente antecedente al periodo di svolgimento dell'evento (R9NF).
- **Utente Bloccato**: ci vuole un trigger che controlli **tentativiLogin**.

Relazioni Anche qui vengono mostrate in elenco solo le relazioni più importanti:

- **Membro**: In origine era una relazione molti a molti tra **utente** e **staff** per associare l'utente allo staff. Viene scomposta nell'entità **Membro** per riuscire a modellare i ruoli.
- **Crea Staff**: quando si crea lo staff, si diviene anche amministratore (R21NFbis).

Trigger Vengono inclusi trigger per far rispettare i vincoli:

- **Evento**: controllo che il periodo di svolgimento sia valido (inizio < fine, periodo nel futuro), lo stato dell'evento deve rispettare il diagramma di comportamento.
- **Tipologia Prevendita**: controllo che il periodo di vendita sia valido (inizio < fine, periodo nel futuro). Se sono state vendute prevendite non posso eliminare o modificare la tipologia (R16NF).
- **Prevendita**: consistenza dell'evento e della tipologia di prevendita. La data di vendita deve essere conforme con il periodo di vendita. L'evento deve essere valido per l'inserimento. Lo stato della prevendita deve rispettare il diagramma di comportamento.
- **Prevendita Timbrata**: Inserimento effettuato nel periodo dell'evento, l'evento deve essere valido e la prevendita deve essere valida e non annullata.
- **Membro**: Verifica del requisito R14F/NF.
- **Utente**: Verifica unicità username, un utente può creare al massimo uno staff.

Protezione della persistenza del database Essendo il database gestito su una macchina isolata remota, il rischio di accesso non autorizzato è limitato. Si consiglia di limitare l'accesso ai soli ip dello strato middleware.
Per migliorare ulteriormente la sicurezza, si consiglia di spostare l'entità amministratore di sistema in un RDMBS separato.

Viste Un'ulteriore aggiunta, alla sicurezza e alla semplificazione della logica di business, può essere l'introduzione di viste di accesso al database:

Vista	Entità	Accesso
Amministratore di Sistema	Amministratore Di Sistema	Lettura/Scrittura
Amministratore di Sistema	Utente	Lettura/Scrittura
Utente	Utente	Lettura/Scrittura
Utente	Staff	Lettura/Scrittura
Utente	Evento	Lettura/Scrittura
Utente	Membro	Lettura/Scrittura
Utente	Tipologia Prevendita	Lettura/Scrittura
Utente	Prevendita	Lettura/Scrittura
Utente	Statistiche Membro	Lettura
Utente	Statistiche Evento	Lettura

Integrità della persistenza del database Si devono effettuare transazioni dove necessario, con dovuti **savepoint** e **rollback** in caso di errore, evitando stati non consistenti del database.

4.4.2 Formato del file del log

Il file di log deve contenere informazioni utili per l'amministratore di sistema. Sono raccolte molte operazioni svolte, soprattutto quelle ritenute importanti. L'amministratore di sistema può aggiungere note personali.

```
1    <timestamp> <livello> <descrizione>
2    <timestamp> <livello> <descrizione>
3    <timestamp> <livello> <descrizione>
4    <timestamp> <livello> <descrizione>
```

Protezione della persistenza dei log Essendo il file salvato nel livello middleware, il rischio di accesso di malintenzionati si riduce, ma per migliorare la protezione, bisogna impostare i giusti permessi utente nel file system. Ovviamente solo l'amministratore di sistema può accedere a queste informazioni.

