

**OikoNaos**  
**Requirements Analysis**  
**Versione 3.2**



Data: 26/10/2025

Progetto: OikoNaos	Versione: 3.2
Documento: Requirement Analysis	Data: 27/10/2025

**Coordinatore del progetto:**

Nome	Matricola
Luigi Potestà	0512120076

**Partecipanti:**

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

<b>Scritto da:</b>	Giulia Buonafine
--------------------	------------------

**Revision History**

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

10/10/2025	2.2	<b>Problem Statement:</b> Realizzazione scenari e inserimento scadenze	Giulia Buonafine
13/10/2025	2.3	<b>Problem Statement:</b> Inserimento requisiti non funzionali e ridefinizione ambiente di sviluppo	Maria Antonietta Ruggiero
21/10/2025	3.0	<b>Requirements Analysis:</b> Creazione struttura del documento e definizione del purpose e audience	Mario Sinopoli
23/10/2025	3.1	<b>Requirements Analysis:</b> Scrittura della parte introduttiva dal punto 1.1.1 a 1.1.6	Maria Antonietta Ruggiero
26/10/2025	3.2	<b>Requirements Analysis:</b> Definizione del Sistema Attuale, Sistema Proposto e modelli di sistema	Giulia Buonafine, Luigi Potestà

# Indice

1.	Purpose.....	5
2.	Audience .....	5
2.1	Purpose of the System .....	5
2.2	Scope of the System .....	5
2.3	Objectives and success criteria of the project .....	6
2.4	Definitions, acronyms and abbreviations .....	6
2.5	Overview .....	6
3.	Current System.....	6
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	Legal .....	10
4.4	System Models .....	10
4.4.1	Scenarios.....	10
4.5	Use case model.....	11
4.5.1	Object model.....	16
4.5.2	Dynamic model .....	16
4.5.3	User interface.....	16
5.	Glossary .....	20

# 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 coinquilini, 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 architettuale 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 disagi, 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 la registrazione, l'utente deve inserire i propri dati personali (nome, cognome, email, password e eventuali informazioni di contatto) e un codice identificativo speciale fornito dall'amministrazione della co-housing. Il sistema deve verificare la validità del codice prima di completare la registrazione
- **RF02. Login:** Il sistema deve consentire all'utente registrato di accedere alla piattaforma inserendo le proprie credenziali (email e password). In caso di credenziali errate, deve essere mostrato un messaggio di errore.
- **RF03. Logout:** Il sistema deve permettere all'utente autenticato di terminare la propria sessione in modo sicuro, reindirizzandolo alla pagina di login.
- **RF04. Recupero password:** Il sistema deve consentire agli utenti di recuperare la propria password tramite email, generando un link temporaneo di reimpostazione.
- **RF05. Gestione profilo:** Il sistema deve permettere all'utente di visualizzare e modificare i propri dati personali (nome, email, foto profilo, password), garantendo la persistenza delle modifiche.

### Calendario Prenotazioni

- **RF06. Creazione prenotazione:** Il sistema deve consentire all'utente di creare una nuova prenotazione selezionando data, ora e risorsa disponibile, registrando i dati nel database.
- **RF07. Modifica prenotazione:** Il sistema deve permettere all'utente di modificare una prenotazione esistente, aggiornando le informazioni (data, ora o risorsa) nel rispetto delle regole di disponibilità.
- **RF08. Cancellazione prenotazione:** Il sistema deve consentire all'utente di cancellare una prenotazione precedentemente creata, rendendo di nuovo disponibile la risorsa associata.

### Ticket di Manutenzione

- **RF09. Apertura ticket:** Il sistema deve permettere agli utenti di segnalare un guasto o una richiesta di manutenzione compilando un modulo con descrizione, priorità e foto opzionale.
- **RF10. Cancellazione ticket:** Il sistema deve consentire all'utente di annullare un ticket aperto, purché non sia già stato preso in carico dagli addetti alla manutenzione.
- **RF11. Visualizzazione cronologia ticket:** Il sistema deve fornire all'utente un elenco dei ticket aperti, in lavorazione e chiusi, con informazioni su data, stato e eventuali risposte ricevute. Sull'elenco sono applicabili dei filtri sulla base dello stato e del periodo apertura dei ticket.

