

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA ELETTRICA E
TECNOLOGIE DELL'INFORMAZIONE

CORSO DI LAUREA IN INFORMATICA
INGEGNERIA DEL SOFTWARE

DietiEstetes25

Piattaforma per la gestione di agenzie immobiliari

Raimondo Morosini - N86003839

Roberto Spena - N86003552

Lorenzo Sepe - N86003622

Anno Accademico 2024/2025

Questa pagina è stata lasciata intenzionalmente bianca

Contents

1 Revisioni	4
2 Requisiti Software	5
2.1 Glossario	8
2.2 Modellazione dei Casi d’Uso	9
2.3 Analisi Target Utenti	14
2.4 Descrizione Requisiti	18
2.4.1 Requisiti non funzionali	18
2.4.2 Requisiti di dominio	20
2.5 Descrizione Testuale strutturata del Sistema	22
2.5.1 Cockburn: nuovo annuncio	22
2.5.2 Cockburn: Attivazione e Disattivazione categorie notifiche	26
2.5.3 Cockburn: Effettua controproposta di un’offerta	31
2.5.4 Cockburn: Registra nuovo agente	33
2.6 Mock up	34
2.6.1 Caso d’Uso: Aggiungi un Annuncio Immobiliare	35
2.6.2 Caso d’Uso: Attivazione e Disattivazione Notifiche	48
2.6.3 Caso d’Uso: Fare una controproposta di un offerta	65
2.6.4 Caso d’Uso: Registra nuovo agente	71
3 Design del Sistema	76
3.1 Descrizione dell’architettura proposta	76
3.2 Descrizione e motivazione delle scelte tecnologiche adottate	77
3.3 Descrizione dello schema per la persistenza dati	77
4 Processo di sviluppo	79
5 Testing e valutazione dell’usabilità.	80
5.1 Progettazione e implementazione dei test unitari	80
5.1.1 addDipendente(DipendenteRequest request, String aliasAgenzia)	80
5.1.2 utentiInteressati(CriteriDiRicercaUtenti request)	88
5.2 Test di usabilità	93

Chapter 1

Revisioni

Table 1.1: Tabella Revisioni documento

Data	Versione	Autore/i	Descrizione
22/01/25	0.1	R. Spena, R. Morosini, L. Sepe	Prima Stesura
24/01/25	0.2	R. Spena, R. Morosini, L. Sepe	Aggiunte Personas
27/01/25	0.3	R. Spena, R. Morosini, L. Sepe	Aggiunte Tabelle Cockburn
29/01/25	0.4	L. Sepe	Completate Tabelle Cockburn Inizio Lavoro su MockUp
31/01/25	0.5	R. Spena, R. Morosini, L. Sepe	Revisioni minori
19/02/25	0.6	L. Sepe	Revisione Tabelle Cockburn
21/02/25	0.7	R. Spena	Inserita grafica tabelle dei requisiti
21/02/25	0.7	R. Morosini	Inserita descrizione ai mockup delle estensioni

Chapter 2

Requisiti Software

Introduzione

In questa sezione scriveremo un testo contenendo cosa dovrà fare il nostro software così da usarlo come base

Obiettivo

Il sistema deve fornire un'area amministrativa e un'area Cliente, con funzionalità specifiche per i diversi ruoli utente e per la gestione degli immobili. Le funzionalità sono state riformulate in un linguaggio tecnico per favorire chiarezza e comprensione.

Area Amministrativa

L'area amministrativa consente la gestione degli account e degli immobili. I ruoli previsti sono:

- Amministratore
- Amministratore di supporto
- Agenti immobiliari

Accesso all'Area Amministrativa

Amministratore:

L'accesso avviene dopo la registrazione di un'agenzia immobiliare tramite richiesta al sistema. Una volta approvata, vengono fornite credenziali predefinite per l'accesso.

Amministratore di supporto:

Creati dall'Amministratore principale e associati alla stessa agenzia immobiliare. Le credenziali sono generate dall'Amministratore.

Agenti immobiliari:

Creati dall'Amministratore o dall'Amministratore di supporto. Le credenziali sono anch'esse generate dall'Amministratore.

Area Cliente

L'area Cliente è destinata agli utenti registrati che possono:

- **Ricercare immobili:**

- Ricerca avanzata con filtro per posizione geografica (selezionando un punto e un raggio su una mappa).
- Visualizzazione degli immobili in lista o su mappa.
- Filtri aggiuntivi per caratteristiche dell'immobile e del contratto (es. metratura, numero di stanze, ecc.).

- **Visualizzare immobili completi di dettagli:**

- Inclusi dati come metratura, stanze, servizi, ecc.
- Informazioni sui luoghi di interesse vicini (es. scuole, parchi, metro) utilizzando l'integrazione con Geopify.

- **Proporre offerte:**

- L'utente registrato può inviare una proposta sull'immobile, che verrà gestita dall'agente immobiliare.

- **Gestire notifiche push:**

- Notifiche categorizzate (promozionali, private, ecc.).
- Possibilità di disattivare categorie specifiche.

Funzionalità dei Ruoli

Amministratore

- Creare account per Amministratori di supporto e agenti immobiliari.
- Modificare e cancellare definitivamente immobili creati dagli agenti.
- Modificare la propria password.

Amministratore di supporto

Nota: Non sono stati definiti requisiti specifici per questo ruolo. Si propone di assegnare le stesse funzionalità dell'Amministratore, con eventuali limitazioni future da specificare.

Agenti immobiliari

- Creare, modificare e cancellare immobili da loro creati.
- Visualizzare le proposte ricevute sugli immobili.
- Registrare proposte esterne.
- Accettare o rifiutare proposte.
- Modificare la propria password.

2.1 Glossario

Amministratore: Ruolo all'interno del sistema che gode di accesso a tutti gli Endpoint, autorizzazioni e metodi.

Agente: Ruolo all'interno del sistema che è autorizzato a creare, modificare ed eliminare gli Annunci per gli immobili gestiti dall'Agenzia.

Agenzia: Entità caratterizzata dal suo Amministratore e Agenti in impiego. Alla sua creazione nella base dati verrà automaticamente creato un Amministratore.

Annunci: Entità caratterizzata dal un Immobile e il tipo di Contratto.

2.2 Modellazione dei Casi d’Uso

Dopo aver definito i requisiti funzionali e i vincoli del sistema, questa sezione illustra i casi d’uso (use case), che descrivono in dettaglio le principali interazioni tra gli utenti e il sistema per il raggiungimento degli obiettivi. Nella nostra analisi preliminare, abbiamo individuato quattro attori principali che interagiscono con il sistema: Guest, Utente, Agente e Amministratore.