4.5 Collaudo

Data la natura multi-layer non prevista nella sezione di analisi, sono stati introdotti specifici test sia per il layer di front-end, sia per il layer di middleware. I test introdotti sono molti, ma vengono mostrati solo due test, uno per lo strato di middleware e un altro per lo strato front-end.

```
1  package it.unibo.prevenditaelettronica.frontend.model.net.broker;
2
3  import org.junit.Before;
4  import org.junit.Test;
5
6  import static org.junit.Assert.assertEquals;
7
8  public class TestHTTPSBroker {
9
10    //Serie di oggetti finti per simulare un servizio middleware
11    private MiddlewareFakeService serviceA;
12    private MiddlewareFakeService serviceB;
13    private MiddlewareFakeService serviceC;
14
15    //Classe da testare
16    private HTTPBrokerInterface broker;
17
18     /**
19      * Inizializzazione di una prova di HTTPSBroker.
20      */
21     @Before
22     public void setup(){
23        //Inizializzazione oggetti mock: ogniuno in listen su una porta locale diversa.
24        serviceA = new MiddlewareFakeService("https://localhost:8080");
25        serviceB = new MiddlewareFakeService("https://localhost:8081");
26        serviceC = new MiddlewareFakeService("https://localhost:8082");
27
28        //Inizializzazione broker
29        broker = new HTTPSBroker();
30     }
31
32
33     /**
34      * Test del broker a vuoto.
35      */
36     @Test
37     public void voidBrokerTest(){
38
39         //Non registro i servizi e vedo cosa restituisce
40         assertEquals(null, broker.procuraCanale());
41     }
42
43     /**
44      * Test del broker con la (de)registrazione dei servizi
45      */
46     @Test
47     public void registerServiceBrokerTest(){
48
49         broker.registraServizio("https://localhost:8080", ServicePriority.PRIMARY);
50         broker.registraServizio("https://localhost:8080", ServicePriority.BACKUP);
51         broker.registraServizio("https://localhost:8080", ServicePriority.LOCAL);
52
53         //Dovrebbe restituire il primario
54         assertEquals("https://localhost:8080", broker.procuraCanale().getUrl());
55
56         //Stacco il servizio primario e rifaccio il test
57         broker.disdiciServizio("https://localhost:8080");
58         assertEquals("https://localhost:8081", broker.procuraCanale().getUrl());
59
60         //Stacco il servizio backup e rifaccio il test
61         broker.disdiciServizio("https://localhost:8081");
62         assertEquals("https://localhost:8082", broker.procuraCanale().getUrl());
```

```
    }

    /**
     * Test del broker simulando un attacco DoS.
     */
    @Test
    public void dosServiceBrokerTest(){

        broker.registraServizio("https://localhost:8080", ServicePriority.PRIMARY);
        broker.registraServizio("https://localhost:8080", ServicePriority.BACKUP);
        broker.registraServizio("https://localhost:8080", ServicePriority.LOCAL);

        //Dovrebbe restituire il primario
        assertEquals("https://localhost:8080", broker.procuraCanale().getUrl());

        //Simulo un attacco sul servizio primario e rifaccio il test
        serviceA.dos();
        assertEquals("https://localhost:8081", broker.procuraCanale().getUrl());

        //Simulo un attacco sul servizio backup e rifaccio il test
        serviceB.dos();
        assertEquals("https://localhost:8081", broker.procuraCanale().getUrl());

        //Simulo un attacco sul servizio local e rifaccio il test
        serviceC.dos();
        assertEquals(null, broker.procuraCanale().getUrl());
    }

}
```

```
package it.unibo.prevenditaelettronica.middleware.view;

import org.junit.Before;
import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class TestJSONPrinter {

    //Classe da testare
    private JSONPrinter printer;

    //Serve per simulare una connessione HTTPS, per leggere il risultato JSON prodotto.
    private HTTPSInterfaceMock httpsMock;

    /**
     * Inizializzazione di una prova di JSONPrinter.
     */
    @Before
    public void setup(){
        httpsMock = new HTTPSInterfaceMock();
        printer = new JSONPrinter(httpsMock);
    }


    /**
     * Test di una stampa JSON di un risultato.
     */
    @Test
    public void printResultTest(){

        //Imposto la printer
        printer.reset();
        printer.setComando(Comando.ECHO);
        printer.setStatus(Stato.OK);
        printer.setResult(null);
        printer.printResponse();