### Bacheca eventi e annunci

- **RF12. Visualizzazione della bacheca:** Il sistema deve consentire a tutti gli utenti autenticati di visualizzare la bacheca degli eventi e degli annunci pubblicati dall'amministrazione.
- **RF13. Iscrizione agli eventi:** Il sistema deve permettere all'utente di iscriversi a un evento disponibile, registrando la partecipazione nel database e aggiornando il numero dei posti disponibili.
- **RF14. Visualizzazione mappa:** Il sistema deve fornire una mappa della struttura per guidare i coinquilini verso i vari ambienti.

### Gestione risorse condivise

- **RF15. Richiesta risorsa:** Il sistema deve permettere all'utente di richiedere l'utilizzo di una risorsa condivisa (es. sala comune, veicolo, attrezzatura), verificandone la disponibilità nel periodo specificato dall'utente.
- **RF16. Presa visione delle regole d'uso:** Prima di confermare la richiesta di una risorsa, il sistema deve mostrare all'utente le regole di utilizzo, richiedendone l'accettazione esplicita. In caso di infrazione è prevista una penale da pagare.
- **RF17. Visualizzazione cronologia risorse condivise:** Il sistema deve consentire all'utente di consultare lo storico delle risorse utilizzate, incluse data di eventuale presa e restituzione, orari e stato delle richieste. Nella cronologia è possibile filtrare le risorse attraverso le voci "restituite" e "in uso".

### Gestione Spese

- **RF18. Pagamento tassa trimestrale:** Il sistema deve consentire all'utente di visualizzare gli importi dovuti



e di effettuare il pagamento della tassa trimestrale tramite un metodo di pagamento integrato (es. carta o bonifico), aggiornando automaticamente lo stato del pagamento.

## 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, senza consultare documentazione esterna.
- **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 sistema:** Il sistema deve garantire una disponibilità operativa minima del 99% durante il periodo di attività previsto.
- **RNF04. Recupero in caso di errore:** 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. Robustezza ai guasti:** 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 un Pagina 7 di 9 Ingegneria del Software Ingegneria del Software Pagina 8 di 9 livello di accuratezza del 99,9%.
- **RNF09. Disponibilità del servizio:** Il servizio deve essere accessibile 24 ore su 24, 7 giorni su 7, salvo periodi di manutenzione pianificata.

### 4.3.4 Supportability

- **RNF10. Documentazione del codice:** Il codice sorgente deve essere commentato e documentato in conformità agli standard di sviluppo adottati. Ogni funzione deve avere almeno un commento descrittivo e un'intestazione con autore e data di modifica.
- **RNF11. Configurabilità:** Le principali impostazioni del sistema (es. parametri di connessione, lingua, limiti di sessione) devono poter essere modificate senza intervento sul codice sorgente.
- **RNF12. Internazionalizzazione:** Il sistema deve supportare la traduzione dei testi in almeno due lingue tramite file di risorse esterni.
- **RNF13. Portabilità:** L'applicazione deve poter essere eseguita su diversi ambienti server e browser (almeno Chrome, Firefox, Edge, Safari) senza modifiche al codice.

### 4.3.5 Implementation

- **RNF14. 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).
- **RNF16 Strumenti di Sviluppo:** L'ambiente di sviluppo primario deve essere **IntelliJ 2025.2**, e la gestione delle versioni del codice deve avvenire tramite **Git**.

- **RNF17 Modellazione:** L'implementazione deve essere coerente con i modelli **UML (Object Model, Dynamic Model)** definiti nella fase di analisi.

### 4.3.6 Interface

- **RNF18 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.
- **RNF19 Notifiche:** Il sistema deve supportare l'invio di notifiche via **email** agli utenti in caso di eventi critici (es. conferma di registrazione, recupero password).
- **RNF20 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

- **RNF21 Distribuzione:** Il sistema deve essere impacchettato come file **.WAR** (Web Application Archive) per la distribuzione sull'Application Server Tomcat.
- **RNF22 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.
- **RNF23 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 Legal

- **RNF24 Protezione Dati Personali:** 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.
- **RNF25 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.
- **RNF26 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**.
- **RNF27 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.