- Guest: rappresenta l’utente non autenticato, che accede al sito senza effettuare il login. Questo attore ha accesso limitato, ma può comunque compiere alcune azioni di base, come effettuare ricerche sugli Immobili e visualizzare informazioni pubbliche su di essi.
- User: questo attore è un utente autenticato con un account registrato. Rispetto al Guest, ha funzionalità aggiuntive, come fare offerte sugli Immobili, monitorare il proprio storico delle ricerche effettuate e ricevere notifiche in base alle preferenze che ha esibito. Queste notifiche possono essere ulteriormente personalizzata nelle impostazioni con dei filtri.
- Agente: rappresenta l’utente che può creare e modificare gli Annunci e gli Immobili.
- Amministratore: l’attore che gestisce l’agenzia immobiliare nella sua interezza, avendo il potere di appuntare sia Agenti che altri Amministratori. Inoltre l’amministratore può curare il catalogo di Annunci, modificando o eliminando elementi dalla lista.

Per ciascun attore, sono stati definiti i casi d’uso relativi, illustrati nei diagrammi seguenti. Questi schemi mostrano le modalità principali d’interazione con il sistema e forniscono una visione chiara, delineando le possibilità di interazione per ciascun tipo di utente. In questo modo, è possibile comprendere come le funzionalità definite nei requisiti trovano applicazione pratica all’interno delle azioni eseguibili da ciascun attore, evidenziando i passaggi chiave delle interazioni.

