Problem Statement

Purpose

OikoNaos è una piattaforma volta alla gestione e organizzazione di una comunità di Co-housing a cura di un'Azienda Coordinatrice. La piattaforma offre diversi servizi utili per la comunità gestiti da diverse figure: Supervisionatore ticket critici e Monitoratori di spese, turni, eventi e risorse.
Ogni coinquilino - pagando un supplemento mensile e munito di credenziali apposite - ha la possibilità di monitorare le attività attraverso una bacheca e un calendario per la prenotazione di spazi comuni, di chiedere assistenza, gestire risorse condivise e spese comuni. Inoltre la comunità fa riferimento a un Rappresentante eletto dai residenti che vive nella Co-housing: raccoglie i feedback dei vari coinquilini e mantiene il contatto con l'azienda.

Audience

• Cliente:
  - Azienda Coordinatrice: organizzazione esterna che coordina le comunità di co-housing e richiede la piattaforma.

• Utenti finali:
  - Coinquilini: utenti che utilizzano la piattaforma per gestire gli spazi comuni, i beni e le richieste di assistenza.
  - Rappresentante dei residenti: eletto democraticamente dai residenti, rappresenta l'azienda e dunque funge da intermediario tra gli utenti finali.

• Project Management e Analisti di Sistema: Gruppo OikoNaos di sviluppo, si occupa di raccogliere le esigenze, progettare e implementare.
  - Componenti: Luigi Potestà, Maria Antonietta Ruggiero, Giulia Buonafine, Mario Sinopoli.

Outline

1. Problem domain

OikoNaos migliora la qualità e la gestione delle co-housing concretizzando in un'unica piattaforma tutte le attività della comunità. Il calendario condiviso, con la possibilità di aggiungere prenotazioni, riduce i conflitti su spazi e risorse comuni; il sistema di invio di ticket traccia le richieste di manutenzione attraverso campi quali priorità, stato per tempi di risposta più rapidi; la bacheca eventi invece favorisce comunicazioni chiare.
La gestione delle spese offre ripartizioni sicure tra i residenti con storici e resoconti trasparenti, garantendo così la fiducia reciproca. Ruoli e permessi (coinquilino, rappresentante, coordinatore) specificano responsabilità e ordine. Le notifiche e i promemoria aiutano l'adesione ai turni e la partecipazione alle iniziative che coinvolgono la comunità. Tutto questo permette un migliore coordinamento e senso di comunità, con benefici garantiti da tempi di risoluzione brevi, equità nelle spese e serenità dei residenti.

2. Scenarios

1. Prenotazione spazio comune

• Attore: Coinquilino;
• Obiettivo: prenotare la sala studio;
• Interazione: il coinquilino apre il calendario, seleziona fascia oraria conforme alle regole, conferma; il sistema notifica eventuali conflitti (prenotazioni già confermate) e propone alternative;
• Esito: prenotazione registrata, notifica ai residenti interessati.

2. Segnalazione guasto (ticket)

• Attore: Coinquilino;

• Obiettivo: segnalare lavatrice rotta;
• Interazione: apre “Nuovo ticket”, inserisce categoria, descrizione, foto; il sistema assegna priorità e inoltra al supervisore.
• Esito: ticket creato, stato tracciato con aggiornamenti automatici.

3. Approvazione turni
• Attore: Rappresentante dei residenti.
• Obiettivo: approvare il calendario pulizie.
• Interazione: visualizza proposta generata dal sistema, modifica assegnazioni critiche, pubblica.
• Esito: turni confermati, notifiche e promemoria ai coinquilini.

3. Functional requirements

1. Autenticazione: login/logout, recupero password, gestione profili;
2. Calendario prenotazioni: creazione/modifica/cancellazione prenotazioni degli spazi comuni, gestione dei conflitti relativi alle prenotazioni con suggerimenti su orari liberi;
3. Ticket di manutenzione: apertura con categoria/priorità/allegati, stato del ticket(aperto, in lavorazione, risolto), cronologia.
4. Bacheca eventi/annunci: pubblicazione, iscrizione a eventi, commenti, allegati; sincronizzazione della bacheca con calendario.
5. Gestione risorse condivise: richiesta/assegnazione di beni (es. attrezzi, lavanderie, giochi), regole d’uso e tracciamento di date di assegnazione e di restituzioni.
6. Gestione spese: registrazione criteri di ripartizione, saldi individuali, rendiconti ed esportazione di ricevute.
7. Notifiche e promemoria: eventi imminenti, scadenze spese, aggiornamenti ticket/turni (email/in app).
8. Ricerca e filtri su prenotazioni, ticket, eventi e spese.
9. Mappa interattiva della comunità con localizzazione di spazi/risorse.

4. Nonfunctional requirements

• Usabilità: semplice da imparare e da usare. Pronto all’uso in meno di 10 minuti, senza manuali.
• Affidabilità: servizio disponibile almeno al 99,5% nelle ore operative. In caso di errore non si perdono dati. Backup giornaliero.
• Prestazioni: pagine che si caricano in < 1 secondo; conferma di prenotazioni e ticket in < 2 secondi; supporto per ≥ 100 utenti contemporanei.
• Sicurezza: accessi gestiti per ruoli e permessi; collegamenti protetti (HTTPS); password salvate in modo sicuro; difese contro le vulnerabilità più comuni; log di sicurezza.
• Privacy/Conformità: pieno rispetto del GDPR (Regolamento generale sulla protezione dei dati personali); regole chiare sulla conservazione dei dati e sul diritto alla cancellazione
• Manutenibilità/Supportabilità: struttura modulare, test automatici, monitoraggio continuo e documentazione completa per API e amministratori.
• Portabilità: funzionamento su Tomcat 10.1, Java 17 e MySQL 8; compatibile con i browser principali come Chrome, Firefox, Edge, Safari.
• Accessibilità: layout responsive, testi leggibili con contrasti adeguati, navigazione coerente, etichette e messaggi chiari.

5. Target environment

• Server/applicazione: Mac OS e Windows, Java 17, Tomcat 10.1 (Jakarta EE), MySQL 8.
• Rete e sicurezza: Sicurezza garantita da connessioni HTTPS, uso di porte standard e procedure di backup giornaliero del database.
• Browser supportati: ultime 2 versioni di Chrome, Firefox, Edge, Safari; layout responsive.
• Capacità minima attesa (test di sistema): ≥ 100 utenti concorrenti, pagine tipiche < 1 s, conferma prenotazione/ticket < 2 s.

• Dati di prova: Dataset anonimo usato per testare il sistema e le funzionalità principali.

6. Deliverable & deadlines

1. Problem Statement: Documento che descrive il dominio del problema, gli stakeholder, gli obiettivi e i confini del sistema. - 14 ottobre 2025
2. Requisiti e casi d'uso: Identificazione dei requisiti funzionali e non funzionali, definizione degli attori e dei principali use case. - 28 ottobre 2025
3. Requirements Analysis Document (RAD): Analisi approfondita dei requisiti, modellazione UML, definizione delle relazioni tra use case e classi. - 11 novembre 2025
4. System Design Document (SDD): Progettazione dell'architettura del sistema e dei principali moduli software. - 25 novembre 2025
5. Specifica delle interfacce: Descrizione tecnica delle interfacce dei moduli del sottosistema da implementare (parte dell'Object Design Document). - 16 dicembre 2025
6. Piano di test e specifica dei casi di test: Definizione dei test di sistema e dei casi di test per il sottosistema implementato. - 16 dicembre 2025