## 4.4 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 email **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 email: **luca.g@gmail.com** e la sua password **3050ciao!** accedendo così alla home page. Dal menu a tendina 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.

### **Modifica prenotazione sala studio**

Il coinquilino Luca apre la sezione "Calendario prenotazioni". Scorrendo le sue prenotazioni attive, nota di aver prenotato erroneamente la sala studio per il giorno 15/10/2025 dalle 8:00 alle 12:00, ma desidera spostarla al pomeriggio. Il coinquilino seleziona quindi la prenotazione in questione e clicca sull'opzione "Modifica". Il sistema gli permette di aggiornare i dati scegliendo una nuova fascia oraria (15:00–18:00) per lo stesso giorno, verificando automaticamente che la postazione sia ancora disponibile in quell'orario. Dopo aver confermato la modifica, il sistema aggiorna il calendario e mostra un messaggio di conferma: "Prenotazione modificata con successo". La nuova fascia oraria risulta ora occupata (colorata in rosso) e quella precedente torna disponibile per gli altri utenti.

### **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 OikoNaos, entra nella sezione "Calendario prenotazioni" e seleziona 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 per gli altri coinquilini, e Matteo riceve un messaggio di conferma dell'avvenuta cancellazione visualizzabile anche nella sezione "Storico prenotazioni".

### **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 email: sara09@gmail.com e la sua password 8960sara!! accedendo così alla home page. Dal menu a tendina seleziona la voce "Segnalazioni" poi la categoria del guasto ("Tv"), una breve descrizione del problema e opzionalmente una foto per poi confermare la sua scelta. Il sistema ora elaborerà la richiesta a cui assegna automaticamente una priorità e inoltra il ticket 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 "Ticket" in cui verranno ci saranno aggiornamenti automatici. L'utente può inoltre scegliere di cancellare il ticket, purché 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 comune tramite la piattaforma, si accorge che il problema è stato risolto autonomamente da un altro residente. Decide quindi di annullare il ticket appena inviato. Dopo aver effettuato il login con le proprie credenziali, Matteo accede alla sezione "Ticket" del proprio profilo personale, dove è presente l'elenco delle richieste aperte. Seleziona il ticket relativo al guasto del frigorifero e clicca sull'opzione "Cancella ticket". Il sistema verifica che la segnalazione non sia ancora stata presa in carico dal supervisore o dagli addetti alla manutenzione; constatata la disponibilità all'annullamento, procede con l'eliminazione del 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 schermata iniziale e seleziona l'opzione "Registrati ora", possedendo il Codice Identificativo Speciale fornito dall'Azienda Coordinatrice.

**Flusso di Eventi:**

1. L'utente accede alla schermata di registrazione.

2. L'utente inserisce i dati personali (nome, email, password, contatti) e il **Codice Identificativo Speciale**.
3. L'utente conferma l'invio dei dati.
4. Il sistema **verifica la validità del Codice Identificativo Speciale**.
5. Il sistema salva le credenziali cifrate e crea il profilo.
6. Il sistema invia una notifica di avvenuta registrazione (es. via email).

**Exit Condition:** Il nuovo Coinquilino è registrato nel database e viene reindirizzato alla schermata di Login.

**Eccezioni:**

- **Codice Invalido:** Se il Codice Identificativo Speciale risulta errato (punto 4), il sistema mostra un messaggio di errore e blocca la creazione dell'account.

## UC02. Gestione Profilo Utente

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

**Attore:** Utente (Coinquilino).

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

**Flusso di Eventi:**

1. L'utente seleziona la voce "Profilo personale" o l'icona utente.
2. Il sistema mostra la schermata del profilo con i dati attuali (nome, email, password cifrata, foto).
3. L'utente seleziona l'opzione di modifica (es. "Modifica Dati").
4. L'utente aggiorna uno o più campi (es. modifica la foto profilo o la password).
5. L'utente clicca "Salva Modifiche".
6. Il sistema convalida i nuovi dati (es. verifica la validità della nuova email).
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:**

- **Cambio Password:** Se l'utente modifica la password, il sistema può richiedere di reinserire la vecchia password come misura di sicurezza.