Diagrammi Casi d'uso

Figure 2.1: Casi d'uso Guest

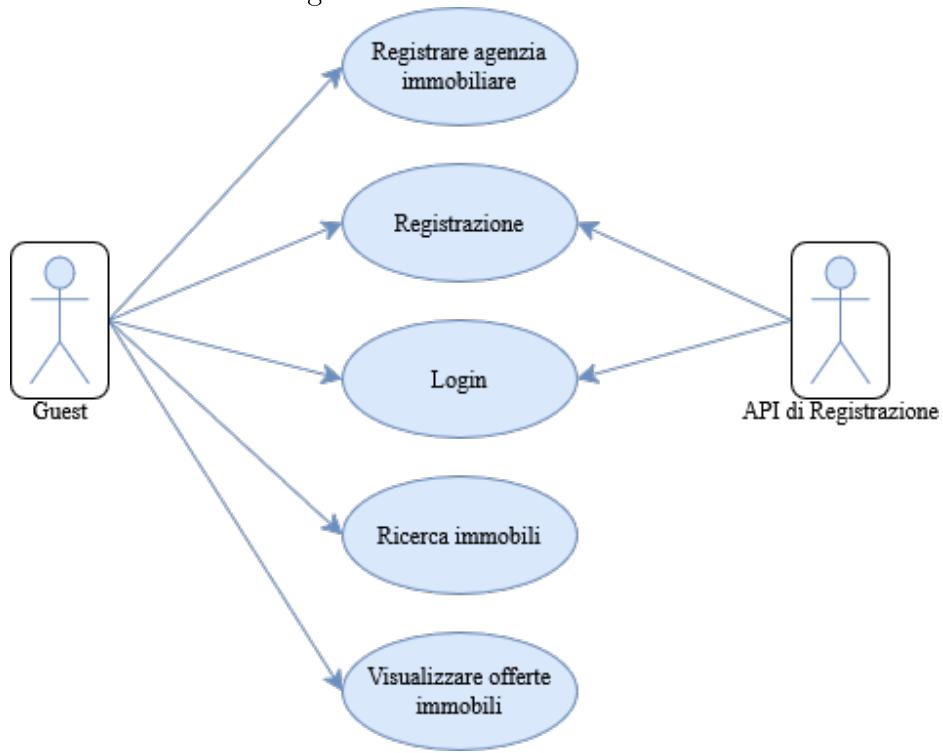


Figure 2.2: Casi d'uso Utente

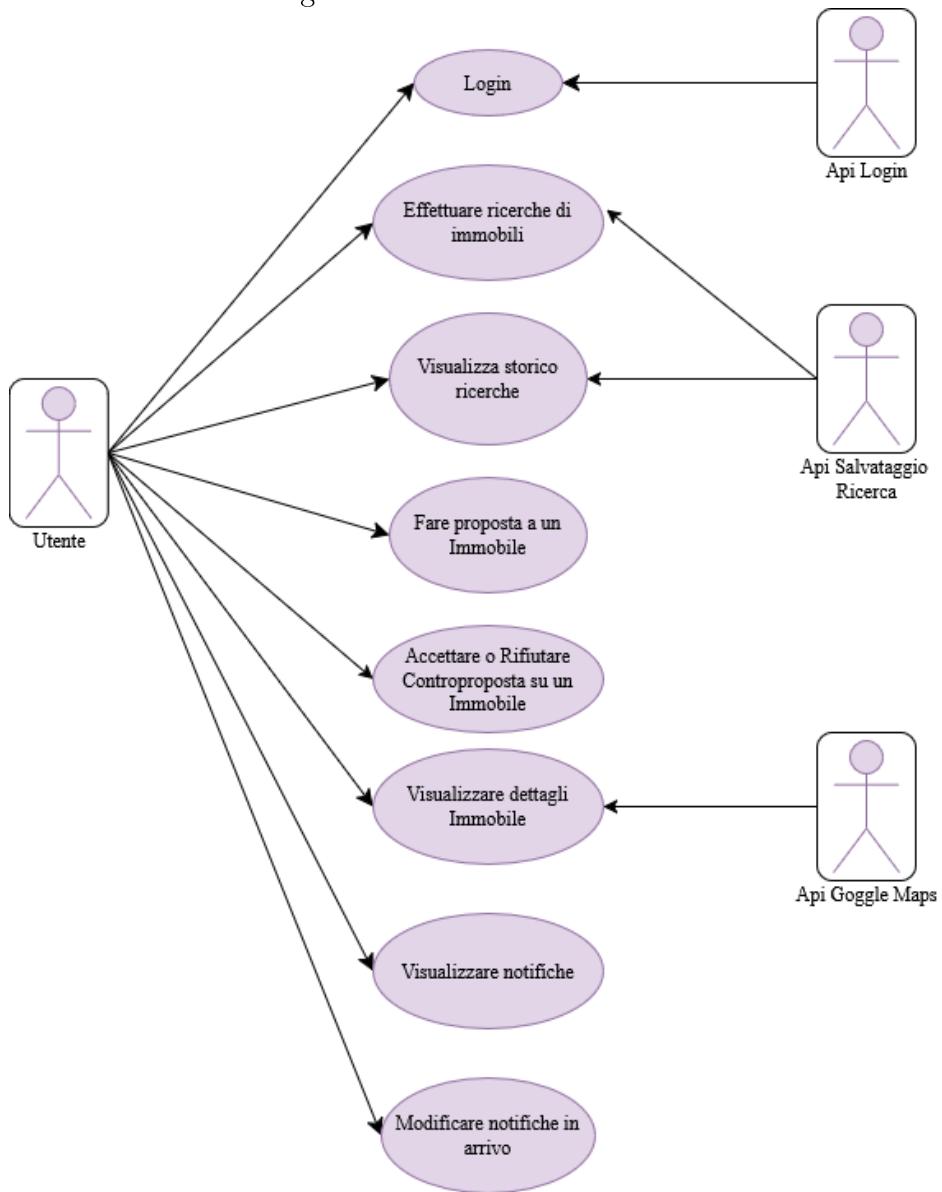


Figure 2.3: Casi d'uso Agente

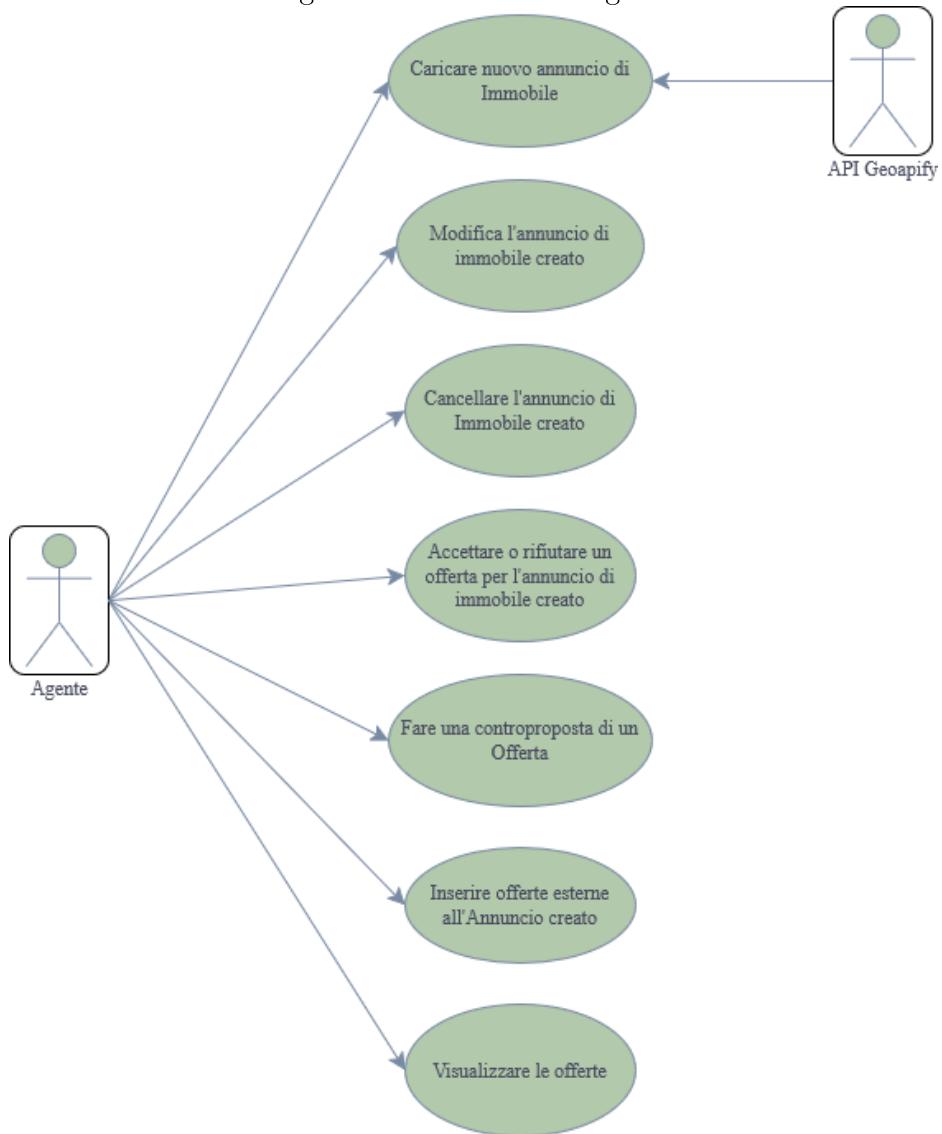
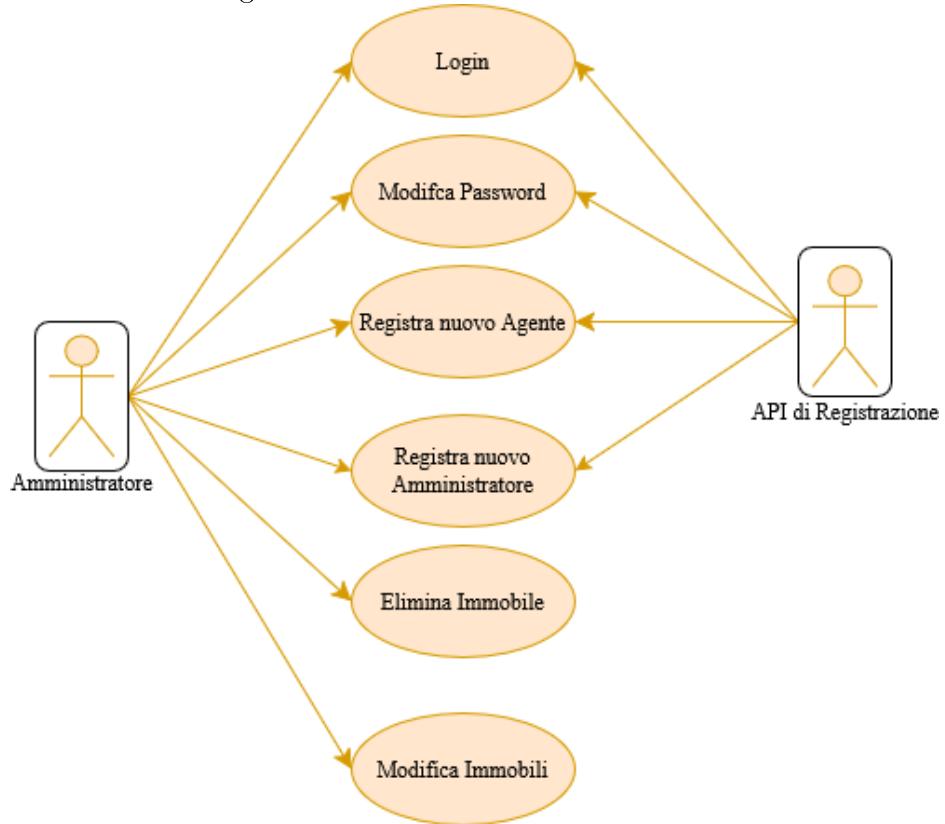


Figure 2.4: Casi d'uso Amministratore



2.3 Analisi Target Utenti

Quando si crea un software, è importante ricordare che non è necessario soddisfare ogni singolo utente possibile, ma solo la categoria di utenti che usufruiranno del nostro sistema più frequentemente.

