

OikoNaos
Requirements Analysis
Versione 3.7



Data: 11/11/2025

Progetto: OikoNaos	Versione: 3.7
Documento: Requirement Analysis	Data: 11/11/2025

Coordinatore del progetto:

Nome	Matricola
Luigi Potestà	0512120076

Partecipanti:

Nome	Matricola
Luigi Potestà	0512120076
Maria Antonietta Ruggiero	0512121126
Giulia Buonafine	0512119695
Mario Sinopoli	0512119113

Scritto da:	Giulia Buonafine
-------------	------------------

Revision History

Data	Versione	Descrizione	Autore
26/09/2025	1.0	Proposta di Progetto: Formazione del gruppo, inserimento dei partecipanti e prima bozza della descrizione del progetto	Giulia Buonafine
29/09/2025	1.1	Proposta di Progetto: Stesura del titolo di progetto, prime specifiche del progetto	Mario Sinopoli
30/09/2025	1.2	Proposta di Progetto: Inserimento del logo e stesura dei ruoli	Luigi Potestà
01/10/2025	1.3	Proposta di Progetto: Decisione e stesura ultime specifiche di progetto	Maria Antonietta Ruggiero
09/10/2025	2.0	Problem Statement: Scrittura dominio del problema, definizione audience e scopo del sistema	Mario Sinopoli
10/10/2025	2.1	Problem Statement: Definizione dominio del problema e requisiti funzionali	Luigi Potestà

10/10/2025	2.2	Problem Statement: Realizzazione scenari e inserimento scadenze	Giulia Buonafine
13/10/2025	2.3	Problem Statement: Inserimento requisiti non funzionali e ridefinizione ambiente di sviluppo	Maria Antonietta Ruggiero
21/10/2025	3.0	Requirements Analysis: Creazione struttura del documento e definizione del purpose e audience	Mario Sinopoli
23/10/2025	3.1	Requirements Analysis: Scrittura della parte introduttiva dal punto 1.1.1 a 1.1.6	Maria Antonietta Ruggiero
26/10/2025	3.2	Requirements Analysis: Definizione del Sistema Attuale, Sistema Proposto e modelli di sistema	Giulia Buonafine, Luigi Potestà
05/11/2025	3.3	Requirements Analysis: Ridefinizione e miglioramento Casi d'uso e Requisiti funzionali	Maria Antonietta Ruggiero Giulia Buonafine
06/11/2025	3.4	Requirements Analysis: Inserimento e definizione diagramma delle classi con relative tabelle	Luigi Potestà Mario Sinopoli
07/11/2025	3.5	Requirements Analysis: Inserimento Navigation Path e diagrammi di sequenza	Giulia Buonafine Maria Antonietta Ruggiero
08/11/2025	3.6	Requirements Analysis: Inserimento diagrammi di stato	Luigi Potestà Mario Sinopoli
11/11/2025	3.7	Requirements Analysis: Revisione progetto con miglioramenti	Mario Sinopoli Luigi Potestà Maria Antonietta Ruggiero Giulia Buonafine

Indice

1.	Purpose	5
2.	Audience.....	5
2.1	Purpose of the System.....	5
2.2	Scope of the System.....	6
2.3	Objectives and success criteria of the project	6
2.4	Definitions, acronyms and abbreviations	6
2.5	Overview	6
3.	Current System	7
4.	Proposed System.....	7
4.1	Overview	7
4.2	Functional Requirements	8
4.3	Nonfunctional Requirements	9
4.3.1	Usability	9
4.3.2	Reliability	9
4.3.3	Performance	9
4.3.4	Supportability	9
4.3.5	Implementation	9
4.3.6	Interface	10
4.3.7	Packaging	10
4.3.8	Security	10
4.3.9	Legal	10
4.4	System Models.....	10
4.4.1	Scenarios.....	10
4.5	Use case model	11
4.5.1	Object model.....	17
4.5.2	Dynamic model	29
4.5.3	User interface	37
5.	Glossary	43

1. Purpose

Il presente documento di Requirements Analysis (RAD), raccoglie e formalizza i risultati delle attività di requirements elicitation e di analysis relative al progetto OikoNaos, una piattaforma web per la gestione integrata delle comunità di co-housing. Il RAD ha lo scopo di descrivere il sistema in termini di requisiti funzionali e non funzionali, traducendo in specifiche tecniche le necessità espresse dal cliente, ossia l’Azienda Coordinatrice. Questo documento costituisce una base contrattuale tra cliente e gruppo di sviluppo, assicurando che le funzionalità attese siano interpretate correttamente e che ogni requisito possa essere tracciato lungo tutto il ciclo di vita del software. Esso rappresenta inoltre un riferimento metodologico per le fasi successive di progettazione, implementazione e verifica, fornendo una visione completa e coerente del sistema da realizzare.

2. Audience

Il presente documento è destinato a tutte le figure coinvolte nel processo di analisi e sviluppo del sistema OikoNaos.

In primo luogo, esso si rivolge al cliente, identificato nell’Azienda Coordinatrice, ente responsabile della gestione e supervisione delle comunità di co-housing. Tale figura rappresenta il committente del sistema e utilizzerà il documento per verificare che le specifiche rispondano alle esigenze funzionali e organizzative espresse.

Il RAD è altresì riferito agli utenti finali, ossia i coinvilini, che, pur non partecipando direttamente alla fase di sviluppo, costituiscono i principali beneficiari del sistema e i soggetti attorno ai quali sono stati definiti i requisiti funzionali e di usabilità.

Il documento è inoltre indirizzato al gruppo di sviluppo OikoNaos, che riveste simultaneamente i ruoli di project management, system analysts e system designers. Il team è infatti responsabile della pianificazione, della raccolta dei requisiti, della modellazione concettuale e della progettazione architettonale del sistema.

Introduzione

2.1 Purpose of the System

Il sistema OikoNaos nasce con l’obiettivo di fornire una piattaforma digitale completa per la gestione e l’organizzazione di comunità abitative di tipo co-housing, supervisionate da un’Azienda Coordinatrice. Il sistema intende centralizzare in un unico ambiente web tutte le attività operative e comunicative legate alla vita comunitaria, sostituendo strumenti frammentati e processi manuali con un modello informatizzato efficiente, trasparente e sicuro.

OikoNaos consente ai residenti di accedere ai principali servizi della comunità: la prenotazione di spazi comuni, la segnalazione di guasti e la gestione dei ticket di manutenzione, la partecipazione ad eventi e la consultazione della bacheca digitale, nonché il controllo delle spese e dei contributi periodici. Parallelamente, offre all’Azienda Coordinatrice e ai responsabili della struttura un insieme di funzionalità di monitoraggio e supervisione che semplificano l’amministrazione, la gestione dei turni e il coordinamento delle risorse comuni.

Il suo scopo ultimo è quello di favorire la collaborazione, migliorare la comunicazione e promuovere un senso di comunità attraverso la tecnologia.

2.2 Scope of the System

L'ambito di applicazione del sistema OikoNaos comprende l'intero ecosistema gestionale di una comunità di co-housing. Esso copre i processi di autenticazione, la gestione dei profili personali, la prenotazione e modifica delle risorse condivise, la creazione e il tracciamento dei ticket di assistenza, la pubblicazione e consultazione di eventi, l'amministrazione delle spese comuni e la visualizzazione della mappa della struttura.

Dal punto di vista tecnologico, OikoNaos è una piattaforma web-based accessibile tramite browser. Tutte le informazioni vengono archiviate in un database relazionale centralizzato, che assicura consistenza e integrità dei dati. Il sistema adotta criteri di sicurezza basati su autenticazione tramite credenziali e gestione dei ruoli, garantendo che le operazioni amministrative siano riservate agli utenti autorizzati.

Lo scopo del sistema si estende quindi alla completa digitalizzazione della gestione comunitaria, fornendo un unico punto di accesso alle funzionalità operative, informative e amministrative della co-housing, in modo da migliorare l'efficienza e la trasparenza del coordinamento tra utenti e azienda.

2.3 Objectives and success criteria of the project

Gli obiettivi fondamentali del progetto OikoNaos sono la semplificazione delle attività gestionali e la promozione di un modello abitativo partecipativo basato su trasparenza, collaborazione e responsabilità condivisa. Il sistema mira a ridurre i tempi di comunicazione tra residenti e amministrazione, a rendere più agevole la prenotazione di spazi e la gestione delle segnalazioni, e a favorire una ripartizione chiara e verificabile delle spese comuni.

Il successo del progetto sarà determinato dalla capacità del sistema di garantire una fruizione fluida e intuitiva, un'elevata affidabilità operativa e la corretta corrispondenza tra le funzionalità implementate e i requisiti descritti in questo documento. Ulteriori criteri di successo includono la stabilità del servizio, la consistenza dei dati memorizzati, la scalabilità del sistema rispetto al numero di utenti e la compatibilità con i principali browser.

OikoNaos sarà ritenuto pienamente riuscito qualora, al termine della fase di validazione, tutte le funzionalità previste risultino operative, i requisiti non funzionali soddisfatti e il sistema risulti utilizzabile da utenti con competenze informatiche di base senza necessità di supporto esterno.

2.4 Definitions, acronyms and abbreviations

Nel contesto del presente documento si adottano le seguenti definizioni:

Co-housing: modello abitativo in cui più persone condividono spazi comuni pur mantenendo alloggi privati autonomi.

Azienda Coordinatrice: ente che gestisce la struttura residenziale, coordina i servizi e supervisiona le attività amministrative e manutentive.

Residente/Coinquilino: utente finale della piattaforma, abilitato all'uso dei servizi di prenotazione, assistenza e gestione spese.

Ticket: richiesta di intervento o segnalazione di guasto generata dal sistema e tracciata nel tempo.

Bacheca Eventi: sezione informativa che consente la pubblicazione di eventi, comunicazioni e annunci da parte dell'amministrazione.

2.5 Overview

La sezione introduttiva del documento fornisce il contesto generale del progetto e ne chiarisce gli obiettivi, la struttura e la terminologia adottata. Le sezioni successive approfondiscono gli aspetti specifici dell'analisi dei requisiti. In particolare, la Sezione 2 descrive lo stato attuale dei processi di gestione della co-housing, evidenziando le problematiche e le limitazioni del sistema pre-esistente o delle procedure manuali; la Sezione 3 illustra in dettaglio il sistema proposto, con la descrizione dei requisiti funzionali e non funzionali e con la formalizzazione dei modelli di sistema; infine, la Sezione 5 raccoglie il glossario dei termini

principali per assicurare uniformità di linguaggio e chiarezza nella comunicazione tra tutte le parti coinvolte.

3. Current System

Attualmente, la gestione delle comunità di co-housing avviene in modo frammentato e scarsamente digitalizzato. Le attività quotidiane, come la prenotazione degli spazi comuni, la segnalazione dei guasti o la condivisione di informazioni tra residenti e amministrazione, vengono spesso svolte attraverso strumenti eterogenei, quali comunicazioni via email, messaggi informali o moduli cartacei. Questo approccio, seppur sufficiente in contesti di piccole dimensioni, si rivela inefficiente e difficilmente scalabile in realtà più complesse, dove il numero di utenti, risorse e attività da gestire cresce rapidamente.

La mancanza di un sistema centralizzato genera una serie di problematiche operative. In primo luogo, la gestione manuale delle prenotazioni degli spazi comuni comporta frequenti sovrapposizioni, ritardi e disguidi, poiché non esiste un meccanismo automatizzato che impedisca la duplicazione delle richieste o notifichi in tempo reale la disponibilità delle risorse. Allo stesso modo, la segnalazione di guasti o richieste di manutenzione risulta inefficiente: i residenti comunicano le problematiche direttamente al coordinatore tramite email o verbalmente, rendendo difficile il tracciamento delle richieste e la loro priorità di risoluzione. Dal punto di vista amministrativo, l'assenza di un'infrastruttura informatica integrata complica la gestione delle spese comuni e delle quote periodiche, che vengono spesso annotate manualmente o tramite fogli di calcolo non condivisi. Ciò limita la trasparenza e aumenta il rischio di errori o discrepanze nei conteggi.

La comunicazione interna alla comunità è anch'essa frammentata: non esiste una bacheca digitale o un canale ufficiale per la diffusione di annunci, eventi o aggiornamenti. Di conseguenza, la partecipazione dei residenti alle attività comuni è discontinua e dipende fortemente dalle iniziative individuali o dalla disponibilità del personale amministrativo.

In generale, l'attuale sistema di gestione delle co-housing risulta quindi disorganico, con processi basati su interazioni isolate e prive di un supporto tecnologico strutturato. Questa situazione non solo riduce l'efficienza operativa, ma ostacola la costruzione di una comunità realmente collaborativa, in cui la condivisione e la partecipazione siano favorite da strumenti digitali adeguati.

Il progetto OikoNaos nasce proprio per rispondere a tali criticità, con l'obiettivo di integrare tutte le funzioni di gestione, comunicazione e supervisione in un'unica piattaforma coerente e accessibile.

4. Proposed System

4.1 Overview

Il sistema OikoNaos è concepito come una piattaforma web integrata per la gestione completa delle comunità di co-housing. Il suo obiettivo è centralizzare in un'unica interfaccia tutte le attività legate alla vita comunitaria, amministrative, operative e comunicative; eliminando la frammentazione degli strumenti attualmente utilizzati e migliorando l'efficienza gestionale dell'Azienda Coordinatrice. L'architettura di OikoNaos è basata su un modello multilivello che separa logicamente la gestione dei dati, la logica applicativa e l'interfaccia utente, garantendo così modularità, scalabilità e sicurezza, il sistema mira ad assicurare un'esperienza uniforme e coerente sia per gli utenti finali sia per gli amministratori.

Dal punto di vista funzionale, OikoNaos consente ai residenti di autenticarsi mediante credenziali personali e di accedere a un profilo personalizzato. All'interno della piattaforma, gli utenti possono prenotare spazi comuni, segnalare guasti tramite la creazione di ticket, visualizzare eventi e annunci pubblicati sulla bacheca digitale, e gestire le proprie spese o contributi trimestrali. Tutte le operazioni vengono tracciate e archiviate nel database, assicurando trasparenza e la possibilità di consultare lo storico delle operazioni.

Parallelamente, il sistema fornisce agli amministratori e all'Azienda Coordinatrice strumenti dedicati per la supervisione delle attività. Questi includono la possibilità di monitorare lo stato dei ticket di manutenzione, gestire la disponibilità delle risorse comuni, approvare o modificare eventi, aggiornare il calendario delle

prenotazioni e verificare l'andamento delle spese della comunità.

OikoNaos si configura dunque come un sistema a supporto della collaborazione e della coesione sociale, in cui la tecnologia diventa strumento di coordinamento e responsabilizzazione. La piattaforma favorisce una gestione partecipata, basata su regole condivise e su un'interfaccia intuitiva che consente anche a utenti privi di competenze tecniche avanzate di operare in modo autonomo e sicuro.