```

```
39          String sniffResponseJSON = httpsMock.sniffing();
40
41          assertEquals("{comando:\"ECHO\", stato:\"OK\", risultato:null}",
                sniffResponseJSON);
42
43          //Test Comando non nullo
44          Timestamp now =  Timestamp.now();
45
46          //Imposto la printer
47          printer.reset();
48          printer.setComando(Comando.TIMESTAMP);
49          printer.setStatus(Stato.OK);
50          printer.setResult(now);
51          printer.printResponse();
52
53          String sniffResponseJSON = httpsMock.sniffing();
54
55          assertEquals("{comando:\"TIMESTAMP\", stato:\"OK\", risultato:\""+ now.toString
                () +"\"}", sniffResponseJSON);
56      }
57
58       /**
59       * Test di una stampa JSON di una eccezione
60       */
61      @Test
62      public void printExceptionTest(){
63
64          //Imposto la printer
65          printer.reset();
66          printer.setComando(Comando.ECHO);
67          printer.setStatus(Stato.EXCEPTION);
68          printer.setException(new Exception("Prova"));
69          printer.printResponse();
70
71          String sniffResponseJSON = httpsMock.sniffing();
72
73          assertEquals("{comando:\"ECHO\", stato:\"EXCEPTION\", exception:{cause:\"Prova
                \"}}", sniffResponseJSON);
74      }
75
76
77  }
```

4.6 Deployment

Il deployment del sistema avviene in due modi separati, a seconda del layer middleware o front-end. Il deployment dello strato intermedio viene gestito dall'amministratore di sistema, egli si occupa anche della procedura di aggiornamento. A seconda dell'implementazione, verrà distribuito all'amministratore di sistema un pacchetto automatico o manuale per l'aggiornamento del sistema. Per il deployment lato front-end sarà previsto un check automatico da parte del client. In caso di aggiornamento, viene mostrato un pop-up all'utente:

Figura 75: aggiornamento_dialog.png

5 Implementazione

5.1 Scelte tecnologiche

Nella fase di implementazione abbiamo deciso di implementare lo strato di persistenza utilizzando il RDBMS MySQL, mentre lo strato middleware viene implementato usando il linguaggio PHP 7. Abbiamo deciso di implementare lo strato front-end suddividendolo in tre client, uno web-based per la gestione di amministrazione di sistema, un altro web-based per l'amministrazione dello staff e infine un client mobile sviluppato sotto la piattaforma Android per il cassiere e il pr, dato che il loro lavoro viene fatto in vari ambienti. La scelta di queste tecnologie è pensata per risolvere il requisito R5NF. Infatti la maggior parte delle soluzioni hosting gratuite offre queste tecnologie di default. La scelta di Android invece che l'utilizzo di iOS è dovuta sempre per lo stesso motivo: per sviluppare con iOS serve hardware proprietario. Android adotta Java come linguaggio di sviluppo. I client web-based sono implementati mediante una combinazione di HTML+CSS+Javascript, con varie librerie come JQuery e Bootstrap, utilizzando la tecnica AJAX per le chiamate allo strato middleware. I client web-based si trovano nello stesso spazio di hosting dello strato middleware.