Questa decisione ci permette di concentrare la nostra analisi e ridurre funzioni che, al momento del rilascio del software al committente, sono superflue.

Uno strumento utile per fare questa analisi è la Persona, un modello di utente ideale che può essere basato su ricerche di mercato oppure da input dai committenti stessi.

Personas Individuate

Per creare Personas è stato adottato un approccio basato su ipotesi e ricerche secondarie.

Abbiamo così definito il contesto, identificando il pubblico target, i loro obiettivi e le situazioni in cui potrebbero utilizzare il prodotto. Si possono utilizzare archetipi comuni del dominio di riferimento per immaginare utenti tipo, discutendo possibili bisogni, obiettivi e preferenze degli utenti.

Abbiamo creato le Personas a partire da una template predefinita e usando Figma per il design e l'aggiornamento di esse in corso d'opera.

Figure 2.5: Proprietario Agenzia immobiliare

The persona card for Marco Bianchi is divided into three main sections: Objectives, Interests, and Bio.

Marco Bianchi

Obiettivi

- Gestire l'agenzia Immobiliare in un'unica piattaforma.
- Centralizzare la gestione dei collaboratori, tenendo traccia delle loro prestazioni e semplificando lassegnazione degli incarichi.
- Ottimizzare il controllo degli immobili inseriti, monitorare le offerte dei possibili immobili.

Interessi

- Viaggiare
- Assiduo frequentatore di Seminari di Aggiornamento, eventi di Networking e vari eventi per far crescere il suo business
- Trading.

Bio

Marco Bianchi è un imprenditore di 45 anni, proprietario di un'agenzia Immobiliare con sede a Roma. È esperto del settore Immobiliare e apprezza l'efficienza digitale per migliorare la gestione del suo team e monitorare i progressi delle vendite. Marco cerca una piattaforma che centralizzi tutte le attività dell'agenzia, dal coordinamento dei collaboratori alla supervisione degli annunci Immobiliari.

Figure 2.6: Agente immobiliare



Aldo Imparato

Età: 29
Professione: Agente Immobiliare
Luogo: Milano, Italia
Esperienza: 3 anni nel settore

Obiettivi

- Estremamente ambizioso, vuole ottenere il più alto bonus di fine anno nell'agenzia.
- Punta prima o poi a diventare un imprenditore indipendente con il capitale accumulato dalle sue vendite.
- Cercherà sempre di convincere i clienti che hanno bisogno del meglio, incurante del budget.

Interessi

- Corse di auto.
- Mercati Immobiliari.
- Frequentatore di palestre.

Bio

Aldo ha uno spirto competitivo che lo ha aiutato in molti aspetti della sua vita, cogliendo occasione di una carriera nel settore Immobiliare come una sfida.
L'unico ambito che non scaturisce il suo interesse è quello sociale, inemicandosi molti dei suoi colleghi con le sue tattiche sleali.

Figure 2.7: Manager di azienda tecnologica



Giorgia Esposito

Età: 37
Professione: Manager di un'azienda tecnologica
Luogo: Napoli, Italia

Obiettivi

- Trovare un ufficio che favorisca la produttività e l'innovazione. Trovarsi in una zona strategica, ben collegata ai trasporti pubblici.
- Investire in uno spazio che rappresenti un valore a lungo termine per l'azienda.

Interessi

- Disegno.
- Sperimentare in cucina.
- Partecipazione a corsi per acquisire nuove competenze.

Bio

Giorgia è una professionista che lavora nel settore tecnologico. Ha bisogno di un ufficio spazioso con una buona connessione Internet, accesso a mezzi pubblici e un ambiente che favorisca la produttività e l'innovazione.
Ama leggere libri di economia e partecipare a eventi aziendali.

Figure 2.8: Padre di famiglia



Giovanni Rossi

Età: 42
Professione: Ingegnere civile
Luogo: Milano, Italia
Stato Sociale: sposato, 2 figli (8 e 5 anni)

Obiettivi

- Acquistare una casa con 3 camere da letto e giardino.
- Vivere in una zona ben collegata e sicura.
- Scegliere una proprietà che rappresenti un investimento a lungo termine.

Interessi

- Attività all'aperto con la famiglia.
- Costruire modelli in scala di linee ferroviarie.
- Organizzare barbecue con i colleghi.

Bio

Giovanni è un padre di famiglia impegnato e motivato. Vuole migliorare la qualità della vita della sua famiglia acquistando una casa più grande, con uno spazio all'aperto per i suoi figli. Tende ad accontentare la moglie per ogni piccola decisione, e fa fatica a ritagliare tempo per se stesso.

Tratti Caratteristici

Analizzando queste Personas possiamo notare dei tratti che possiamo usare nella nostra applicazione:

- L'Agente- Aldo Imparato: questa categoria di utenti è parte dello staff dell'Agenzia immobiliare e quindi ha accesso al lato di amministrazione degli Immobili, e potrebbe volere una sezione nella sa pagina personale che gli mostra tutti gli Immobili a suo carico.
- Il Padre di Famiglia - Giovanni Rossi: l'archetipo che incarna il Utente medio che ha vaghe idee su cosa vuole e si affida all'agente immobiliare. Dobbiamo quindi tener conto che le informazioni visibili a questo tipo di utenti deve essere chiara e concisa, come ad esempio le etichette che indicano punti di interesse come parchi e scuole.
- Il Manager di Azienda Tecnologica - Giorgia Esposito: l'utente che sta cercando qualcosa di ben definito, ma l'agenzia non offre Immobili con le caratteristiche cercate. Per questo tipo di persone offriremo un servizio di iscrizione ai tag di interesse per ricevere notifiche quando un immobile che rispecchia tali desideri viene messo in vendita.
- Il Proprietario dell'Agenzia Immobiliare - Marco Bianchi: gli amministratori del sistema che gestiscono dipendenti e il catalogo di Immobili, questo tipo di utente deve gestire multipli dati personali e sensibili. Quando crea un Agente o Amministratore, le nuove informazioni verranno generate dal sistema e inviate tramite un servizio di posta elettronica.