4.2 Functional Requirements

Autenticazione

- **RF01. Registrazione:** Il sistema deve consentire la registrazione di un nuovo coinquilino non registrato.
- **RF02. Login:** Il sistema deve consentire all'utente registrato di accedere alla piattaforma inserendo le proprie credenziali.
- **RF03. Logout:** Il sistema deve permettere all'utente autenticato di effettuare logout.

Gestione Profilo

- **RF04. Recupero password:** Il sistema deve consentire agli utenti registrati di recuperare la propria password.
- **RF05. Visualizzazione profilo:** Il sistema deve permettere all'utente autenticato di visualizzare i propri dati personali.
- **RF06. Modifica profilo:** Il sistema deve permettere all'utente autenticato di modificare i propri dati personali.
- **RF07. Modifica password:** Il sistema dovrà fornire all'utente autenticato la possibilità di modificare la propria password.

Calendario Prenotazioni

- **RF08. Creazione prenotazione:** Il sistema deve consentire all'utente autenticato di creare una nuova prenotazione.
- **RF09. Cancellazione prenotazione:** Il sistema deve consentire all'utente autenticato di cancellare una prenotazione precedentemente creata.

Ticket di Manutenzione

- **RF10. Apertura ticket:** Il sistema deve permettere agli utenti di segnalare un guasto o una richiesta di manutenzione.
- **RF11. Cancellazione ticket:** Il sistema deve consentire all'utente autenticato di annullare un ticket creato.
- **RF12. Visualizzazione cronologia ticket:** Il sistema deve fornire all'utente autenticato un elenco dei ticket da lui creati.

Bacheca eventi e annunci

- **RF13. Visualizzazione della bacheca:** Il sistema deve consentire a tutti gli utenti autenticati di visualizzare la bachecca degli eventi e degli annunci.
- **RF14. Iscrizione agli eventi:** Il sistema deve permettere all'utente autenticato di iscriversi a un evento disponibile.
- **RF15. Visualizzazione mappa:** Il sistema deve consentire agli utenti autenticati di visualizzare una mappa della struttura.

Gestione risorse condivise

- **RF16. Prenotazione risorsa:** Il sistema deve permettere all'utente autenticato di prenotare una risorsa condivisa accettandone le regole d'uso.
- **RF17. Visualizzazione cronologia prenotazione risorse:** Il sistema deve consentire all'utente di consultare lo storico delle risorse prenotate.

Gestione Spese

- **RF18. Pagamento tassa trimestrale:** Il sistema deve consentire all'utente autenticato di effettuare il

- pagamento della tassa trimestrale.
- **RF19. Visualizzazione tasse:** Il sistema deve consentire all’utente autenticato di visualizzare lo stato delle tasse.

4.3 Nonfunctional Requirements

4.3.1 Usability

- **RNF01. Facilità d’uso:** Il sistema deve consentire a un nuovo utente di completare la registrazione e accedere alle funzionalità principali. (Messaggi per guidare l’utente durante le interazioni)
- **RNF02. Coerenza dell’interfaccia:** Tutte le schermate dell’applicazione devono utilizzare stili grafici, colori e terminologia coerenti per garantire un’esperienza d’uso uniforme.

4.3.2 Reliability

- **RNF03. Disponibilità del servizio:** Il servizio deve essere accessibile 24 ore su 24, 7 giorni su 7, salvo periodi di manutenzione pianificata.
- **RNF04. Recupero in caso di guasto:** In caso di interruzione imprevista, il sistema deve essere in grado di ripristinare lo stato precedente, recuperando lo stato precedente entro 30 secondi dal riavvio, senza perdita di dati utente.
- **RNF05. Gestione degli errori:** Il sistema deve gestire gli input non validi senza interrompere l’esecuzione e mostrando un messaggio d’errore.

4.3.3 Performance

- **RNF06. Tempo di risposta:** Il tempo medio di risposta a una richiesta utente non deve superare i 2 secondi in condizioni di carico normale.
- **RNF07. Carico massimo supportato:** Il sistema deve poter gestire fino a 500 utenti simultanei senza degrado significativo delle prestazioni.
- **RNF08. Accuratezza dei dati:** I risultati delle operazioni di elaborazione devono garantire dati precisi e privi di errori.

4.3.4 Supportability

- **RNF09. Documentazione del codice:** Il codice sorgente deve essere commentato e documentato. Ogni funzione deve avere almeno un commento descrittivo e un’intestazione con autore e data di modifica.
- **RNF10. Configurabilità:** Le principali impostazioni del sistema (es. parametri di connessione, lingua, limiti di sessione) devono poter essere modificate senza intervento sul codice sorgente.
- **RNF11. Internazionalizzazione:** Il sistema deve supportare la traduzione dei testi in almeno due lingue tramite file di risorse esterni. (*Media Priorità*)

4.3.5 Implementation

- **RNF12. Tecnologie:** L’implementazione deve avvenire utilizzando le tecnologie specifiche stabilite: Java 17 per la logica di business, **Tomcat 10.1 (Jakarta EE)** come Application Server e MySQL 8 come Database Management System (DBMS).
- **RNF13 Strumenti di Sviluppo:** L’ambiente di sviluppo primario deve essere **IntelliJ 2025.2**, e la gestione delle versioni del codice deve avvenire tramite **Git**.
- **RNF14 Modellazione:** L’implementazione deve essere coerente con i modelli **UML (Object Model, Dynamic Model)** definiti nella fase di analisi.
- **RNF15. Compatibilità:** L’applicazione deve poter essere eseguita su diversi ambienti server e browser (almeno Chrome, Firefox, Edge, Safari) senza modifiche al codice.

4.3.6 Interface

- **RNF16 Standard Interfaccia Utente (UI):** L'interfaccia utente (web) deve aderire a un design **responsive**, garantendo la visualizzazione e l'uso ottimale su tutti i tipi di dispositivi.
- **RNF17 Notifiche:** Il sistema deve supportare l'invio di notifiche via **email** agli utenti in caso di eventi critici (es. conferma di registrazione, recupero password).
- **RNF18 Formato dei Dati:** I dati scambiati tra il client (browser) e il server devono essere formattati in **JSON** o **XML** per l'interoperabilità, sebbene l'applicazione non debba connettersi a sistemi esterni.

4.3.7 Packaging

- **RNF19 Distribuzione:** Il sistema deve essere impacchettato come file **.WAR** (Web Application Archive) per la distribuzione sull'Application Server Tomcat.
- **RNF20 Requisiti di Installazione:** L'installazione deve prevedere un set minimo di istruzioni per la configurazione del DBMS (MySQL) e la copia del file **.WAR** nella directory di *deployment* di Tomcat.
- **RNF21 Dipendenze:** Il pacchetto distribuibile deve includere tutte le librerie (JAR) necessarie per l'esecuzione, ad eccezione di quelle fornite nativamente dall'ambiente **Java/Jakarta EE** e da Tomcat.

4.3.8 Security

- **RNF22 Crittografia Credenziali:** Le password degli utenti devono essere archiviate utilizzando algoritmi di **hashing** unidirezionale e sicuro (es. bcrypt o SHA-256 con salt) per impedire la lettura diretta in caso di accesso non autorizzato al database.
- **RNF23 Sicurezza della Trasmissione:** Tutte le comunicazioni tra il browser dell'utente e il server (inclusi login e transazioni di pagamento) devono essere cifrate tramite protocollo **HTTPS/TLS**.
- **RNF24 Controllo Accessi:** Il sistema deve implementare un meccanismo di controllo degli accessi basato sui ruoli (*Role-Based Access Control - RBAC*) per garantire che le operazioni amministrative (es. gestione regole, modifica utenze) siano riservate agli utenti con ruolo di Amministrazione/Supervisore.

4.3.9 Legal

- **RNF25 Protezione Dati Personalni:** Il sistema deve essere conforme alle normative sulla protezione dei dati personali (**GDPR** o equivalenti), richiedendo il **consenso esplicito** dell'utente per il trattamento dei dati al momento della registrazione.

4.4 System Models

4.4.1 Scenarios

Autenticazione

Il coinquilino Matteo vuole effettuare il login sulla piattaforma web per poter visionare il proprio profilo personale. Matteo apre il browser e inserisce nella pagina di login la propria e-mail matteo.rossi@gmail.com, la password associata al suo account e il codice dato dall'Azienda Coordinatrice per identificare i residenti della comunità. Matteo, quindi, clicca sul pulsante “Accedi”, il sistema verifica le credenziali e, se corrette, lo reindirizza alla home page personale. In caso di credenziali errate, il sistema mostra un messaggio di errore che invita a riprovare. L'utente ha inoltre la possibilità di recuperare la password tramite un link temporaneo inviato all'email registrata, qualora non riuscisse ad accedere.

Prenotazione spazio comune

Il coinquilino Luca vuole prenotare una postazione della sala studio. Le postazioni sono in totale 10 con tre fasce orarie disponibili per ciascuna: 8:00-12:00, 12:00-15:00, 15:00-18:00. Il coinquilino effettua il login nel sistema inserendo la propria e-mail: luca.g@gmail.com e la sua password 3050ciao!) accedendo così alla home page. Dalla barra di navigazione seleziona il calendario. Luca successivamente seleziona la sala studio, la postazione

2, il giorno della prenotazione (11/10/2025) e la fascia oraria desiderata (12:00-15:00) per poi confermare la sua scelta. Il sistema ora presenterà tale postazione nell'orario e nel giorno specifico di colore rosso dunque non selezionabile.

Cancellazione prenotazione sala studio

Il coinquilino Luca decide di non utilizzare più la sala studio che aveva prenotato per il giorno 18/10/2025 dalle 12:00 alle 15:00. Dopo aver effettuato l'accesso alla piattaforma, dall'area personale, seleziona "Calendario prenotazioni" e la prenotazione da annullare. Clicca sull'opzione "Cancella prenotazione" e conferma l'operazione. Il sistema chiede una breve conferma per evitare cancellazioni accidentali e, dopo l'approvazione, rimuove la prenotazione dal calendario. A questo punto, la postazione precedentemente occupata torna disponibile e dal calendario viene rimossa l'icona relativa alla specifica prenotazione.

Segnalazione guasto (ticket)

La coinquilina Sara ha avuto problemi relativi al guasto della propria tv. Il sistema permette di segnalare disagi attraverso la creazione di ticket per manutenzioni esterne: di essi occorre specificare categoria, descrizione, foto. Sara effettua il log-in nel sistema inserendo la propria e-mail: sara09@gmail.com e la sua password 8960sara!! . Dall'area personale Sara seleziona la voce "Ticket" poi la categoria del guasto ("Tv"), una breve descrizione del problema, la priorità e opzionalmente una foto per poi confermare la sua scelta. Il sistema ora elaborerà la richiesta inoltrandola al supervisore specifico. Il sistema conferma poi la creazione del ticket e assegna ad esso un codice univoco, l'utente può monitorare lo stato di elaborazione nella propria pagina personale nella sezione "Cronologia ticket" in cui ci saranno aggiornamenti automatici. L'utente può inoltre scegliere di cancellare il ticket, purchè esso sia ancora nello stato APERTO, se si sono verificati errori nella compilazione della richiesta o se la segnalazione non è più necessaria.

Cancellazione ticket di segnalazione

Il coinquilino Paolo, dopo aver segnalato un guasto al frigorifero della cucina tramite la piattaforma, si accorge che il problema è stato risolto. Decide quindi di annullare il ticket appena inviato. Dopo aver effettuato il login con le proprie credenziali, Matteo accede alla sezione "Ticket" dalla propria area personale, dove è presente l'elenco delle richieste. Seleziona il ticket relativo al guasto del frigorifero e clicca sull'opzione "Cancella ticket". Una volta completata l'operazione, il sistema mostra un messaggio di conferma: "Il ticket è stato cancellato con successo", e la richiesta scompare dall'elenco dei ticket attivi. Se invece il ticket fosse già stato preso in carico, il sistema mostrerebbe un messaggio informativo: "Impossibile annullare il ticket: la richiesta è già in lavorazione".

4.5 Use case model

UC01. Registrazione Nuovo Coinquilino

Questo caso d'uso copre il requisito **RF01**.

Attore: Utente (Coinquilino non registrato).

Entry Condition: Il potenziale coinquilino si trova sulla home page (Img1) e seleziona l'opzione "Registrati", possedendo il Codice Identificativo Speciale fornito dall'Azienda Coordinatrice.

Flusso di Eventi:

1. L'utente dalla home page (Img1) accede alla schermata di registrazione tramite l'opzione "Registrati".
2. L'utente si trova sulla schermata di registrazione.
3. L'utente inserisce i dati personali (foto, nome, e-mail, password, contatti) e il **Codice Identificativo Speciale**.
4. L'utente conferma l'invio dei dati.
5. Il sistema verifica la validità del Codice Identificativo Speciale e il formato dei dati inseriti.
6. Il sistema salva le credenziali cifrate e crea il profilo.
7. Il sistema invia una notifica di avvenuta registrazione (es. via e-mail).

Exit Condition: Il nuovo coinquilino è registrato nel database e si trova all'interno della propria area personale.

Flusso alternativi

- **Codice Invalido:** Se il Codice Identificativo Speciale risulta errato al punto 5, il sistema mostra un messaggio di errore e riporta l'utente sulla schermata di registrazione.

UC02. Autenticazione

Attore: Utente (Coinquilino);

Questo caso d'uso copre il requisito **RF02**.

Entry Condition: Il coinquilino si trova nella home page (Img1) del sito nella schermata e seleziona la voce “Accedi”.

Flusso di eventi:

1. Il coinquilino si trova nella schermata di login (Img2)
2. Il coinquilino inserisce username e password
3. Il coinquilino invia i dati al sistema
4. Il sistema effettua un controllo sulle credenziali
5. Il sistema reindirizza l'utente alla sua area personale
6. L'utente si trova nell'area personale (Img3)

Exit Condition: Il coinquilino autenticato e si trova nell'area personale.

Flusso alternativo: Se al punto 3 il controllo ha un riscontro negativo allora il sistema dovrà notificare errore di autenticazione con “username o password errate” e ripristinerà la schermata di login (UC03 Autenticazione fallita).

UC03. Autenticazione fallita

Attore: Utente (Coinquilino);

Questo caso d'uso copre il requisito **RF02**.

Entry Condition: Il coinquilino tenta di effettuare il login con credenziali errate.

Flusso di eventi:

1. Il sistema mostra il messaggio di errore “username o password errate”.
2. Il sistema reindirizza l'utente sulla schermata di login.
3. L'utente si trova nella schermata di login

Exit Condition: L'utente si trova nella pagina di autenticazione del sito.

UC04. Logout

Questo caso d'uso copre il requisito **RF03**.

Attore: Utente (Coinquilino)

Entry Condition: Il coinquilino è autenticato e si trova nella propria area personale (Img3).