5.2 Scelte Implementative

- Ridotto enumeratore ServicePriority da PRIMARY, BACKUP, LOCAL a PRIMARY, LOCAL. Il servizio di hosting non espone il database all'esterno e quindi non è possibile replicare lo strato middleware.
- Package DAO del middleware implementato usando PDO, che supporta molti rdbms.
- Utilizzo di chiavi surrogate per identificare gli oggetti, anche a lato client.
- Implementazione della Sessione mediante sessione gestita da PHP.
- Nel client Android, viene usata la libreria Volley come HTTPSClient.
- Nel client Android viene usata la libreria GSON per la serializzazione/deserializzazione delle informazioni. Le interfacce JSONSerializable e JSONDeserializable diventano superflue utilizzando la libreria.
- Nel database le credenziali vengono salvate come coppia (username, salted hash), utilizzando come algoritmo di hashing SHA256.
- Nello strato middleware, l'interfaccia HTTPSInterface è superflua, in quanto la comunicazione HTTPS è intrinseca nel linguaggio PHP.
- Il certificato HTTPS viene fornito dal servizio di hosting.
- Nello strato middleware non c'è bisogno delle interfacce JSONSerializable e JSONDeserializable, in quanto viene usata la libreria integrata nel linguaggio.
- Per rendere sicuro la cartella dei log è stato utilizzato un file .htaccess, per restringerne l'accesso.
- I client web-based non hanno bisogno dell'interfaccia HTTPClient, in quanto utilizzano il browser stesso.
- I client web-based non utilizzano il broker in quanto il servizio di replicazione non è necessario per le funzionalità dei client web-based.
- I client web-based non hanno bisogno delle interfacce JSONSerializable e JSONDeserializable, in quanto viene usata la libreria integrata nel browser.
- HTTPSBroker è stato implementato in maniera diversa: il broker fornisce solo gli URL, è HTTPSClient a verificare lo stato del servizio, e in caso fosse necessario, richiede l'URL alternativo al broker.
- Non è stato possibile separare l'entità Utente e l'entità AmministratoreDiSistema in due database distinti, neppure un accesso al database con autorizzazioni diverse. Il servizio di hosting offre un solo database con credenziali di accesso uniche.
- Come consigliato, abbiamo scelto di utilizzare la tecnologia QRCode come documento digitale. Per la consegna al cliente viene mandato direttamente la foto, oppure un link dove è possibile generare il QRCode sul momento.
- Viene aggiunta una vista nel client Android per aggiungere l'URL del servizio replicato in locale.
- Per replicare il servizio in locale, l'amministratore di sistema esegue un backup del database mediante l'interfaccia fornita dal servizio di hosting, e utilizzando il software XAMPP replica il servizio. I client dovranno aggiungere il servizio tramite il broker.

- Si è deciso di ampliare le statistiche, catalogandole per tipologia di prevendita.
- Nel client Android abbiamo modificato alcuni ViewModel del front-end, rendendoli osservatori dei form delle viste, per rendere la validazione dell'input in tempo reale.
- Le connessioni al database sono sincronizzate con il fuso orario dello strato middleware e dati temporali ricevuti e inviati ai client sono formattati con standard **ISO8601**: così facendo, si rende l'applicativo utilizzabile in un contesto globale.

5.3 Collaudo

Il collaudo su middleware è stato integrato mediante libreria PHPUnit. Nel client Android viene usata la libreria JUnit, integrata nell'ambiente di sviluppo. Per testare il codice Javascript, presente nei client web-based abbiamo deciso di implementare i test senza l'ausilio di librerie esterne, ma utilizzando il Web Developer Kit per monitorare il codice di testing.

6 Deployment

6.1 Note di configurazione sul middleware

Il servizio middleware deve essere configurato dall'amministatore di sistema, configurando l'accesso al database:

```
<?php

define('TIMEZONE', 'UTC');
date_default_timezone_set(TIMEZONE);

$GLOBALS["databaseURL"] = "localhost";
$GLOBALS['databaseType'] = "mysql";
$GLOBALS['databaseName'] = "my_prapp";
$GLOBALS['databaseUsername'] = "root";
$GLOBALS['databasePassword'] = "";
$GLOBALS['databaseCharset'] = "utf8";

$GLOBALS['defaultSystemAdministatorUsername'] = "admin";
$GLOBALS['defaultSystemAdministatorPassword'] = "admin";
```

6.2 Note di configurazione sul front-end

I client web-based sono configurati dall'amministratore di sistema, in quanto sono salvati nello spazio di hosting che ospita anche lo strato middleware:

```
var middlewareURL = "https://prapp.altervista.org";
```

Il client Android richiede solamente di inserire l'URL del servizio middleware, fornito dall'amministratore di sistema.