## UC03. Modifica Prenotazione

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

**Attore:** Utente (Coinquilino).

**Entry Condition:** L'utente è autenticato e ha una o più prenotazioni attive.

**Flusso di Eventi:**

1. L'utente accede alla sezione "Calendario prenotazioni".
2. L'utente seleziona la prenotazione che intende modificare.
3. L'utente clicca sull'opzione "Modifica Prenotazione".
4. L'utente seleziona una nuova risorsa, data o fascia oraria.
5. Il sistema **verifica la disponibilità** della nuova selezione.
6. Il sistema aggiorna il calendario, annullando la vecchia prenotazione e attivando la nuova.
7. Il sistema mostra un messaggio di conferma ("Prenotazione modificata con successo").

**Exit Condition:** La prenotazione originale è aggiornata con i nuovi parametri e la disponibilità della risorsa viene modificata di conseguenza.

**Flussi Alternativi / Eccezioni:**

- **Modifica Fallita:** Se il sistema rileva che la nuova risorsa o fascia oraria (punto 5) è già occupata, mostra un messaggio di errore e l'utente torna alla pagina di modifica.

## UC04. Cancellazione Prenotazione

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

**Attore:** Utente (Coinquilino).

**Entry Condition:** L'utente è autenticato e ha una prenotazione attiva.

**Flusso di Eventi:**

1. L'utente accede alla sezione "Calendario prenotazioni".
2. L'utente seleziona la prenotazione da annullare.
3. L'utente clicca sull'opzione "Cancella Prenotazione".
4. Il sistema richiede una **conferma esplicita** all'utente per evitare annullamenti accidentali.

5. L'utente conferma l'annullamento.
6. Il sistema rimuove la prenotazione dal calendario e aggiorna il database.
7. Il sistema mostra un messaggio di conferma e rende la risorsa immediatamente disponibile.

**Exit Condition:** La prenotazione è stata annullata e la risorsa associata è nuovamente disponibile.

#### UC05. Visualizzazione Cronologia Ticket

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

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

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

**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.
3. L'utente utilizza i filtri per **stato** (Aperto, In Lavorazione, Chiuso, Annullato) o per **periodo di apertura**.
4. Il sistema aggiorna la lista in base ai filtri selezionati.
5. L'utente può selezionare un ticket per visualizzarne il dettaglio completo (descrizione, priorità, storico degli aggiornamenti).

**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, la lista include *tutti* i ticket della co-housing (non solo quelli creati dall'utente).

#### UC06. Iscrizione a Evento

Questo caso d'uso copre il requisito **RF13**.

**Attore:** Utente (Coinquilino).

**Entry Condition:** L'utente è autenticato e l'evento selezionato è ancora disponibile.

**Flusso di Eventi:**

1. L'utente accede alla sezione "Bacheca Eventi".
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 un messaggio di conferma di iscrizione.

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

**Eccezioni:**

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

#### UC07. Richiesta Risorsa Condivisa con Accettazione Regole

Questo caso d'uso copre i requisiti **RF15** e **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 seleziona la risorsa desiderata (es. veicolo, attrezzatura) e il periodo di utilizzo (RF15).
2. Il sistema verifica la disponibilità.
3. Prima della conferma, il sistema mostra le **Regole d'Uso** specifiche per quella risorsa (RF16).
4. L'utente deve selezionare esplicitamente una casella di spunta per **accettare le regole d'uso**.
5. L'utente clicca "Richiedi Risorsa".
6. Il sistema registra la richiesta e la data di previsione di presa/restituzione nel database.
7. Il sistema fornisce un messaggio di conferma.

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

**Flussi Alternativi:**

- **Mancata Accettazione:** Se l'utente non accetta esplicitamente le regole (punto 4), la richiesta non viene finalizzata.