Flusso di eventi:

1. Il coinquilino visualizza la propria area personale (Img3).
2. Il coinquilino clicca sull'icona utente per la gestione del profilo.
3. Il coinquilino seleziona la voce “Logout”.
4. Il sistema invalida la sessione utente.
5. Il sistema reindirizza l'utente alla home page (Img1)
6. L'utente si trova nella home page (Img1)

Exit Condition: Il coinquilino non è più autenticato e si trova nella home page del sito.

Flussi alternativi:

- Se durante la disconnessione si verifica un errore di sistema, il sistema notifica all'utente che l'operazione non è andata a buon fine e lo invita a riprovare.

UC05. Modifica Profilo Utente

Questo caso d'uso copre il requisito **RF05 e RF06**.

Attore: Utente (Coinquilino).

Entry Condition: L'utente è autenticato e si trova nella propria area personale (Img3).

Flusso di Eventi:

1. L'utente seleziona l'icona utente.
2. L'utente si trova in “Gestione profilo” in cui visualizza i dati attuali

3. L'utente seleziona l'opzione di modifica (es. "Modifica Dati").
4. L'utente aggiorna uno o più campi (es. modifica la foto profilo).
5. L'utente clicca "Salva Modifiche".
6. Il sistema convalida i nuovi dati (es. verifica la validità della nuova e-mail).
7. Il sistema aggiorna il database e mostra un messaggio di conferma.

Exit Condition: I dati personali dell'utente sono stati aggiornati correttamente nel sistema.

Flussi Alternativi:

- **Formato non valido:** quando il sistema effettua un controllo sui dati (al punto 6), rileva che i dati inseriti dall'utente non rispettano i formati richiesti e mostra un messaggio di errore con reindirizzamento alla pagina di "Gestione profilo".

UC06. Modifica Password

Questo caso d'uso copre il requisito **RF07**.

Attore: Utente (Coinquilino)

Entry Condition: L'utente è autenticato e si trova nella propria area personale (*Img3*), nella sezione "Gestione Profilo".

Flusso di Eventi:

1. L'utente seleziona l'opzione "Modifica Password" all'interno della sezione "Gestione Profilo".
2. Il sistema mostra il form per la modifica della password.
3. Nel form l'utente inserisce la password attuale e la nuova password e conferma la nuova password ripetendola nel campo dedicato.
4. L'utente clicca su "Conferma".
5. Il sistema verifica:
 - che la password attuale sia corretta
 - che la nuova password rispetti i requisiti minimi
 - che la conferma corrisponda alla nuova password
6. Il sistema aggiorna la password nel database.
7. Il sistema mostra un messaggio di conferma (es. "Password aggiornata con successo").

Exit Condition: La nuova password è stata memorizzata e sostituisce quella precedente, l'utente viene reindirizzato sulla pagina "Gestione profilo".

UC07. Prenotazione postazione sala studio

Questo caso d'uso copre il requisito **RF08**.

Attore: Utente (Coinquilino);

Entry Condition: L'utente è autenticato e si trova nell'area personale (*Img3*).

Flusso di eventi:

1. L'utente seleziona la voce Calendario.
2. L'utente si trova nel Calendario.
3. L'utente seleziona la voce "Nuova prenotazione"
4. Il sistema genera un form con le sezioni: ambiente, postazione, data e fascia oraria (*Img4*).
5. L'utente completa i campi e clicca "Prenota".
6. Il sistema verifica la disponibilità.
7. Se tale selezione è disponibile, l'utente conferma la prenotazione (*Img5*).
8. Il sistema aggiorna il calendario.

Exit Condition: L'utente ha prenotato la postazione e il sistema mostra nel calendario la prenotazione nella data e ora corrispondente.

Flussi alternativi: Se nel punto 6 il sistema mostra che tale selezione non è disponibile, verrà visualizzato un messaggio di errore (*Img6*) e il sistema reindirizzerà l'utente al form di "Nuova Prenotazione".

UC08. Cancellazione Prenotazione

Questo caso d'uso copre il requisito **RF09**.

Attore: Utente (Coinquilino)

Entry Condition: L'utente è autenticato e visualizza una o più prenotazioni attive nel calendario.

Flusso di Eventi Principale:

1. L'utente accede alla sezione Calendario dall'area personale.
2. L'utente seleziona la prenotazione che desidera cancellare.
3. L'utente clicca sull'icona menù, visualizza e sceglie l'opzione “Cancella prenotazione”.
4. Il sistema mostra una finestra di conferma per evitare cancellazioni accidentali.
5. L'utente conferma l'operazione di cancellazione.
6. Il sistema rimuove la prenotazione dal calendario e aggiorna il database.
7. Il sistema mostra un messaggio di conferma e rende nuovamente disponibile la postazione nella data e fascia oraria corrispondente.

Exit Condition: La prenotazione risulta cancellata e non compare più nel calendario.

Flussi Alternativi: L'utente annulla la conferma (cancellazione non completata). Al punto 5, l'utente seleziona “Annulla” invece di confermare; dunque, il sistema interrompe l'operazione e riporta l'utente al calendario senza modificare la prenotazione.

UC09. Creazione ticket

Questo caso d'uso copre il requisito **RF10**.

Attore: Utente (Coinquilino);

Entry Condition: L'utente ha effettuato l'accesso e si trova nella propria area personale.

Flusso di eventi:

1. L'utente si è autenticato e si trova nella propria area personale (*Img3*);
2. L'utente seleziona l'icona “Ticket”;
3. L'utente clicca su “Nuovo Ticket”;
4. Il sistema mostra la schermata di creazione di ticket con i campi da compilare;
5. L'utente inserisce nei rispettivi campi categoria, descrizione, priorità e foto relative alla segnalazione;
6. L'utente conferma e invia il ticket;
7. Il sistema controlla la correttezza e completezza dei campi inseriti dall'utente;
8. Il sistema assegna un codice al ticket;
9. Il sistema invia il ticket al supervisore;
10. Il sistema aggiunge alla cronologia dei ticket (*Img13*) quello appena creato con i rispettivi dettagli

Exit Condition: Il ticket dell'utente è stato creato correttamente, tracciato dal sistema e inserito nella cronologia;

Flussi Alternativi:

Flusso 1 - Dati inseriti non validi o incompleti: al punto 7, dopo dell'invio, se uno o più campi obbligatori risultano mancanti o presentano un formato non valido (es. descrizione troppo lunga o troppo breve, categoria non selezionata, foto non accettata):

1. Il sistema evidenzia i campi da correggere e mostra un messaggio di errore
2. L'utente può modificare i dati inseriti.
3. L'utente clicca nuovamente su “Invia” per confermare l'invio del ticket.

Flusso 2 - L'utente annulla la creazione del ticket: In qualunque momento durante la compilazione, l'utente può selezionare “Annulla”. Il sistema richiede una conferma per evitare l'annullamento accidentale e dopo la conferma, il sistema non salva i dati inseriti e riporta l'utente alla schermata precedente.

UC10. Cancellazione Ticket

Questo caso d'uso copre il requisito **RF11**.

Attore: Utente (Coinquilino);

Entry Condition: L'utente dopo aver confermato l'invio del ticket decide di annullarlo.

Flusso di eventi:

1. L'utente dopo aver effettuato l'autenticazione accede alla sezione “Ticket” nella propria area personale.
2. L'utente seleziona il ticket che intende cancellare.
3. Il sistema renderizza l'utente ad una pagina che mostra i dettagli (descrizione, data, storico degli aggiornamenti, stato, priorità) del ticket.
4. L'utente clicca sulla voce “Cancella ticket”.
5. Il sistema apre una finestra di conferma
6. L'utente conferma la sua scelta.
7. Il sistema rimuove il ticket dalla lista.
8. Il sistema avvisa il supervisore incaricato della cancellazione.

9. L'utente viene reindirizzato alla pagina "Ticket"

Exit Condition: Il ticket è stato annullato e non è più visibile dall'utente.

Flussi alternativi:

Flusso 1 – L'utente annulla la conferma di cancellazione: al punto 6 l'utente seleziona "Annulla" nella finestra di conferma, il sistema chiude la finestra senza rimuovere il ticket e l'utente torna alla schermata alla lista dei ticket.

Flusso 2 – Il ticket non è più cancellabile (es. il ticket è già stato preso in carico da un supervisore): al punto 4, quando l'utente tenta di cancellare il ticket che non è nello stato "Aperto", il sistema rileva che il ticket ha lo stato "In lavorazione" o "Completato" e mostra un messaggio: "Non è più possibile cancellare questo ticket.". L'utente viene dunque riportato alla schermata "Ticket" senza apportare modifiche.

UC11. Visualizzazione Cronologia Ticket

Questo caso d'uso copre il requisito **RF12**.

Attore: Utente (Coinquilino) o Supervisore (Azienda Coordinatrice).

Entry Condition: L'utente è autenticato e si trova nella Home Page (*Img3*).

Flusso di Eventi:

1. L'utente seleziona la sezione "Ticket";
2. Il sistema mostra l'elenco dei ticket creati dall'utente, ordinati per data o stato predefinito (*Img12*).
3. L'utente utilizza i filtri per **stato** (Aperto, In Lavorazione, Chiuso, Annullato) o per periodo di apertura o priorità.
4. Il sistema aggiorna la lista in base ai filtri selezionati.
5. L'utente può selezionare un ticket per visualizzarne il dettaglio completo (*Img13*) (descrizione, priorità, storico degli aggiornamenti, stato attuale).

Exit Condition: L'utente ha consultato in modo efficiente lo stato e la cronologia delle proprie richieste di assistenza.

Flussi Alternativi:

- **Supervisore:** Se l'attore è il Supervisore, al punto 2, la lista include tutti i ticket della co-housing (non solo quelli creati dal singolo utente).

UC12. Iscrizione a Evento

Questo caso d'uso copre il requisito **RF14**.

Attore: Utente (Coinquilino).

Entry Condition: L'utente è autenticato e si trova nella sezione "Bacheca" per iscriversi ad uno o più eventi.

Flusso di Eventi:

1. L'utente accede alla sezione "Bacheca".
2. L'utente seleziona un evento a cui desidera partecipare.
3. L'utente clicca sul pulsante "Iscriviti".
4. Il sistema verifica la disponibilità di posti e, se presenti, registra la partecipazione dell'utente nel database.
5. Il sistema riduce il numero dei posti disponibili per quell'evento.
6. Il sistema mostra la schermata di conferma di iscrizione.
7. Il sistema aggiorna il calendario con l'evento a cui l'utente si è iscritto.
8. L'utente viene reindirizzato alla bacheca.

Exit Condition: L'utente risulta iscritto all'evento e il numero dei posti disponibili viene aggiornato.

Flussi Alternativi:

- **Posti Esauriti:** Se i posti disponibili sono esauriti (punto 4), il sistema mostra un messaggio di errore e non consente l'iscrizione.

UC13. Richiesta Risorsa Condivisa con Accettazione Regole

Questo caso d'uso copre i requisiti **RF16**.

Attore: Utente (Coinquilino).

Entry Condition: L'utente è autenticato e si trova nella sezione di gestione delle risorse condivise.

Flusso di Eventi:

1. L'utente accede alla sezione "Risorse"

2. L'utente seleziona una data dal calendario.
3. Il sistema mostra l'elenco delle risorse disponibili per quel giorno.
4. L'utente seleziona la risorsa desiderata.
5. Prima della conferma, il sistema mostra le Regole d'Uso specifiche per quella risorsa.
6. L'utente deve selezionare esplicitamente una casella di spunta per accettare le regole d'uso.
7. L'utente clicca "Richiedi Risorsa".
8. Il sistema registra la prenotazione giornaliera nel database.
9. Il sistema imposta la risorsa come non disponibile per quella.
10. Il sistema mostra all'utente un messaggio di conferma richiesta.

Exit Condition: La richiesta per la risorsa è registrata nel database per l'intera giornata, e l'utente ha accettato le regole d'uso, inclusa la penale in caso di infrazione.

Flussi Alternativi:

Flusso 1 – Mancata Accettazione delle Regole d'Uso: al punto 6 l'utente tenta di cliccare su "Richiedi risorsa" senza spuntare la casella di accettazione delle regole d'uso, allora il sistema mostra un messaggio: "**È necessario accettare le Regole d'Uso per procedere**" e rimane nella stessa pagina.

Flusso 2 – L'utente annulla la richiesta: in qualsiasi fase prima del punto 7, se l'utente seleziona "Annulla" o torna indietro il sistema richiede una conferma per evitare l'interruzione accidentale, e quando l'utente conferma l'annullamento la procedura si interrompe senza registrare nulla.

Flusso 3 – Nessuna risorsa disponibile nella data selezionata: al punto 3 il sistema mostra il messaggio "Nessuna risorsa disponibile in questa data" e l'utente può selezionare una nuova data.

UC14. Pagamento Tassa Trimestrale

Questo caso d'uso copre il requisito **RF18**.

Attore: Utente (Coinquilino).

Entry Condition: L'utente è autenticato e presenta un importo dovuto relativo alla tassa trimestrale.