2.4 Descrizione Requisiti

2.4.1 Requisiti non funzionali

Sicurezza

ID	Descrizione
NF-S1	Implementazione di protocolli standard di sicurezza (es. HTTPS per il trasporto sicuro dei dati).
NF-S2	Le credenziali devono essere salvate in modo sicuro, utilizzando tecniche di hashing sicuro.
NF-S3	Le richieste alle REST API devono essere autenticate utilizzando JWT (JSON Web Tokens), per garantire che solo gli utenti autenticati possano accedere alle risorse protette.

Interfaccia utente

ID	Descrizione
NF-UI11	L'interfaccia utente deve adattarsi automaticamente alla risoluzione dello schermo dell'utente, garantendo un'esperienza di utilizzo ottimale su dispositivi mobili e desktop.

Prestazioni

ID	Descrizione
NF-P1	La ricerca e la visualizzazione degli immobili devono avere un tempo di risposta inferiore a 2 secondi, con un database contenente almeno 10.000 record.

Manutenibilità e scalabilità

ID	Descrizione
NF-MS1	Il sistema deve essere progettato per supportare l'aggiunta di nuove tipologie di immobili e contratti (ad esempio, affitti brevi) senza modificare l'architettura esistente.

2.4.2 Requisiti di dominio

Gestione dell'agenzia immobiliare

ID	Descrizione
D-A1	Un'agenzia immobiliare è composta da un fondatore e da una lista di dipendenti, ciascuno dei quali ricopre il ruolo di amministratore o agente immobiliare.
D-A2	La registrazione di un'agenzia immobiliare comporta la generazione automatica di un amministratore con il ruolo di fondatore.
D-A3	Alla creazione di un account amministratore vengono assegnate credenziali predefinite che possono essere modificate in seguito.
D-A4	Le credenziali di un amministratore includono uno username, costruito seguendo il formato [nome][cognome]@DIETI25.com, e una password scelta dall'utente.

Gestione degli utenti

ID	Descrizione
D-U1	Un guest può registrarsi al sistema fornendo un'email valida e una password, diventando così un utente.
D-U2	Un guest può registrarsi o accedere tramite l'uso di API di terze parti.

Gestione degli immobili

ID	Descrizione
D-G1	Ogni immobile è descritto da una serie di dettagli obbligatori, tra cui: foto, descrizione, prezzo, dimensioni, indirizzo, numero di stanze, piano, presenza di ascensore, classe energetica e ulteriori servizi (es. portineria, climatizzazione).
D-G2	Ogni immobile è associato a una delle seguenti tipologie contrattuali: "vendita" o "affitto".
D-G3	Ogni immobile, al momento della creazione, ha associata una lista di punti di riferimento generata automaticamente utilizzando il servizio GEOAPIFY.

Ricerca e visualizzazione

ID	Descrizione
D-R1	La ricerca degli immobili consente di effettuare una selezione geografica basata su un punto centrale e un raggio, garantendo una precisione pari o superiore al 95%.
D-R2	La ricerca avanzata degli immobili permette di utilizzare parametri multipli, tra cui tipologia di inserzione, prezzo minimo e massimo, numero di stanze, classe energetica e area geografica tramite mappa interattiva.
D-R3	Gli immobili ricercati possono essere visualizzati su una mappa interattiva.

Gestione delle offerte

ID	Descrizione
D-O1	Le inserzioni degli immobili possono ricevere offerte da parte degli utenti, composte da un prezzo proposto e dalle credenziali dell'offerente. Le offerte sono visibili a tutti gli utenti.

Gestione delle notifiche

ID	Descrizione
D-N1	Il sistema di notifiche consente di inviare avvisi personalizzati agli utenti, categorizzati in base a eventi come: nuove proprietà in linea con ricerche precedenti, conferme o rifiuti di proposte, e messaggi promozionali.
D-N2	Le notifiche promozionali sono composte da un contenuto in formato RICH TEXT, definito dagli amministratori, e sono visibili esclusivamente agli utenti iscritti alla relativa newsletter.

2.5 Descrizione Testuale strutturata del Sistema

In questa sezione faremo un analisi più approfondita di alcuni dei casi d'uso campione del sistema, per esporre il nostro ragionamento per modellare e sviluppare il Software.

Questa descrizione ci servirà per pensare a che tipo di interazioni con l'Utente sono necessarie, i passaggi da effettuare dal sistema in risposta alle richieste ricevute e I casi d'uso scelti per questo scopo sono la Creazione di un nuovo Annuncio di un Immobile e Modifica notifiche in Arrivo.

2.5.1 Cockburn: nuovo annuncio

Table 2.1: Creazione nuovo Annuncio

Use Case 1	Creazione Nuovo Annuncio per Immobile			
Goal In Context	Inserimento nel Catalogo dell'Agenzia Immobiliare un nuovo Annuncio per Immobile.			
Preconditions	<ul style="list-style-type: none"> - Login effettuato con account con ruolo Agente o Amministratore. - Si trova nella pagina di Creazione Annunci. 			
Success End Conditions	Aggiunta in Catalogo dell'Immobile e Salvataggio nel Database.			
Primary Actor	Agente			
Trigger	Agente preme il bottone Aggiungi Immobile nella Schermata del Catalogo.			
Descrizione	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Step n.</th> <th>Agente</th> <th>Sistema</th> </tr> </thead> </table>	Step n.	Agente	Sistema
Step n.	Agente	Sistema		

Table 2.1: Creazione nuovo Annuncio

Use Case 1	Creazione Nuovo Annuncio per Immobile	
Scenario Principale	1	<i>Controlla che non ci siano operazioni in sospeso</i>
	2	<i>Mostra la pagina di creazione Annunci.</i>
	3	Compila 3 campi: - Titolo Annuncio. - Tipologia contratto. Tipologia Immobile.
	4	<i>In base ai dati inseriti mostra il form adatto per la creazione.</i>
	5	Compila Fom Annuncio
	6	Click Bottone Conferma.
	7	<i>Mostra Riepilogo Dati Inseriti.</i>
	8	Click Bottone Pubblica.
	9	<i>Validazione Dei Dati</i>
	10	<i>Salvataggio in Database</i>
	11	<i>Mostra Popup Operazione completata con successo.</i>
	12	<i>Use Case terminato con Successo.</i>