#### UC08. Pagamento Tassa Trimestrale

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

**Attore:** Utente (Coinquilino).

**Entry Condition:** L'utente è autenticato e presenta un debito attivo per la tassa trimestrale.

**Flusso di Eventi:**

1. L'utente accede alla sezione "Gestione Spese" o riceve una notifica di pagamento.
2. Il sistema mostra l'**importo dovuto** e la **scadenza**.
3. L'utente seleziona l'opzione "Paga Ora".
4. Il sistema reindirizza l'utente a un'interfaccia di pagamento integrata (servizio esterno/interno).
5. L'utente inserisce i dettagli del metodo di pagamento (es. carta/bonifico) e conferma la transazione.
6. Il sistema riceve la conferma di successo dalla piattaforma di pagamento.
7. Il sistema **aggiorna automaticamente lo stato del pagamento** nel database da "Non Pagato" a "Pagato".

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

**Flussi Alternativi / Eccezioni:**

- **Pagamento Fallito:** Se la transazione viene rifiutata (punto 6), il sistema notifica l'utente e mantiene lo stato della tassa su "Non Pagato".

#### UC09. Autenticazione

**Attore:** Utente (Coinquilino);

**Entry Condition:** Il coinquilino si trova nella home page del sito nella schermata per effettuare l'autenticazione.

**Flusso di eventi:**

1. Il coinquilino inserisce username e password
2. Il coinquilino invia i dati al sistema
3. Il sistema effettua un controllo sulle credenziali ed esso ha esito positivo
4. Il sistema reindirizza l'utente alla sua home page

**Exit Condition:** Il coinquilino autenticato e si trova nella homepage

**Flussi alternativi / Eccezioni:** 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 log-in (2. Autenticazione fallita).

#### UC10. Autenticazione fallita

**Attore:** Utente (Coinquilino);

**Entry Condition:** Il coinquilino tenta di effettuare il log-in con credenziali errate.

**Flusso di eventi:**

1. Il sistema mostra il messaggio di errore "username o password errate".
2. L'utente rimane sulla schermata di login.

**Exit Condition:** L'utente si trova nella home page del sito con la schermata di autenticazione.

#### UC11. Prenotazione postazione sala studio

**Attore:** Utente (Coinquilino);

**Entry Condition:** L'utente è autenticato e si trova nella home page.

**Flusso di eventi:**

1. L'utente seleziona la voce Calendario.
2. L'utente sceglie l'ambiente, postazione, data e fascia orario.
3. L'utente clicca "Prenota".
4. Il sistema verifica la disponibilità.
5. Se tale selezione è disponibile, l'utente conferma la prenotazione.
6. Il sistema aggiorna il calendario.

**Exit Condition:** L'utente ha prenotato la postazione e il sistema mostra che essa è occupata nella data e ora selezionata.

**Flussi alternativi:** Se nel punto 5 il sistema mostra che tale selezione non è disponibile (4. Prenotazione fallita e le altre fasce orarie e postazioni disponibili).

#### **UC12. Prenotazione fallita**

**Attore:** Coinquilino (Utente);

**Entry Condition:** L'utente seleziona una postazione già occupata.

**Flusso di eventi:**

1. Il sistema mostra il messaggio di errore "Postazione non disponibile nella data e fascia oraria selezionata".
2. Il sistema permette all'utente di scegliere un'altra postazione o un'altra fascia oraria.

**Exit condition:** L'utente resta sul calendario nella schermata delle prenotazioni.

#### **UC13. Creazione ticket**

**Attore:** Utente (Coinquilino);

**Exit Condition:** L'utente ha effettuato l'accesso e si trova nella propria home page;

**Flusso di eventi:**

1. L'utente seleziona si trova nella sezione "Segnalazioni";
2. L'utente clicca su "Nuovo Ticket";
3. Il sistema mostra la schermata di creazione di ticket da compilare;
4. L'utente inserisce categoria, descrizione e foto del guasto;
5. L'utente conferma e invia il ticket;
6. Il sistema assegna una priorità automatica e un codice al ticket;
7. Il sistema invia il ticket al supervisore;