Flusso di Eventi:

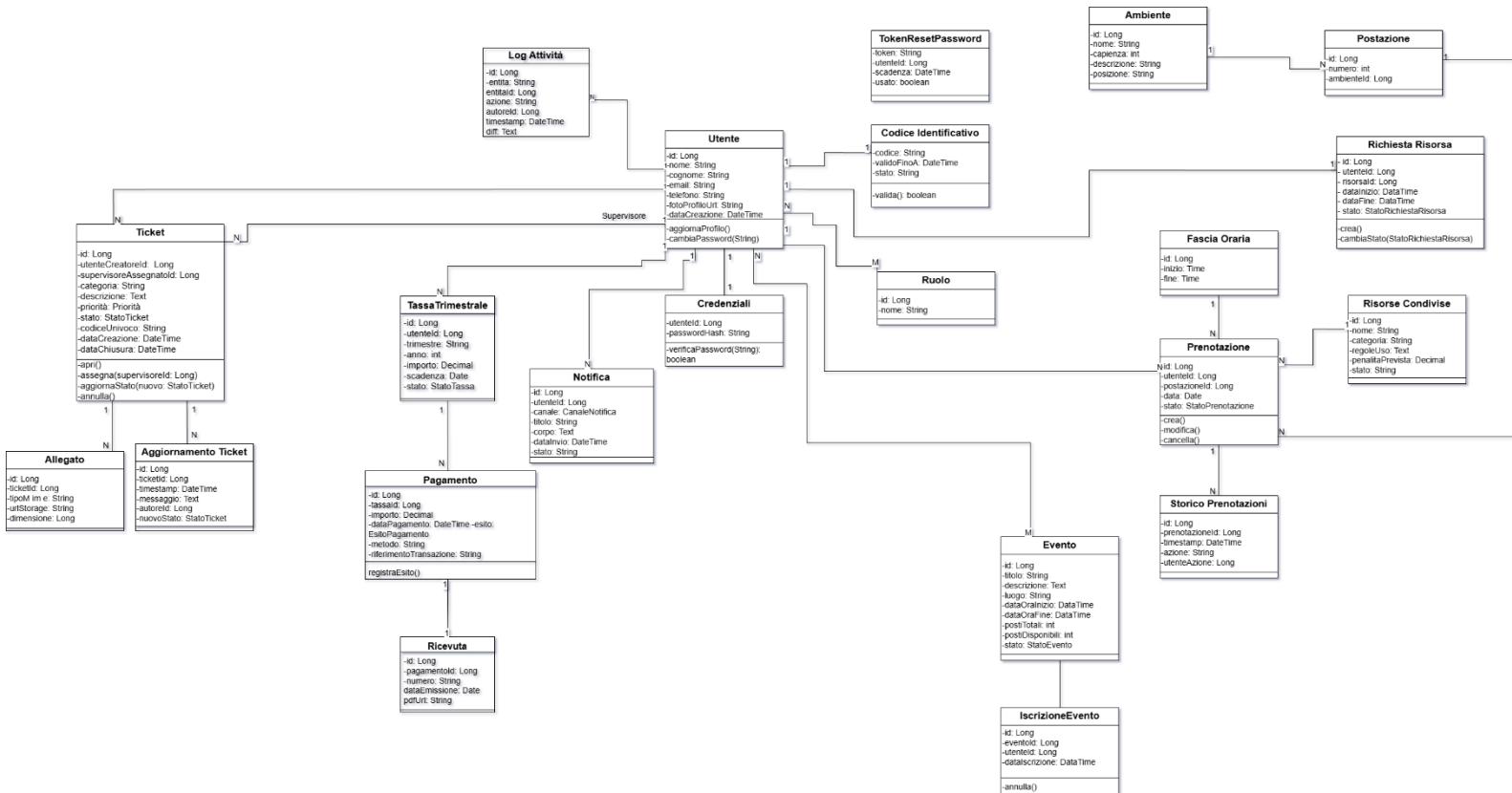
1. L'utente accede alla sezione "Spese" dalla propria area personale (*Img3*).
2. Il sistema mostra la tassa da pagare con relativo importo dovuto, periodo di riferimento e scadenza (*Img11*).
3. L'utente seleziona l'opzione "Paga Ora".
4. Il sistema reindirizza l'utente a una schermata di pagamento integrata (*Img10*).
5. L'utente inserisce i dettagli del metodo di pagamento (es. carta/bonifico) (*Img11*).
6. L'utente conferma la transazione.
7. Il sistema riceve la conferma di successo dalla piattaforma di pagamento (*Img12*).
8. Il sistema aggiorna automaticamente lo stato del pagamento nel database da "Non Pagato" a "Pagato".
9. L'utente viene reindirizzato alla sezione "Spese" e visualizzerà il testo "Pagamento effettuato" in corrispondenza di tale tassa.

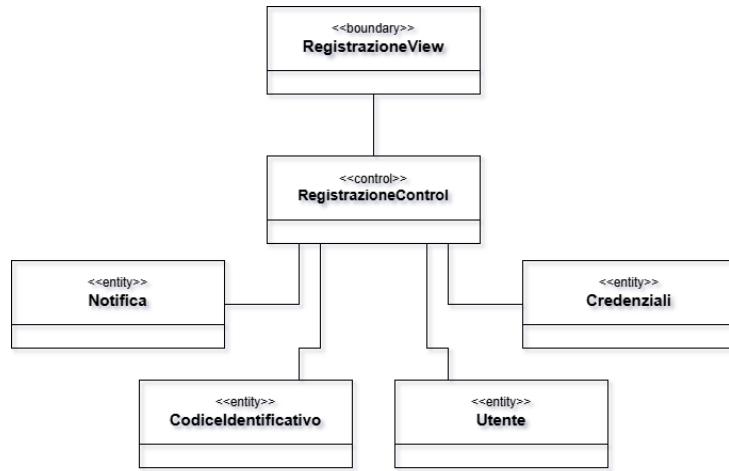
Exit Condition: Il pagamento è stato registrato correttamente e lo stato della tassa trimestrale dell'utente è aggiornato a "Pagato" nel database.

Flussi Alternativi:

- **Pagamento Fallito:** Se la transazione viene rifiutata al punto 7 il sistema mostra all'utente un messaggio di errore "Transazione fallita: si è verificato un errore" (*Img13*) e l'utente può scegliere tra due opzioni: "Riprova" o "Annulla". Il database mantiene lo stato della tassa su "Non Pagato".

4.5.1 Object model

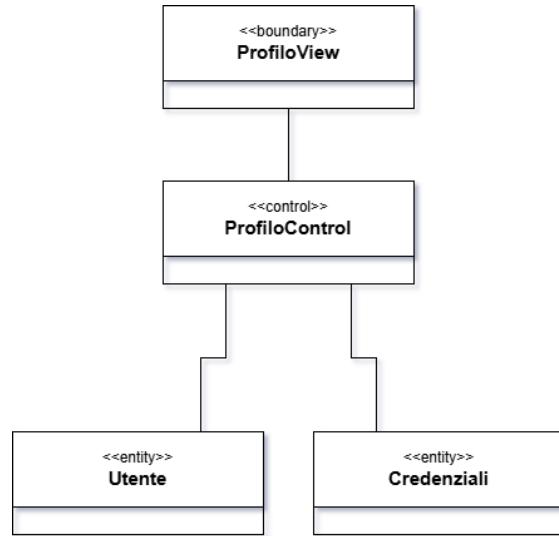




Requisito funzionale coperto: RF01 – *Registrazione*

Caso d'uso associato: UC01 – *Registrazione Nuovo Coinquilino*

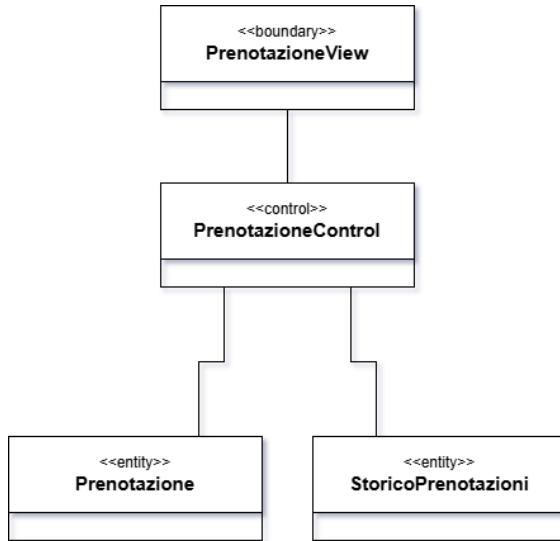
Tipo	Nome Classe	Descrizione
Boundary	RegistrazioneView	Interfaccia grafica che consente all'utente non registrato di inserire i propri dati personali e il codice identificativo speciale fornito dall'Azienda Coordinatrice. Gestisce i messaggi di errore o conferma.
Control	RegistrazioneControl	Coordina il processo di registrazione: riceve i dati dalla vista, ne valida la correttezza, verifica la validità del codice identificativo e crea le entità Utente e Credenziali. Invoca inoltre l'invio di una Notifica di conferma.
Entity	Utente	Contiene le informazioni anagrafiche e di profilo del coinquilino (nome, cognome, e-mail, contatti). È associato a un set di credenziali e a un codice identificativo univoco.
Entity	Credenziali	Rappresenta i dati di autenticazione dell'utente (e-mail e password cifrata). È collegata alla classe Utente.
Entity	CodiceIdentificativo	Contiene il codice speciale fornito dall'amministrazione per verificare che l'utente appartenga a una specifica comunità. Deve essere validato prima della registrazione.
Entity	Notifica	Gestisce l'invio della conferma di registrazione (es. e-mail di benvenuto o notifica interna).



Requisito funzionale: RF05 – Visualizzazione profilo, RF06 – Modifica profilo

Caso d'uso associato: UC05 – Modifica Profilo Utente

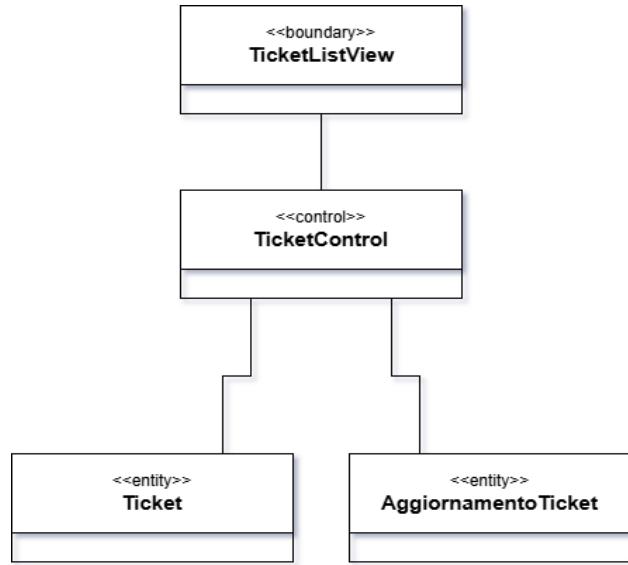
Tipo	Nome Classe	Descrizione
Boundary	ProfiloView	Interfaccia utente che permette la visualizzazione e la modifica dei dati personali e delle credenziali. Fornisce messaggi di conferma o errore dopo le operazioni di aggiornamento.
Control	ProfiloControl	Gestisce la logica di modifica del profilo. Riceve i dati dalla ProfiloView, ne valida la correttezza e aggiorna le informazioni nelle entità Utente e Credenziali. Coordina la persistenza delle modifiche.
Entity	Utente	Contiene i dati anagrafici e di contatto dell'utente (nome, email, foto, ecc.). È l'entità principale legata alle credenziali.
Entity	Credenziali	Rappresenta le informazioni di autenticazione (email, password cifrata). È associata a un solo Utente.



Requisiti funzionali: RF08 – Creazione prenotazione; RF09 – Cancellazione prenotazione

Casi d'uso associati: UC07 – Prenotazione; UC08 – Cancellazione Prenotazione

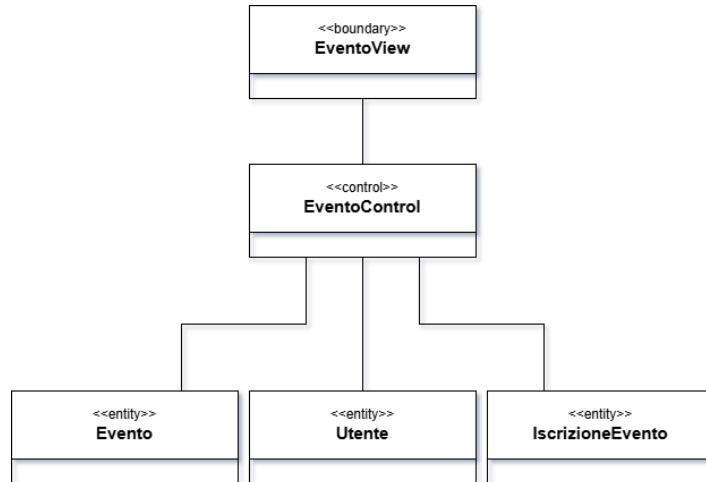
Tipo	Nome Classe	Descrizione
Boundary	PrenotazioneView	Interfaccia grafica che consente all'utente di visualizzare il calendario delle risorse, creare, modificare o cancellare prenotazioni. Mostra messaggi di conferma o errore.
Control	PrenotazioneControl	Gestisce la logica applicativa relativa alle prenotazioni: riceve i dati dalla vista, verifica la disponibilità della risorsa, effettua le modifiche richieste e aggiorna le entità persistenti. Coordina anche la registrazione dello storico delle operazioni.
Entity	Prenotazione	Contiene i dati relativi a una prenotazione (id, utente, risorsa, data, ora, stato). Rappresenta l'istanza attiva della prenotazione.
Entity	StoricoPrenotazioni	Registra tutte le prenotazioni passate e cancellate, consentendo la consultazione dello storico da parte dell'utente. È aggiornata automaticamente da PrenotazioneControl dopo ogni operazione.



Requisiti funzionali: RF10 – Apertura ticket; RF11 – Cancellazione ticket; RF12 – Visualizzazione cronologia ticket

Casi d'uso associati: UC09 – Creazione Ticket; UC10 – Cancellazione Ticket; UC11 – Visualizzazione Cronologia Ticket

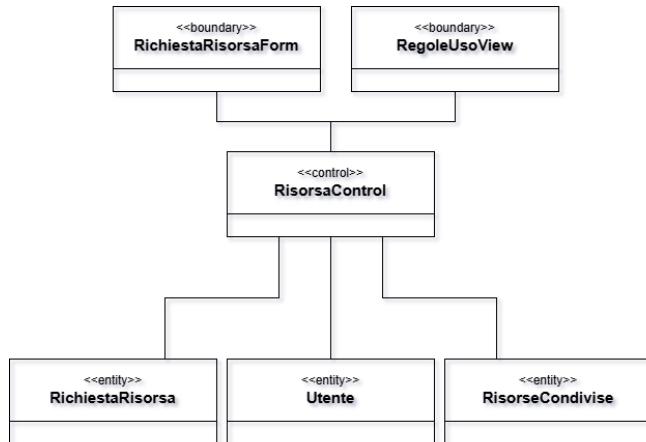
Tipo	Nome Classe	Descrizione
Boundary	TicketListView	Interfaccia grafica che consente all'utente di creare nuovi ticket, visualizzare quelli esistenti, annullarli o consultarne lo stato. Mostra messaggi di conferma o errore.
Control	TicketControl	Gestisce il ciclo di vita dei ticket. Coordina la creazione, l'aggiornamento e la cancellazione dei ticket nel database. Riceve i dati dalla TicketListView, ne valida la correttezza e aggiorna le entità coinvolte.
Entity	Ticket	Rappresenta la richiesta di manutenzione inviata da un utente. Contiene informazioni quali categoria, descrizione, priorità, stato, data di apertura e codice identificativo univoco.
Entity	AggiornamentoTicket	Contiene lo storico degli aggiornamenti di ciascun ticket, come cambi di stato, commenti o notifiche inviate. È collegata a un solo Ticket ma può avere più istanze nel tempo.