6.3 Artefatti

Figura 76: artefatti.vsdx

6.4 Deployment Type-Level

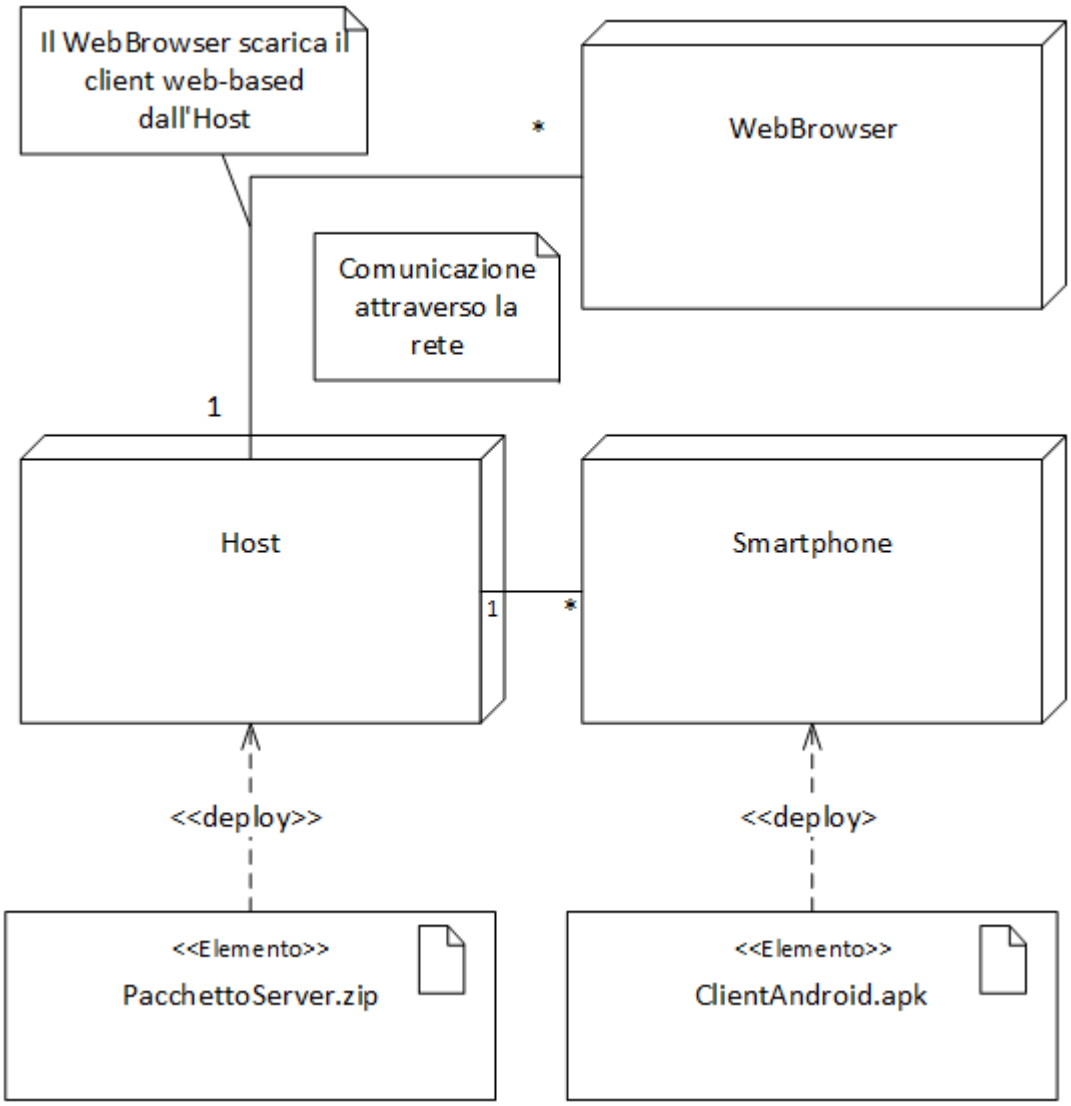

Figura 77: type_deployment.vsdx