**Exit Condition:** Il ticket dell'utente è stato creato correttamente e tracciato dal sistema;

**Flussi Alternativi:** L'utente ha inserito dei campi sbagliati e dunque cancella il ticket.

#### **UC14. Cancellazione Ticket**

**Attore:** Utente (Coinquilino);

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

**Flusso di eventi:**

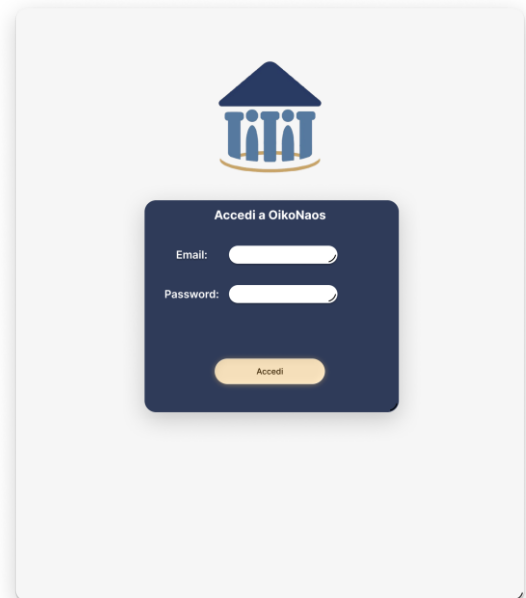
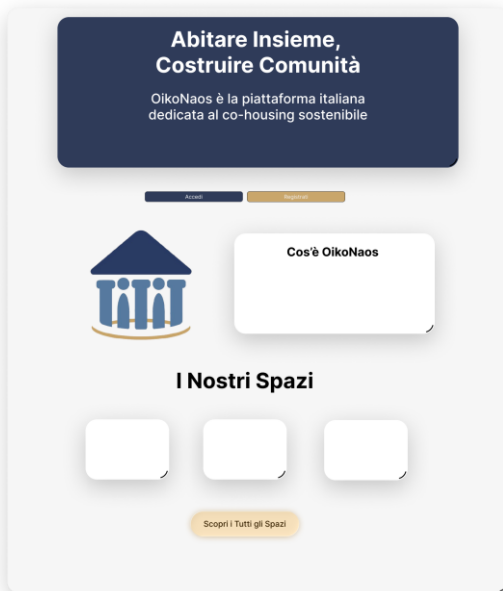
1. L'utente dopo aver effettuato l'autenticazione accede alla sezione "Ticket" nella propria pagina personale.
2. L'utente seleziona il ticket che intende cancellare.
3. Il sistema mostra all'utente i dettagli (categoria, descrizione, foto) del ticket
4. L'utente clicca sulla voce "Cancella ticket".
5. Il sistema apre una finestra di conferma con il messaggio "Sei sicuro di voler procedere alla cancellazione?".
6. L'utente conferma la sua scelta.
7. Il sistema rimuove il ticket dalla lista e gli assegna lo stato "Annullato".
8. Il sistema avvisa il supervisore incaricato della cancellazione.

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

### 4.5.1 Object model

### 4.5.2 Dynamic model

### 4.5.3 User interface



Riguarda i casi d'uso: UC09, UC01, UC02



Ambiente

Sala Studio ▼

Postazione

Postazione 4 ▼


Data

10/10/2025

Fascia Oraria

8:00-12:00 ▼

Prenota





Riguarda i casi d'uso: UC11, UC12

## Gestione spese

**Tassa trimestrale:**  
**€200 - Scad. 31/10/2025**

Paga ora

**Tassa trimestrale**  
**€200 - Scad. 31/07/2025**

Pagamento  
effettuato

**Tassa trimestrale**  
**€200 - Scad. 31/04/2025**

Pagamento  
effettuato

## Dettagli pagamento

### Pagamento tassa

Importo dovuto: €200

☐ Carta di credito

☐ Paypal

Procedi al pagamento

## Pagamento con Paypal

email

password

Conferma pagamento

## Pagamento con carta

Numero carta

Nome titolare

Data Scad.

CVV

Conferma pagamento

## 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](#)

Riguarda i casi d'uso: UC08

## 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)

Riguarda i casi d'uso: UC05

## 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"><li>Entity Object: rappresenta dati</li></ul>

	<p>persistenti (es. "Utente", "Prenotazione", "Ticket").</p> <ul style="list-style-type: none"> <li>• Boundary Object: interfaccia tra utente e sistema (es. "Pagina di login", "Modulo prenotazione").</li> <li>• Control Object: gestisce la logica applicativa (es. "PrenotazioneController").</li> </ul>
--	--