Requisito funzionale: RF14 – Iscrizione agli eventi

Caso d'uso associato: UC12 – Iscrizione a Evento

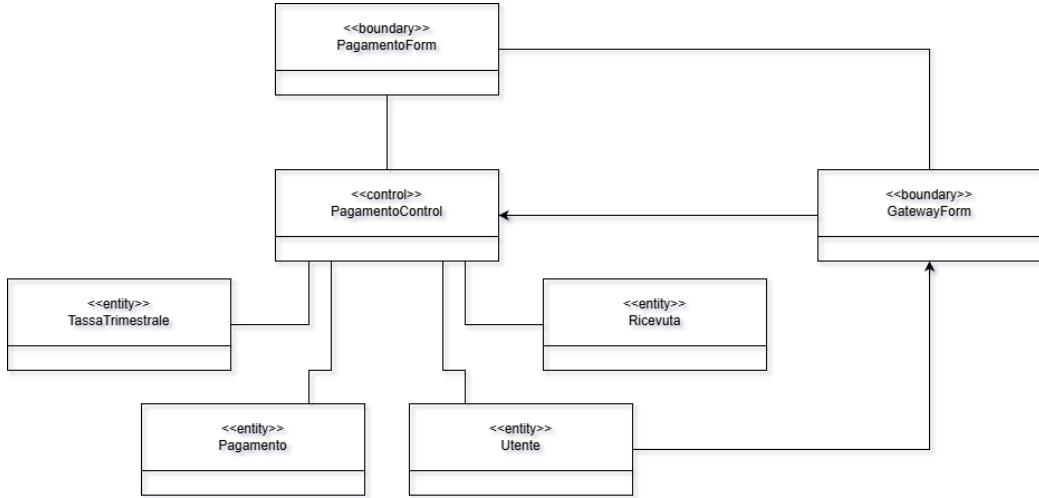
Tipo	Nome Classe	Descrizione
Boundary	EventoView	Interfaccia grafica che permette all'utente di visualizzare la lista degli eventi disponibili e iscriversi a uno di essi. Mostra messaggi di conferma o errore in base alla disponibilità dei posti.
Control	EventoControl	Coordina la logica di iscrizione: riceve la richiesta dell'utente dalla vista, verifica la disponibilità dei posti e registra la partecipazione creando un'associazione tra Utente ed Evento nella classe IscrizioneEvento.
Entity	Evento	Rappresenta un evento o un annuncio pubblicato dall'amministrazione. Contiene informazioni quali titolo, descrizione, data, luogo, posti disponibili e stato.
Entity	Utente	Rappresenta il coinquilino autenticato che intende partecipare all'evento. È collegato alla classe IscrizioneEvento.
Entity	IscrizioneEvento	Associazione che registra la partecipazione di un utente a un evento specifico. Contiene informazioni come data di iscrizione e stato della partecipazione.



Requisiti funzionali: RF16 – Prenotazione risorsa;

Caso d'uso associato: U13 – Richiesta Risorsa Condivisa con Accettazione Regole

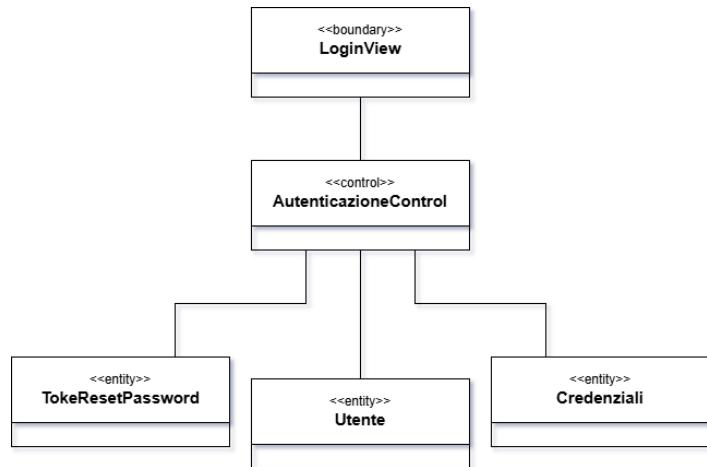
Tipo	Nome Classe	Descrizione
Boundary	RichiestaRisorsaForm	Interfaccia utente che consente all'utente di selezionare la risorsa desiderata, specificare periodo di utilizzo e inviare la richiesta.
Boundary	RegoleUsoView	Interfaccia che visualizza le regole d'uso della risorsa e richiede l'accettazione esplicita prima della conferma della richiesta.
Control	RisorsaControl	Coordina la logica applicativa della gestione delle risorse condivise. Verifica la disponibilità della risorsa, gestisce la conferma della richiesta e registra i dati nelle entità RichiestaRisorsa e RisorseCondivise.
Entity	RichiestaRisorsa	Contiene le informazioni relative alla richiesta della risorsa (utente, data inizio/fine, stato, risorsa selezionata).
Entity	Utente	Rappresenta il coinquilino autenticato che effettua la richiesta. È collegato alle entità delle richieste inviate.
Entity	RisorseCondivise	Rappresenta l'insieme delle risorse comuni disponibili nel sistema (sale, attrezzature, veicoli). Contiene informazioni come disponibilità, regole d'uso e penali.



Requisito funzionale: RF18 – Pagamento tassa trimestrale

Caso d'uso associato: UC14 – Pagamento Tassa Trimestrale

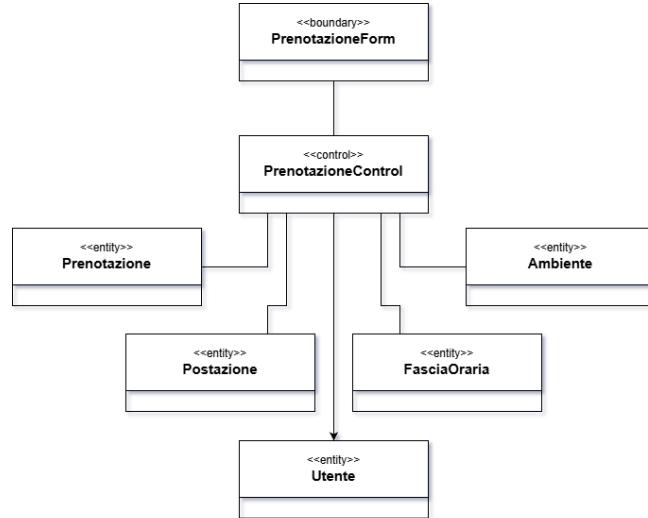
Tipo	Nome Classe	Descrizione
Boundary	PagamentoForm	Interfaccia grafica che consente all'utente di visualizzare l'importo dovuto, scegliere il metodo di pagamento e confermare la transazione. Mostra messaggi di esito positivo o negativo.
Control	PagamentoControl	Coordina la logica applicativa del pagamento: riceve i dati dal form, gestisce la transazione, aggiorna lo stato della tassa nel database e genera la ricevuta.
Entity	TassaTrimestrale	Rappresenta l'importo dovuto dall'utente in un determinato trimestre. Contiene informazioni su scadenze, stato (pagata/non pagata) e importo.
Entity	Pagamento	Contiene i dettagli della transazione effettuata (data, metodo, esito, importo). È associato a un'istanza di TassaTrimestrale.
Entity	Utente	Rappresenta il coinquilino autenticato che effettua il pagamento. È associato a uno o più record di Pagamento.
Entity	Ricevuta	Contiene i dati della ricevuta generata a seguito di un pagamento riuscito (id transazione, data, importo, riferimento utente).



Requisiti funzionali: RF02 – Login; RF04 – Recupero Password

Casi d'uso associati: UC09 – Autenticazione; UC10 – Autenticazione Fallita

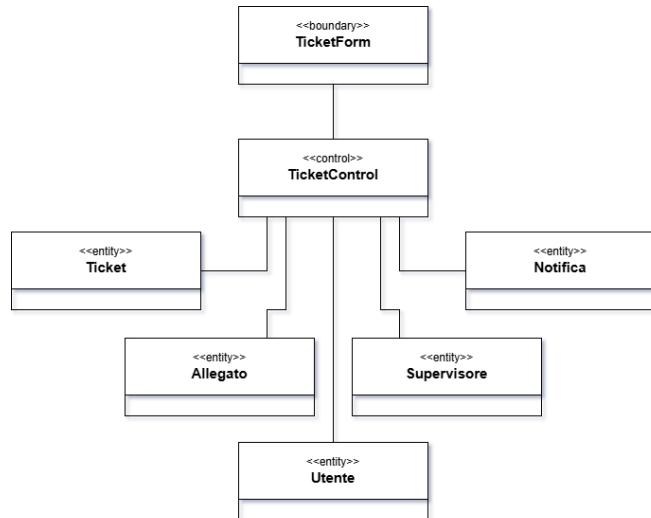
Tipo	Nome Classe	Descrizione
Boundary	LoginView	Interfaccia grafica che consente all'utente di inserire le proprie credenziali di accesso (email e password) e, in caso di smarrimento, di richiedere il reset della password. Mostra messaggi di errore o conferma.
Control	AutenticazioneControl	Gestisce la logica di autenticazione e recupero password. Riceve le credenziali dalla vista, le valida confrontandole con i dati memorizzati e, in caso di richiesta di reset, genera e associa un TokenResetPassword all'utente.
Entity	Credenziali	Contiene i dati di accesso (email e password cifrata). È collegata all'entità Utente e usata per l'autenticazione.
Entity	Utente	Rappresenta il coinquilino registrato che accede alla piattaforma. È collegato alle credenziali e, in caso di reset password, a un token temporaneo.
Entity	TokenResetPassword	Contiene il codice univoco generato per la procedura di recupero password. È temporaneo e associato a un solo utente.



Requisiti funzionali: RF06 – Creazione Prenotazione; RF07 – Modifica Prenotazione; RF08 – Cancellazione Prenotazione

Casi d'uso associati: UC11 – Prenotazione Postazione Sala Studio; UC03 – Modifica Prenotazione; UC04 – Cancellazione Prenotazione

Tipo	Nome Classe	Descrizione
Boundary	PrenotazioneForm	Interfaccia grafica che consente all'utente di selezionare l'ambiente, la postazione, la data e la fascia oraria. Gestisce le operazioni di prenotazione, modifica e cancellazione.
Control	PrenotazioneControl	Coordina la logica applicativa relativa alle prenotazioni: riceve i dati dal form, verifica la disponibilità della postazione nella fascia oraria selezionata e aggiorna il database.
Entity	Prenotazione	Contiene i dati relativi a una prenotazione (id, ambiente, postazione, fascia oraria, utente, data, stato).
Entity	Ambiente	Rappresenta lo spazio condiviso (es. sala studio, palestra, cucina comune) nel quale sono collocate le postazioni prenotabili.
Entity	Postazione	Indica la singola postazione disponibile all'interno di un ambiente. È collegata alle prenotazioni e alle fasce orarie.
Entity	FasciaOraria	Identifica l'intervallo temporale selezionabile per la prenotazione (es. 8:00–12:00, 12:00–15:00, 15:00–18:00).
Entity	Utente	Rappresenta il coinquilino autenticato che effettua la prenotazione. È collegato alle prenotazioni create.



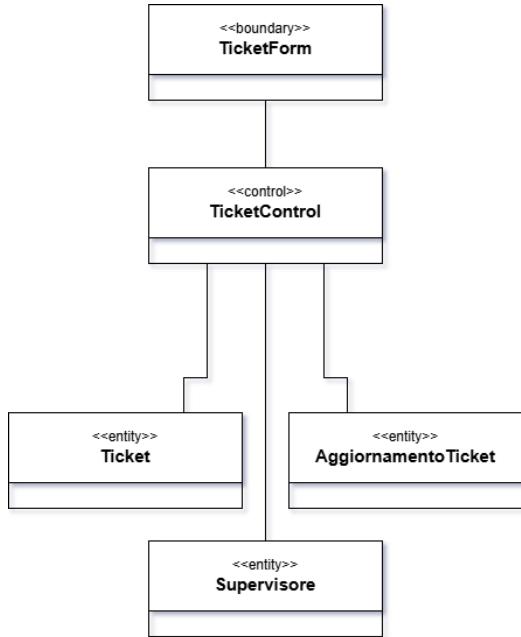
Requisiti funzionali: RF09 – Apertura Ticket; RF10 – Cancellazione Ticket; RF11 – Visualizzazione

Cronologia Ticket

Casi d'uso associati: UC13 – Creazione Ticket; UC14 – Cancellazione Ticket; UC05 – Visualizzazione

Cronologia Ticket

Tipo	Nome Classe	Descrizione
Boundary	<code>TicketForm</code>	Interfaccia grafica che consente all'utente di compilare il modulo di segnalazione, allegare immagini o documenti e inviare il ticket. Permette anche di visualizzare e cancellare le richieste esistenti.
Control	<code>TicketControl</code>	Coordina la logica applicativa dei ticket: riceve i dati dal form, valida le informazioni, gestisce la creazione e la cancellazione del ticket, notifica il supervisore e aggiorna lo stato della richiesta nel database.
Entity	<code>Ticket</code>	Rappresenta la richiesta di manutenzione generata da un utente. Contiene informazioni come codice identificativo, descrizione, categoria, stato, priorità, data di apertura e data di chiusura.
Entity	<code>Utente</code>	Rappresenta il coinquilino autenticato che crea o visualizza i propri ticket. È collegato alle entità <code>Ticket</code> e <code>Allegato</code> .
Entity	<code>Allegato</code>	Contiene eventuali file (foto o documenti) caricati dall'utente al momento della creazione del ticket. È associato a un solo <code>Ticket</code> .
Entity	<code>Supervisore</code>	Figura dell'Azienda Coordinatrice che riceve le segnalazioni e supervisiona l'avanzamento dei ticket. È notificato automaticamente dal sistema quando viene creato o cancellato un ticket.
Entity	<code>Notifica</code>	Rappresenta il messaggio di conferma o avviso inviato all'utente o al supervisore (es. "Ticket creato con successo", "Ticket cancellato", "Aggiornamento stato").