Table 2.1: Creazione nuovo Annuncio

Use Case 1	Creazione Nuovo Annuncio per Immobile		
Extension	Step n.	Utente	Sistema
Sistema riconosce che c'è un Annuncio in creazione sospeso. Utente vuole creare un Nuovo Annuncio	A.2		<i>Vede che c'è un operazione di creazione in sospeso</i>
	A.3		<i>Mostra Popup chiedendo se vuole continuare l'operazione o creare un nuovo Annuncio</i>
	A.4	Click su Nuovo Annuncio	
	A.5		<i>Mostra Dialog per avvisare che i dati dell'Annuncio in sospeso verranno persi</i>
	A.6	Click Conferma	
	A.7		<i>Vai al passo 2 dello Scenario Principale</i>
	B.2		<i>Vede che c'è un operazione di creazione in sospeso</i>
Sistema riconosce che c'è un Annuncio in creazione sospeso. Utente vuole continuare operazione sospesa	B.3		<i>Mostra Popup chiedendo se vuole continuare l'operazione o creare un nuovo Annuncio</i>
	B.4	Click su Continua Operazione	
	B.5		<i>Mostra la pagina di creazione caricando i dati inseriti nell'annuncio sospeso</i>
	B.6		<i>Vai al passo 4 dello Scenario Principale</i>
	C.9		<i>Mostra Messaggio di Errore che avverte che l'operazione non è andata buon fine e che i dati sono stati preservati in locale.</i>
	C.10		<i>Salva dati in locale.</i>

Il Sistema è Offline
al momento della
Pubblicazione oppure
non è Collegato a
Use Case 1
Internet

Table 2.1: Creazione nuovo Annuncio

Creazione Nuovo Annuncio per Immobile		
	C.11	<i>Chiude la schermata. Use Case Terminato in Fallimento.</i>
Dati Inseriti non validi	D.10	Evidenzia campi del form non riempiti/validi
	D.11	Vai al passo 5 dello Scenario Principale

2.5.2 Cockburn: Attivazione e Disattivazione categorie notifiche

Table 2.2:

Use Case 2	Modifica notifiche in Arrivo		
Goal In Context	Cambia lo stato di notiche a scelta		
Preconditions	<ul style="list-style-type: none"> - Login effettuato con account con ruolo Utente. - Si trova nella pagina di Visualizzazione notifiche 		
Success End Conditions	Lo stato di abilitazione di notifica viene modificato		
Primary Actor	Utente		
Trigger	Utente preme il bottone Gestisci Notifiche		
Descrizione	Step n.	Utente	Sistema
	1		<i>Mostra pagina delle notifiche</i>
	2	Click Bottone per modificare stato notifica	
	3		<i>Mostra una Popup con elenco di tutte le categorie che possono essere Modificate</i>
	4	Click su la casella/e relativa/e alla categoria/e da Modificare	
	5	Click Bottone Conferma	
	6		<i>Mostra Dialog con un messaggio che avverte l'Utente che non riceverà più nuove Notifiche relative alla categoria selezionata</i>
	7	Click Bottone Conferma nella Dialog	
	8		<i>Modifica Stato Notifica in Database</i>

Table 2.2:

Use Case 2	Modifica notifiche in Arrivo		
	9		<i>Feedback visivo del cambio di visibilità</i>
	10		<i>Use Case terminato con Successo</i>

Table 2.2:

Use Case 2	Modifica notifiche in Arrivo		
Extension	Step n.	Utente	Sistema
Utente vuole disabilitare una categoria di notifiche a partire da una notifica ricevuta	A.2	click su notifica ricevuta	
	A.3		<i>Mostra il contenuto della notifica</i>
	A.4	Click Bottone Disattiva Notifica	
	A.5		<i>Bottone Disattiva Notifica diventa Bottone Attiva Notifica</i>
	A.6		<i>Vai allo step 6 dello Scenario Principale</i>
	B.6		<i>Il sistema nota che non ci sono state modifiche</i>
Utente non modifica nessuna categoria durante lo Scenario Principale	B.7		<i>Use case Terminato con Successo</i>
	C.2	Utente preme Bottone Disabilita dalla lista delle categorie Visibili	
	C.3		<i>Bottone Disabilita diventa Bottone Abilita</i>
Utente vuole disabilitare categoria dalla lista delle Categorie Attive	C.4		<i>Vai al Passo 6 dello Scenario Principale</i>
	D.2	Utente Preme Bottone Abilita dalla lista delle Categorie disabilitate	
	D.3		<i>Bottone Abilita diventa Bottone Disabilita</i>
Utente vuole abilitare categoria dalla lista delle Categorie Disabilitate	D.4		<i>Vai al Passo 8 dello Scenario Principale</i>

Table 2.2:

Use Case 2	Modifica notifiche in Arrivo		
Il sistema è Offline al click del bottone dell'Utente oppure non è collegato a Internet.	E.7		<i>Mostra messaggio di Errore il quale avverte che l'operazione non è andata a buon fine.</i>
	E.8		<i>Use Case Terminato in Fallimento.</i>

Table 2.2:

Use Case 2	Modifica notifiche in Arrivo		
Utente vuole abilitare una categoria di notifiche a partire da una notifica ricevuta	F.2	Click su notifica ricevuta	
	F.3		<i>Mostra il contenuto della notifica</i>
	F.4	Click Bottone Attiva Notifica	
	F.5		<i>Bottone Attiva Notifiche diventa Bottone Disabilita Notifiche</i>
	F.6		<i>Vai al Passo 8 dello Scenario Principale</i>

2.5.3 Cockburn: Effettua controproposta di un'offerta

Use case 3	Fare una controproposta di un'offerta		
Goal in Context	Proporre		
Preconditions	<ul style="list-style-type: none"> -Essere loggati come agente. -Aver ricevuto almeno un offerta su un immobile pubblicato. 		
Success End Condition	Controproposta inserita nel sistema.		
DESCRIPTION	Step	Agente	Sistema
Scenario principale	1		Mostra lista annunci immobili pubblicati
	2	Clicca sull'annuncio di interesse	
	3		Mostra lista delle offerte ricevute relativo all'annuncio selezionato
	4	Clicca su "controproposta" su un offerta di un utente	
	5		Mostra finestra controproposta
	6	Scrive il prezzo proposto	
	7	Clicca "Invia controproposta"	
	8		Mostra caricamento
	9		Salvataggio della controproposta
	10		Invio notifica all'utente interessato
	11		Use case terminato con successo