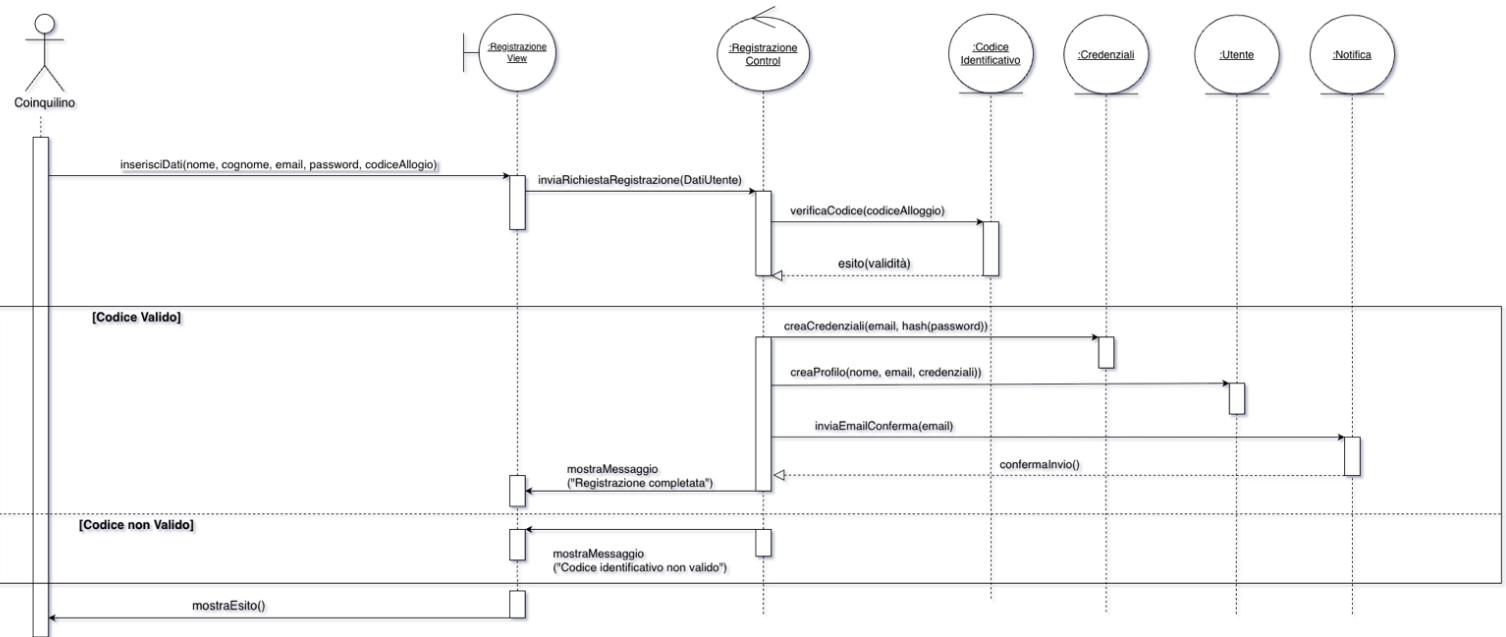
Requisiti funzionali: RF09 – Apertura Ticket; RF10 – Cancellazione Ticket; RF11 – Visualizzazione Cronologia Ticket

Caso d'uso associato: UC05 – Visualizzazione Cronologia Ticket

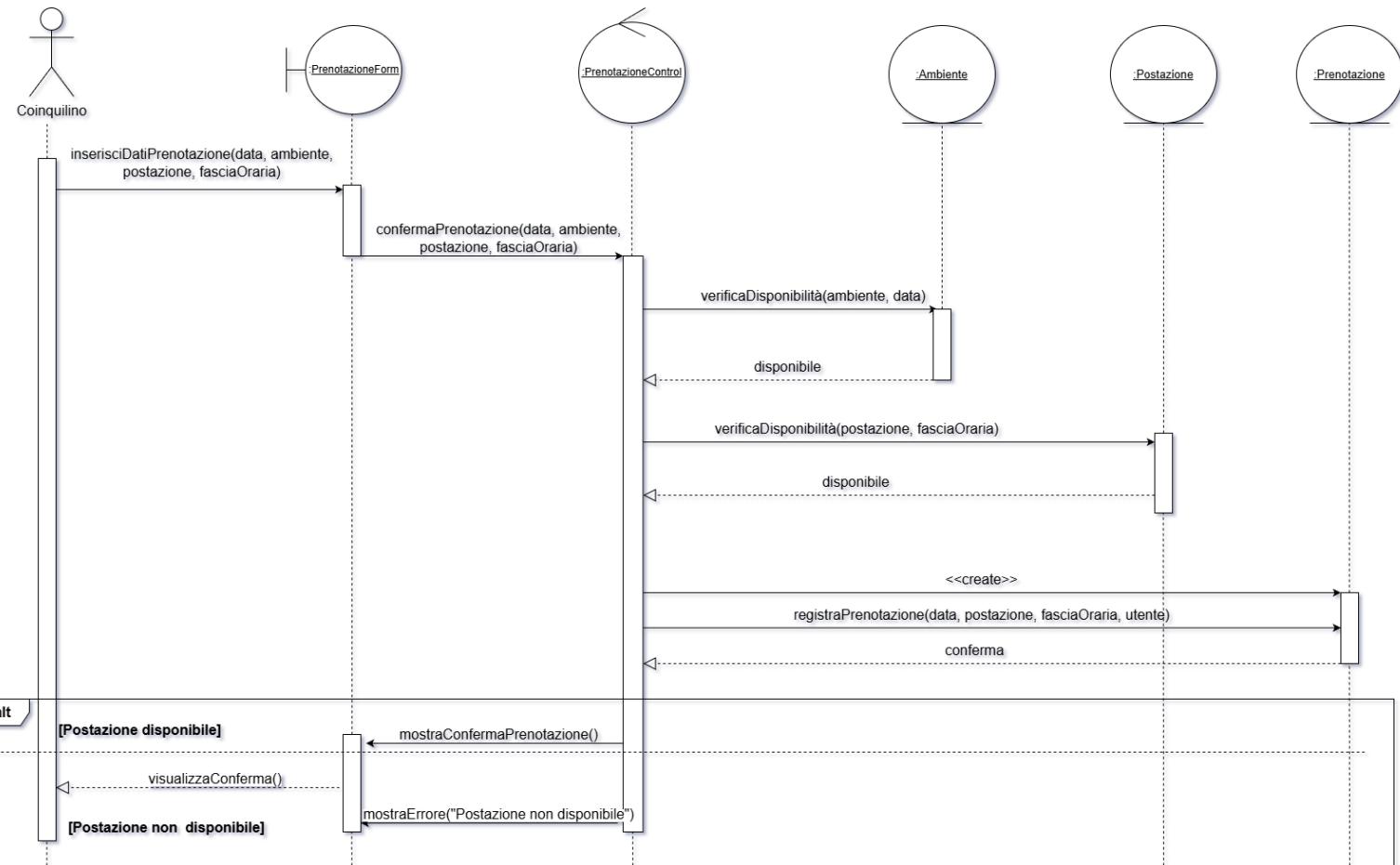
Tipo	Nome Classe	Descrizione
Boundary	TicketForm	Interfaccia grafica che consente la visualizzazione della lista dei ticket e dei relativi aggiornamenti. Permette all'utente o al supervisore di accedere ai dettagli del ticket e monitorarne lo stato.
Control	TicketControl	Coordina la logica applicativa per la gestione e l'aggiornamento dei ticket. Recupera le informazioni dal database, gestisce le modifiche di stato, registra nuovi aggiornamenti e comunica con le entità coinvolte.
Entity	Ticket	Rappresenta la richiesta di manutenzione, con attributi come descrizione, categoria, stato, priorità e data di creazione. È collegata ai relativi aggiornamenti.
Entity	AggiornamentoTicket	Contiene lo storico degli aggiornamenti di un ticket (es. cambio di stato, messaggi del supervisore, note tecniche). Ogni istanza è associata a un solo ticket.
Entity	Supervisore	Figura dell'Azienda Coordinatrice che monitora i ticket e ne aggiorna lo stato. È responsabile della verifica e dell'avanzamento delle richieste.

4.5.2 Dynamic model

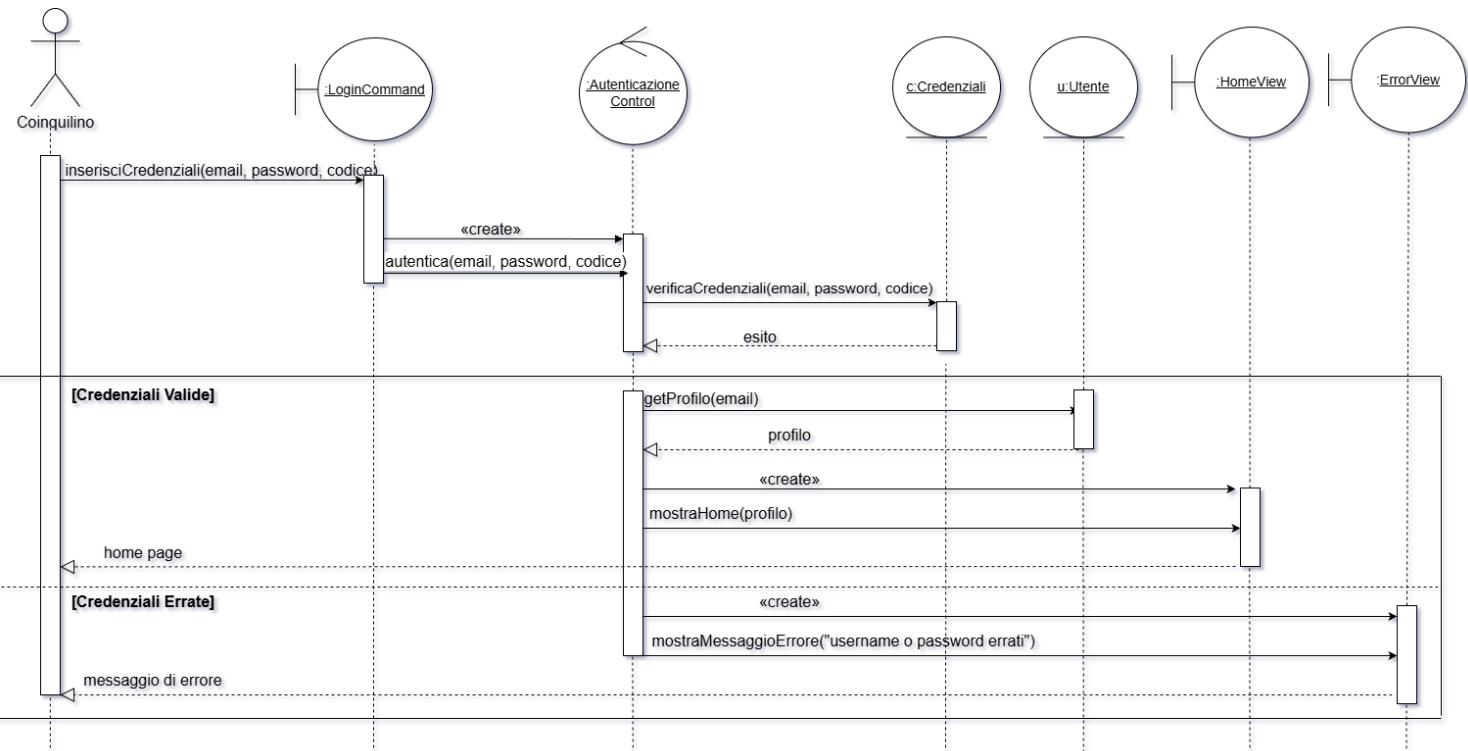
SD_UC02_Autenticazione



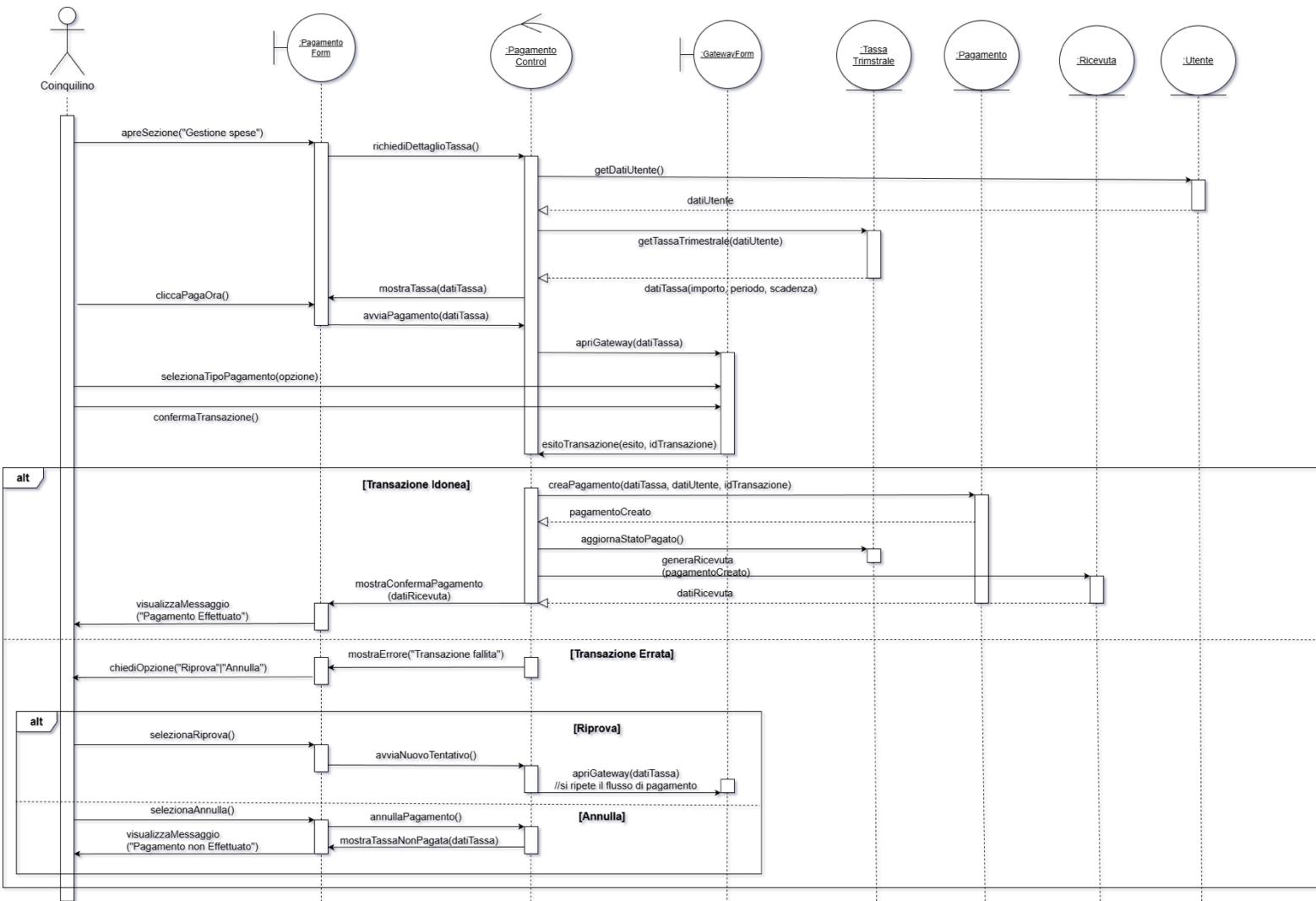
SD_UC07_PrenotazionePostazioneSalaStudio



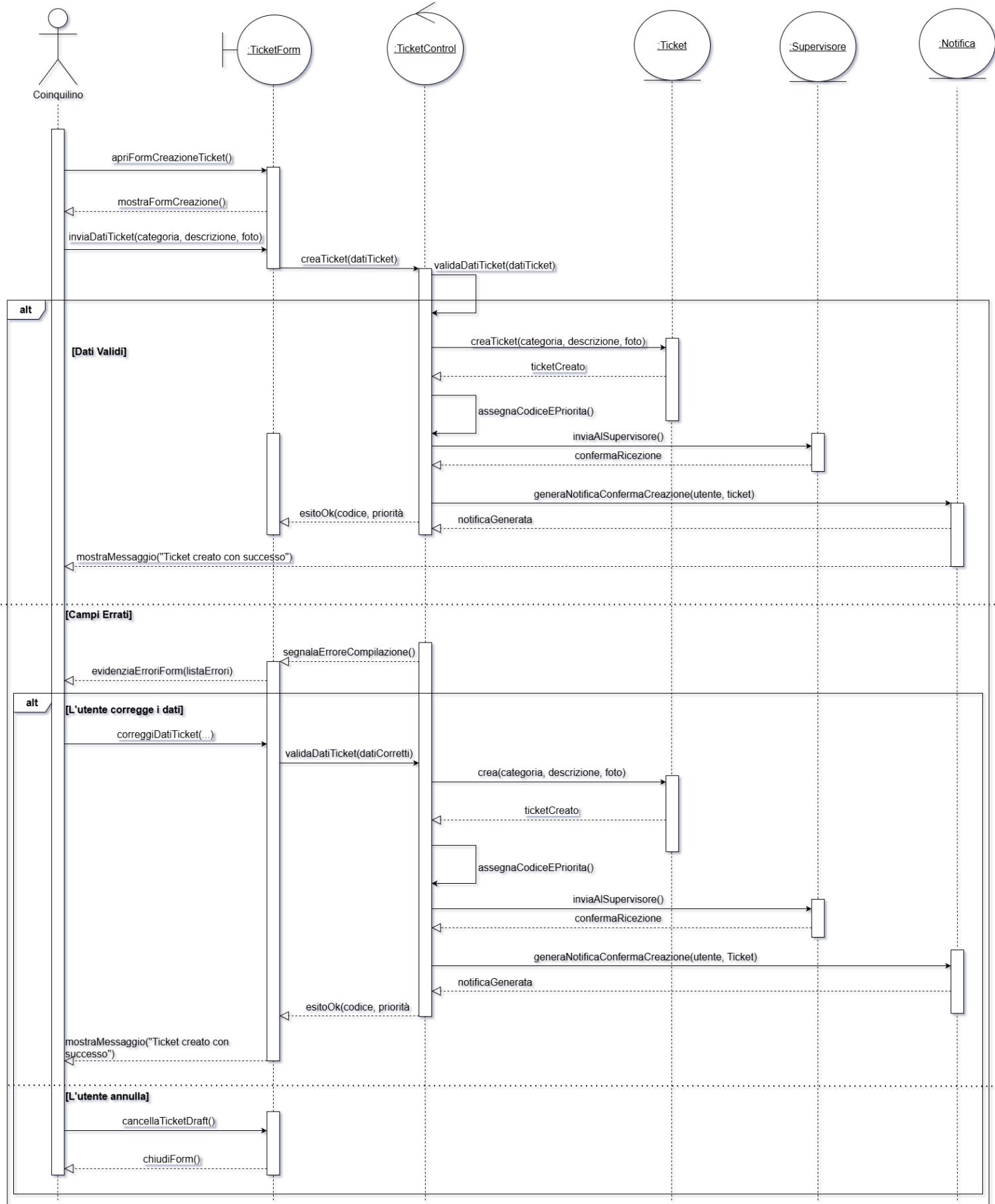
SD_UC01_RegistrazioneNuovoCoinquilino



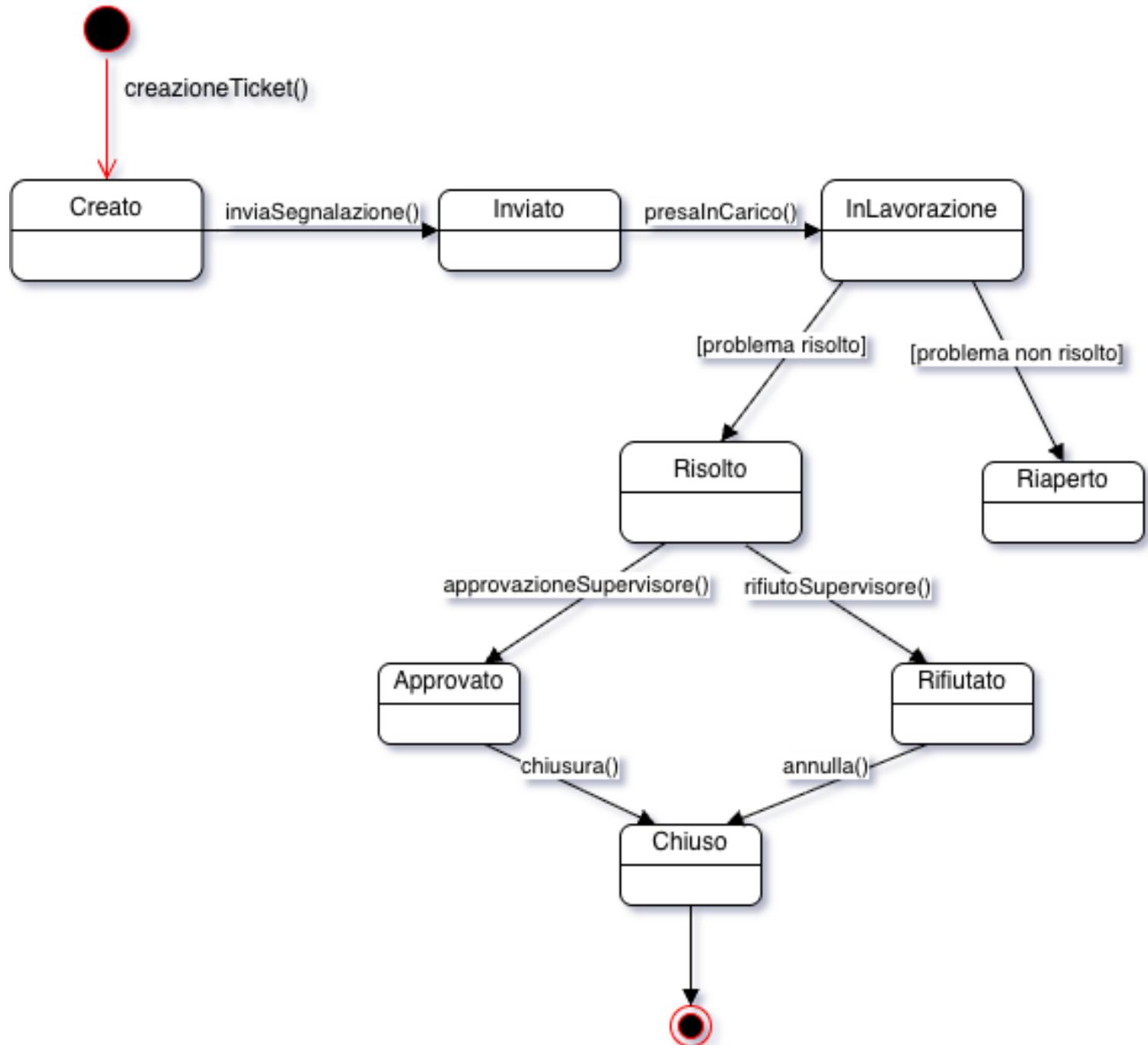
SD_UC14_PagamentoTassaTrimestrale



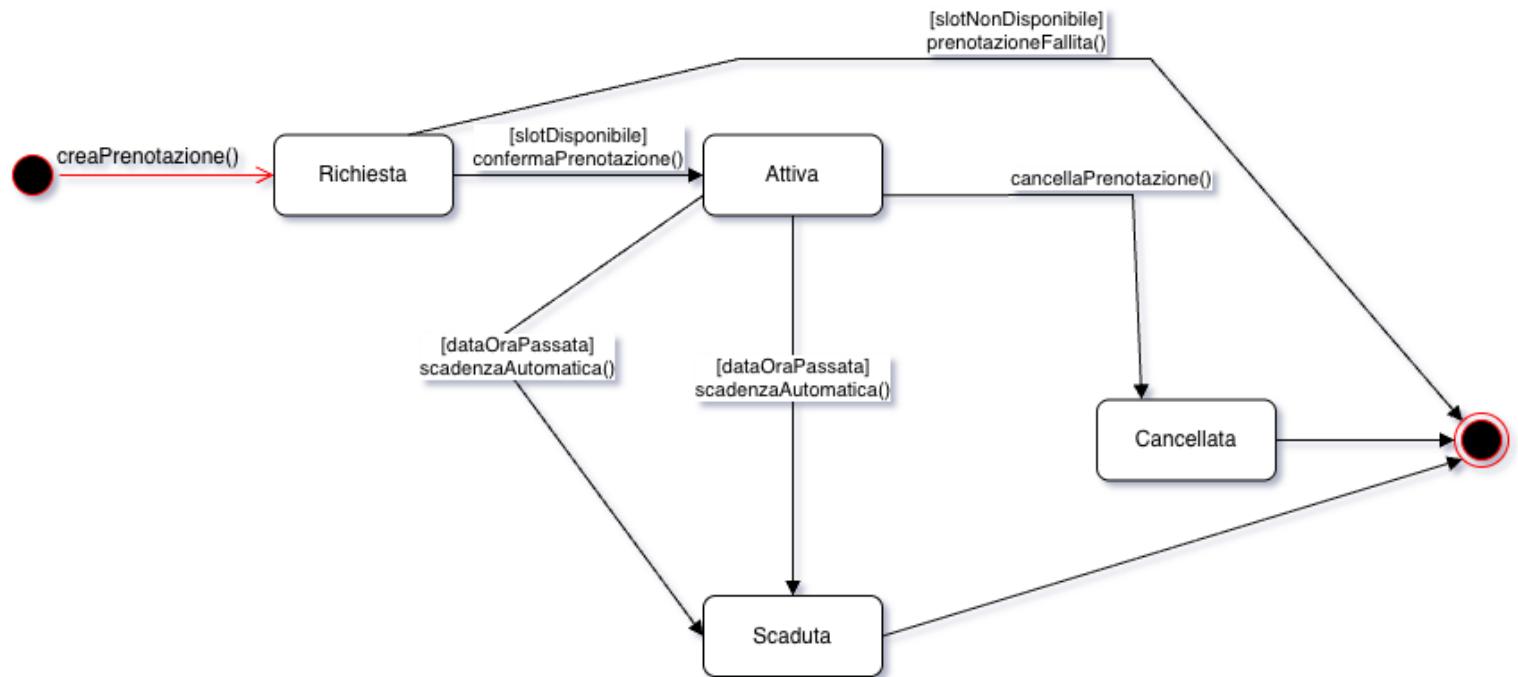
SD_UC09 _UC10_CreazioneTicket/CancellazioneTicket