Use case 3	Fare una controproposta di un'offerta		
EXTENSIONS A	Step	Agente	Sistema
L'agente omette il prezzo della controfferta	6.A	Clicca "Invia controproposta"	
	7.A		Mostra messaggio di errore sotto all'input della controproposta che avverte del campo vuoto.
	8.A		Ritorna al punto 6 dello scenario principale
EXTENSIONS B	Step	Agente	Sistema
L'agente inserisce un prezzo inferiore al prezzo proposto dall'utente	7.B	Clicca "Invia controproposta"	
	8.B		Mostra messaggio di errore sotto all'input della controproposta che avverte che il prezzo inserito è inferiore al prezzo proposta dall'utente
	9.B		Ritorna al punto 6 dello scenario principale
EXTENSIONS C	Step	Agente	Sistema
Il salvataggio della controproposta fallisce per problemi di connessione	7.C	Clicca "Invia controproposta"	
	8.C		Mostra caricamento
	9.C		Mostra allert che avverte che l'operazione non è andata a buon fine e consiglia all'utente di riprovare in un secondo momento
	10.C		Use case fallito

2.5.4 Cockburn: Registra nuovo agente

Use case 4	Registra nuovo agente		
Goal in Context	Inserire un nuovo agente di un agenzia immobiliare nel sistema.		
Preconditions	<ul style="list-style-type: none"> -Essere loggati come agente. -Aver ricevuto almeno un offerta su un immobile pubblicato. 		
Success End Condition	Salvataggio di un nuovo agente nel sistema con credenziali di default.		
DESCRIPTION	Step	Manager	Sistema
Scenario principale	1	Clicca "Registra nuovo dipendente"	
	2		Mostra pagina registrazione dipendente
	3	Compila form selezionando "Agente" come ruolo	
	4	Clicca "Registra dipendente"	
	5		Mostra allert di conferma con opzioni "Annulla" e "Conferma"
	6	Clicca "Conferma"	
	7		Mostra caricamento
	8		Genera credenziali di default
	9		Salvataggio nuovo agente nel sistema con credenziali di default
	10		Mostra messaggio di conferma con allegato le credenziali di default
	11		Use case terminato con successo

Use case 4	Registra nuovo agente		
	Step	Manager	Sistema
EXTENSION A Non tutti i campi sono compilati oppure compilati correttamente	4.A	Clicca "Registra dipendente"	
	5.A		Mostra errori nei campi non compilati correttamente
	6.A		Ritorna al passo 3 dello scenario principale
EXTENSION B Il manager tenta di registrare un agente il cui nome e cognome sono in comune con un altro agente già registrato nel sistema	6.B	Clicca "Conferma"	
	7.B		Mostra caricamento
	8.B		Il sistema genera automaticamente un email alternativa che aggiunge un id univoco
	9.B		Ritorna al passo 9 dello scenario principale
EXTENSION C Clicca "Annulla" invece di "Conferma" nell'alert di conferma	6.C	Clicca "Annulla"	
	7.C		Ritorna al passo 3 dello scenario principale
EXTENSION D L'utente non riesce a connettersi con il sistema	6.D	Clicca "Conferma"	
	7.D		Mostra caricamento
	8.D		Avvisa l'utente dell'operazione fallita per motivi tecnici
	9.D		User case fallito

Figure 2.9:

2.6 Mock up

Introduzione

Dopo aver definito gli use case utilizzando il metodo Cockburn, siamo passati alla fase di progettazione visiva realizzando una serie di mockup interattivi in Figma. Questi mockup hanno lo scopo di rappresentare graficamente le interazioni dell'utente con il sistema, traducendo le specifiche funzionali in un'interfaccia visibile e navigabile. Ogni mockup è stato sviluppato tenendo conto delle diverse casistiche previste negli use case e nelle loro estensioni, in modo da garantire un'esperienza utente coerente e intuitiva.

Nei paragrafi seguenti verranno presentati i vari mockup, suddivisi per use case. Ogni sezione illustrerà le scelte progettuali effettuate, evidenziando come la UI si adatti alle diverse situazioni previste dai casi d'uso.

2.6.1 Caso d'Uso: Aggiungi un Annuncio Immobiliare

Per garantire un'esperienza utente ottimale, abbiamo progettato i mockup del caso d'uso *Aggiungi un Annuncio Immobiliare* seguendo i principi della user experience (UX) e dell'usabilità. Il percorso utente è stato studiato attentamente per minimizzare il carico cognitivo e semplificare l'interazione, in linea con i modelli proposti da Nielsen e Norman [1, 2].

Schermata Iniziale: Gestione Annunci Immobiliari

L'azione di aggiungere un nuovo annuncio parte dalla schermata di Gestione Annunci Immobiliari, dove l'utente trova:

- **Una lista degli immobili a lui associati**, che consente una rapida contestualizzazione.
- **Un pulsante unico ed evidente**, etichettato “Aggiungi Annuncio Immobiliare”, che centralizza l'azione primaria.

Flusso di Navigazione e Segmentazione del Form

Al click del pulsante, l'utente viene indirizzato a una schermata dedicata alla compilazione di un form articolato in più step. Questa segmentazione si basa sul principio del **chunking dell'informazione** [3], riducendo il carico di memoria e rendendo il processo meno gravoso. In particolare:

- **Step 1:** Richiede le informazioni basilari, quali il titolo, il tipo di contratto e il tipo di immobile.
- **Step 2 e successivi:** Raccolgono i dati principali e le caratteristiche secondarie dell'immobile, organizzati in modo logico e intuitivo.

Navigazione Sticky e Accessibilità degli Step

I pulsanti di navigazione, posizionati in alto con comportamento *sticky*, consentono all'utente di passare agevolmente da uno step all'altro, anche in presenza di schermate particolarmente lunghe. Tale scelta:

- Riduce il tempo necessario per individuare i controlli di navigazione.
- Favorisce un'interazione continua e fluida, in linea con i principi di design centrato sull'utente e le best practice in ambito HCI (Human-Computer Interaction) [4].

Integrazione della Mappa Interattiva

Per quanto riguarda l'inserimento dell'indirizzo:

- **La mappa interattiva** permette di visualizzare immediatamente la posizione indicata, offrendo un feedback visivo diretto.
- L'utente ha la possibilità di regolare la posizione direttamente sulla mappa, migliorando la precisione del dato inserito [5].

Gestione delle Immagini e Feedback Visivo

La sezione dedicata alle immagini è progettata per:

- **Consentire l'aggiunta di foto** tramite pulsante dedicato o drag and drop, facilitando l'upload in maniera intuitiva.
- **Permettere l'inserimento di una descrizione** per ogni immagine, migliorando la contestualizzazione visiva dell'annuncio [6].

Anteprima e Conferma Finale

Infine, viene presentato uno schema riepilogativo che funge da anteprima dell'annuncio così come apparirà agli utenti finali. Una volta verificata la correttezza dei dati:

- L'utente può cliccare il pulsante “**Pubblica**”, che innesca un processo di caricamento e validazione.
- Al termine del processo, viene mostrato un messaggio di conferma che attesta il successo dell'operazione, riducendo l'ansia da incertezza e rafforzando la fiducia nel sistema [1].