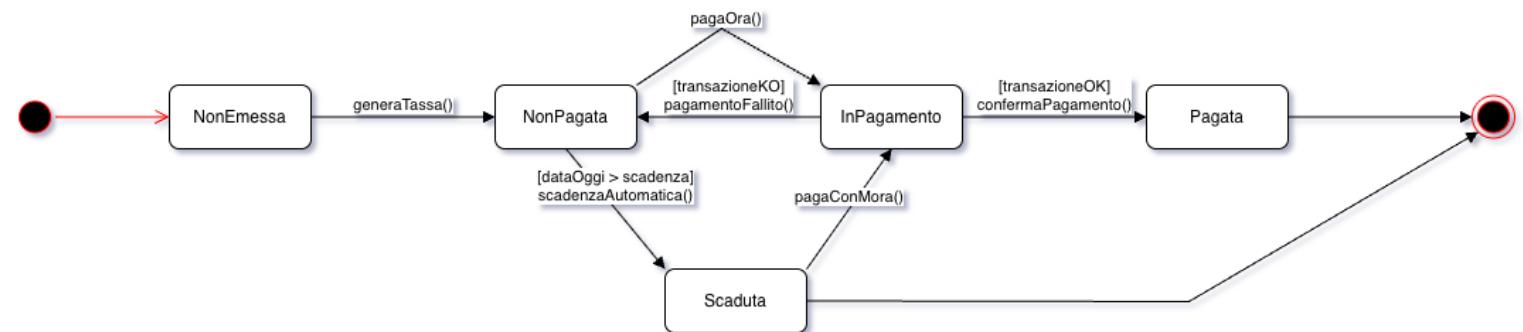
SCD_Ticket



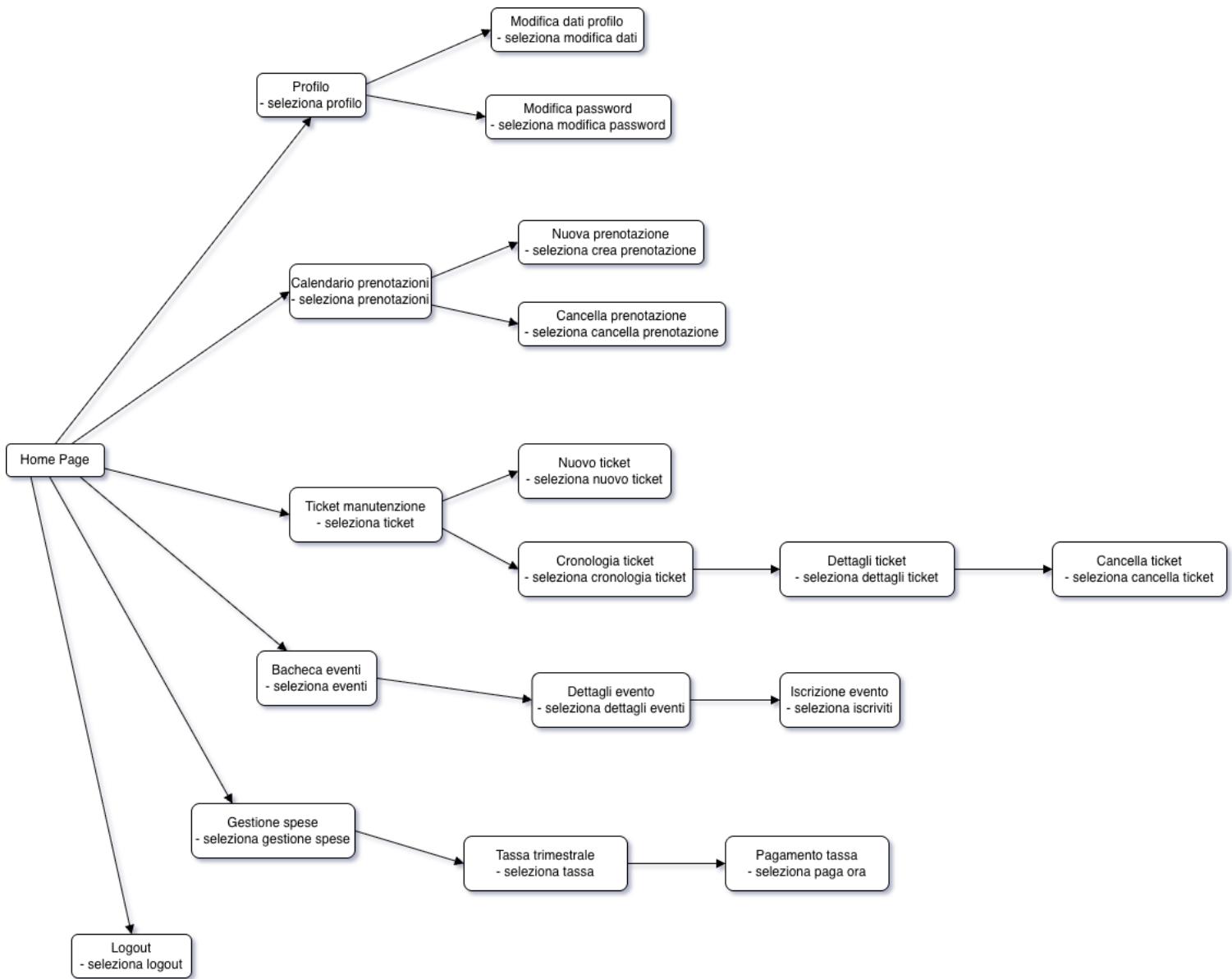
SCD_Prenotazione



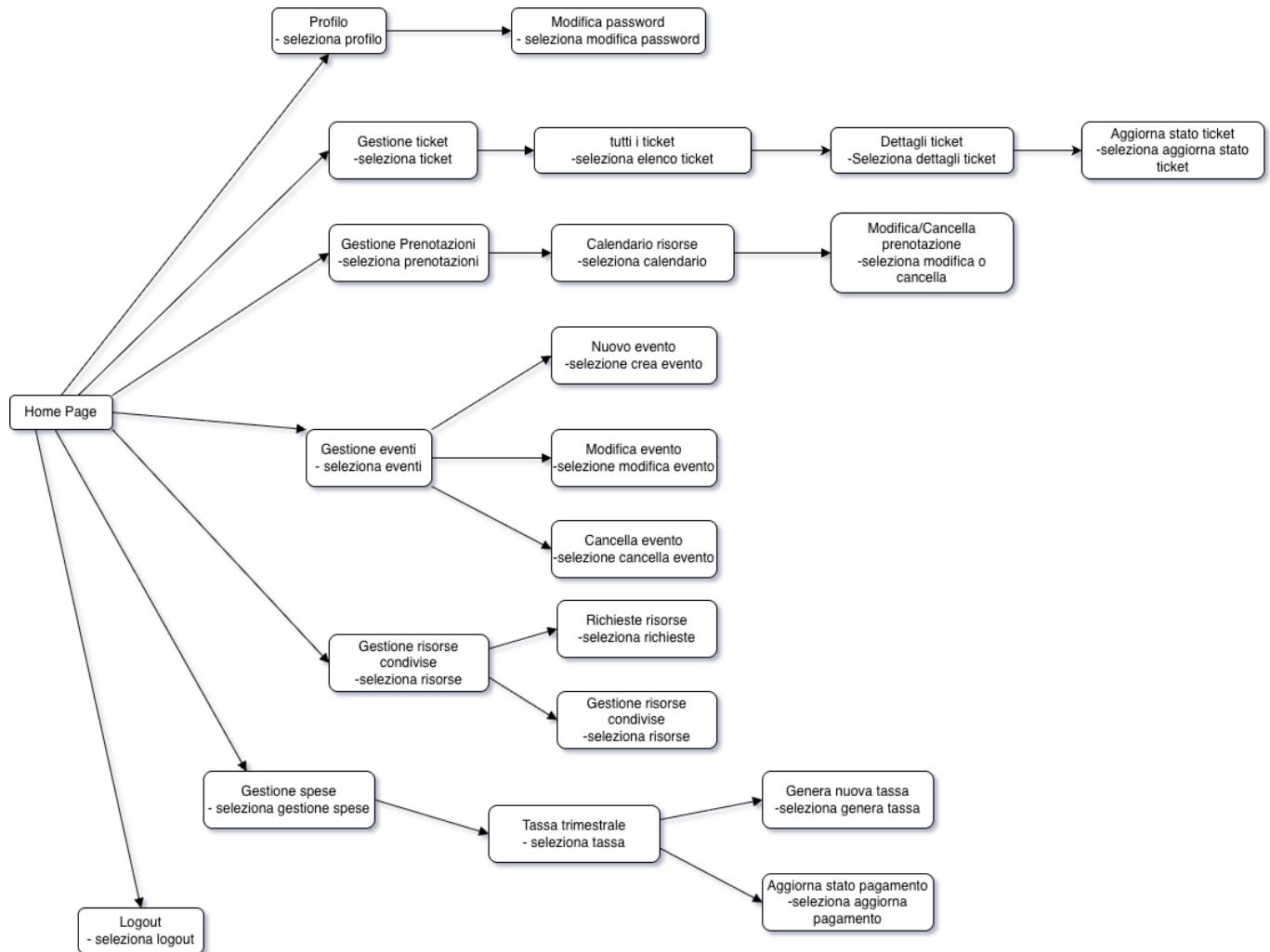
SCD_TassaTrimestrale



Path Navigazionale UI Coinquilino

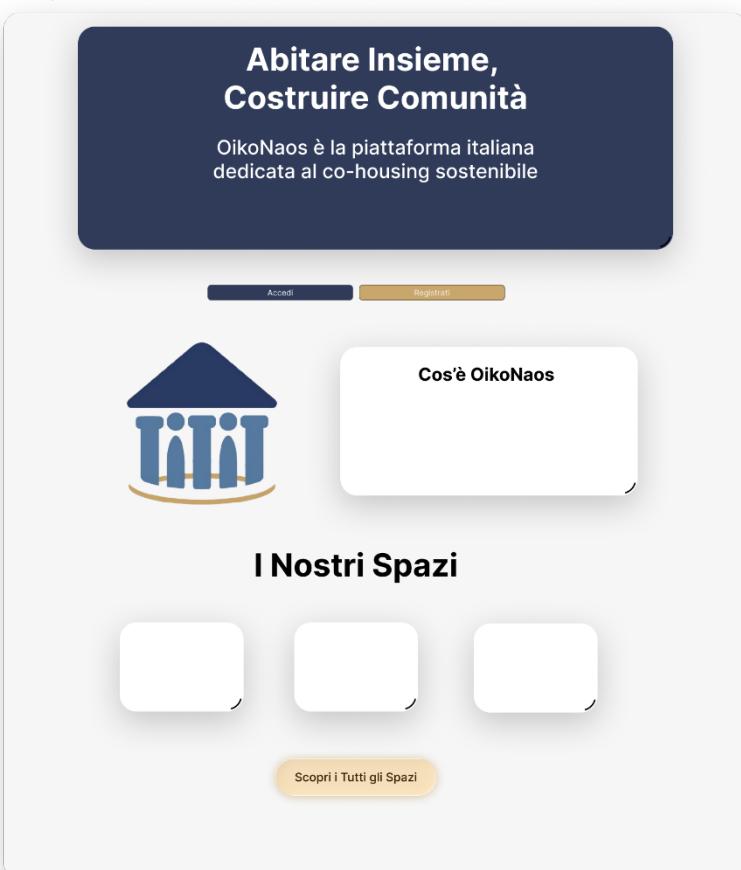


Path Navigazionale UI Azienda Coordinatrice

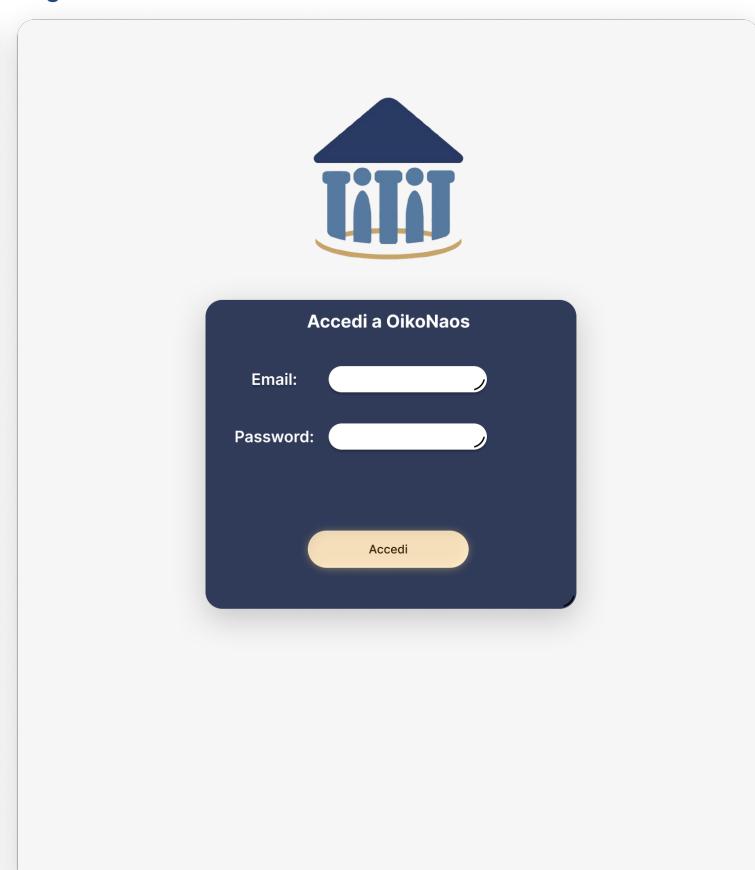


4.5.3 User interface

Img 1



Img 2



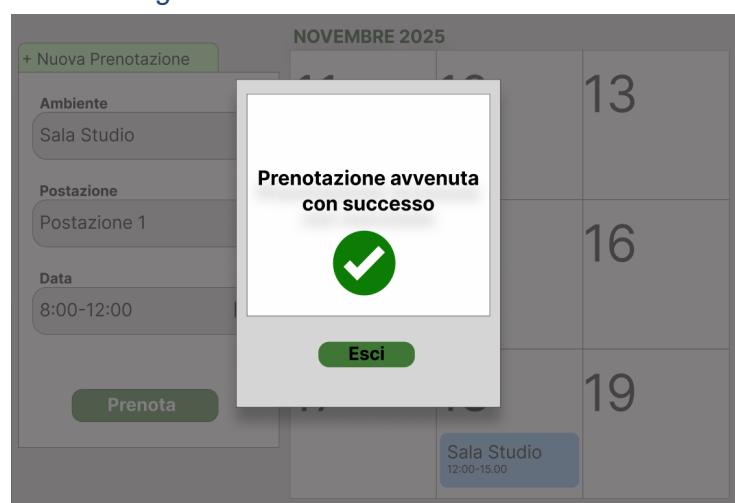
Img 3



Img 4



Img 5



Img 6



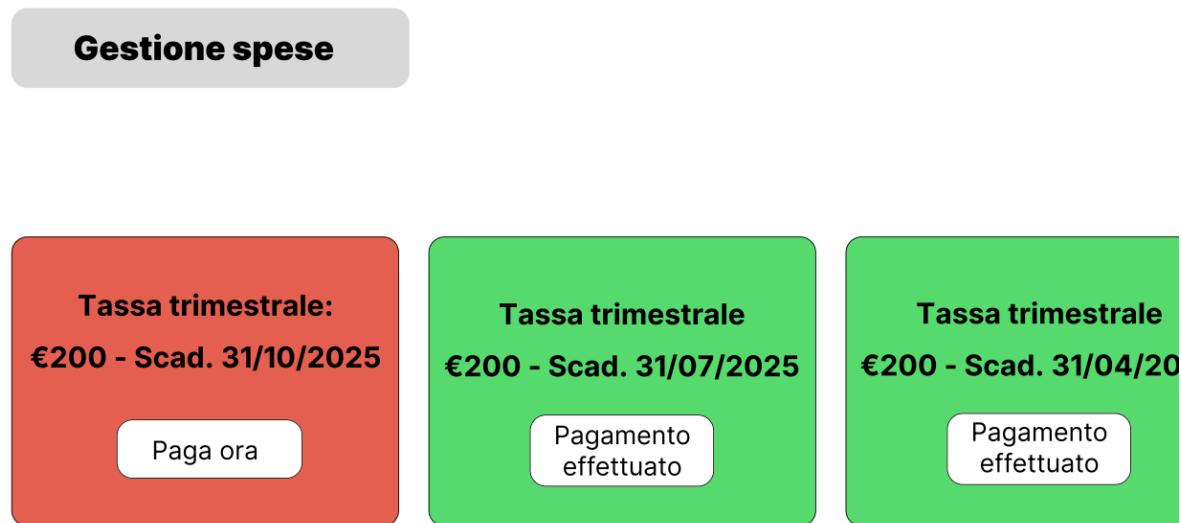
Img 7



Img 8



Img 9



Img 10

Dettagli pagamento

Pagamento tassa

Importo dovuto: €200

- Carta di credito
- Paypal

Procedi al pagamento

Img 11

Pagamento con carta

Pagamento con Paypal

Numero carta

Nome titolare

Data Scad.

CVV

Conferma pagamento

Conferma pagamento

Img 13

Img 12

Conferma pagamento riuscito

Pagamento riuscito!

La tua transazione è stata completata
con successo.

[Torna a 'Gestione spese'](#)

Transazione fallita

Si è verificato un errore.

Riprova o cambia metodo.

[Riprova](#)

[Annulla](#)

Img 14

I tuoi Ticket

Coinquilino

Stato



Periodo



Perdita acqua in cucina

24/10/2025

Aperto

[Vedi dettagli](#)

Porta del bagno rotta

20/10/2025

In lavorazione

[Vedi dettagli](#)

Riscaldamento non funzionante

15/09/2025

Chiuso

[Vedi dettagli](#)

Errata segnalazione

10/09/2025

Annullato

[Vedi dettagli](#)

+ Nuovo Ticket

← Torna alla lista

Perdita acqua in cucina

Aperto il: 24/10/2025

Priorità: Alta

Stato: Aperto

Descrizione

Perdita del tubo sotto il lavello.

L'acqua si accumula sul pavimento sotto

Storico aggiornamenti

25/10/2025: In lavorazione (Mario)

5. Glossary

Termini	Definizioni
Risorsa Condivisa	Qualsiasi bene o servizio utilizzabile da più residenti, gestito tramite prenotazione o richiesta (es. sala comune, auto elettrica, attrezzi). Ogni risorsa ha regole d'uso e può prevedere penali in caso di violazione.
Regole d'Uso	Condizioni specifiche che un utente deve accettare prima di utilizzare una risorsa condivisa. Servono a garantire corretto utilizzo e responsabilità individuale. L'accettazione è esplicita e tracciata dal sistema.
GDPR	Legge dell'Unione Europea che stabilisce norme per la raccolta, l'uso e la conservazione dei dati personali dei residenti UE.
Hashing / Bcrypt / SHA-256	Tecniche di cifratura unidirezionale utilizzate per memorizzare le password in modo sicuro nel database, rendendole non leggibili anche in caso di accesso non autorizzato.
Entity Object / Boundary Object / Control Object	Categorie di oggetti nel modello UML orientato agli oggetti: <ul style="list-style-type: none">• Entity Object: rappresenta dati persistenti (es. "Utente", "Prenotazione", "Ticket").• Boundary Object: interfaccia tra utente e sistema (es. "Pagina di login", "Modulo prenotazione").• Control Object: gestisce la logica applicativa (es. "PrenotazioneController").