Detti Estate 25
Portale Agenzia

I Miei Annunci Le mie Proposte Messaggi Promozionali Team Aldo CASA SRL

Crea il tuo Annuncio

1. Informazioni di Base

Titolo Annuncio

Tipologia immobile

Vuoi vendere o affittare?

Avanti

Cockburn: step 3

Indietro

Crea il tuo Annuncio

2. Inserimento dei Dettagli Principali

Prezzo (€): <input type="text" value="250.000 €"/>	Superficie (m²): <input type="text" value="85 m²"/>	Numero di Stanze: <input type="text" value="2"/>
Numero Bagni <input type="text" value="1"/>	Piano: <input type="text" value="Terzo Piano"/>	
Classe Energetica: <input type="text" value="A4 (<1 Epg)"/>		

Avanti

Cockburn: step4/5

Indietro

Crea il tuo Annuncio

3. Indirizzo e Localizzazione

Indirizzo

Interno e Scala

CAP

Città

Provincia



Avanti

Cockburn: step 4/5

Figure 2.10: Mockup: scenario principale della tabella di Cockburn del caso d'uso nuovo annuncio.

Cockburn: step 4/5

Cockburn: step 6

Figure 2.11: Mockup: scenario principale della tabella di Cockburn del caso d'uso nuovo annuncio.

Crea il tuo Annuncio

6. Anteprima dell'Annuncio

Appartamento in Vendita



Casa tra gli ulivi

Dettagli Immobile

Indirizzo: Via Roma, 123, Napoli

Superficie: 85 m²

Camere: 2

Bagni: 1

Prezzo: € 250.000

Luca Bianchi - Agenzia Casa Mia
 Email: luca.bianchi@agenziacasamia.com
 Telefono: +39 987 654 3210

Caratteristiche dell'Immobile

- Fermata metro a 300m
- Supermercato a 500m
- Scuola primaria a 800m
- Ospedale a 1km

Fai una Proposta

Il tuo nome:

La tua email:

La tua proposta:

Invia Proposta

Proposte Ricevute

Nome	Email	Proposta	Controposta	Status
Mario Rossi	mario.rossi@exa...	€240.000	€245.000	In Trattazione
Giulia Bianchi	giulia.bianchi@exa...	€250.000	-	Accettato
Luigi Verdi	luigi.verdi@exam...	€230.000	-	Rifiutato
Anna Neri	anna.neri@exam...	€240.000	€245.000	In Trattazione

Pubblica

Cockburn: step 7/8

Figure 2.12: Mockup: scenario principale della tabella di Cockburn del caso d'uso nuovo annuncio.

Estensione A: Annuncio in Sospeso

In alcuni casi, un agente immobiliare potrebbe tentare di aggiungere un nuovo annuncio mentre ne ha già uno in sospeso. Per garantire un'esperienza utente fluida e prevenire la perdita accidentale di dati, il sistema mostra un pop-up che chiede esplicitamente se si desidera creare un nuovo annuncio o ripristinare quello precedente.

Gestione del Pop-up e Scelta dell'Utente

All'azione di aggiunta di un nuovo annuncio con un annuncio incompleto in sospeso, viene visualizzato un messaggio di conferma con due opzioni:

- **Crea un nuovo annuncio:** l'utente sceglie di avviare un nuovo processo di inserimento. Il sistema informa chiaramente che i dati dell'annuncio precedente andranno persi.
- **Ripristina l'annuncio precedente:** il sistema recupera i dati salvati localmente e ripristina il modulo alla condizione precedente.

Questa decisione è fondamentale per garantire il controllo dell'utente sull'operazione e per evitare errori involontari che potrebbero compromettere l'inserimento dell'annuncio.

Estensione A: Creazione di un Nuovo Annuncio

Se l'utente seleziona **Crea un nuovo annuncio**, il sistema mostra un messaggio di conferma che avvisa della perdita definitiva dei dati precedentemente salvati. Questa scelta progettuale si basa sul principio della **prevenzione degli errori** [1], poiché aiuta a evitare cancellazioni accidentali e garantisce una maggiore consapevolezza dell'azione intrapresa.

Il messaggio di conferma include due pulsanti:

- **Annulla:** consente di tornare indietro senza perdere i dati in sospeso.
- **Procedi:** conferma l'azione e avvia un nuovo annuncio.

I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Gestione Annunci immobiliari | Aggiungi Annuncio +

Gestione Proposte | Aggiungi Proposta +

Titolo		Copertina	Prezzo	Contratto	Proposte
Panoramic Attic in Monteverde Vecchio			€680/mese	Affitto	10 ↕ X

In Trattativa

Nome	Email	Proposta	Controproposta	azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	✓ ↕ X
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	✓ ↕ X
Luigi Verdi	luigi.verdi@example...	€230.000	-	✓ ↕ X
Anna Neri	anna.neri@example...	€240.000	€245.000	✓ ↕ X

Proposte Accettate

Nome	Email	Proposta	Controproposta
Mario Rossi	mario.rossi@example...	€240.000	€245.000

click nuovo annuncio



I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Gestione Annunci immobiliari | Aggiungi Annuncio +

Annuncio in sospeso rilevato

Panoramic Attic in Monteverde Vecchio Il sistema ha riconosciuto un annuncio in sospeso. Vuoi ripristinare i dati oppure creare un nuovo annuncio?

In Trattativa

Nome	Email	Proposta	Controproposta	azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	✓ ↕ X
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	✓ ↕ X
Luigi Verdi	luigi.verdi@example...	€230.000	-	✓ ↕ X
Anna Neri	anna.neri@example...	€240.000	€245.000	✓ ↕ X

Proposte Accettate

Nome	Email	Proposta	Controproposta
Mario Rossi	mario.rossi@example...	€240.000	€245.000

Cockburn: extension A.2/A.3/A.4



I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Gestione Annunci immobiliari | Aggiungi Annuncio +

Creazione nuovo annuncio

Hai cliccato su Nuovo annuncio.
Attenzione: i dati dell'annuncio in sospeso andranno persi.
Confermi di voler procedere?

In Trattativa

Nome	Email	Proposta	Controproposta	azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	✓ ↕ X
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	✓ ↕ X
Luigi Verdi	luigi.verdi@example...	€230.000	-	✓ ↕ X
Anna Neri	anna.neri@example...	€240.000	€245.000	✓ ↕ X

Cockburn: extension A.6

Figure 2.13: Mockup: estensione A della tabella di Cockburn del caso d'uso nuovo annuncio

Estensione B: Ripristino di un Annuncio Precedente

Se l'utente sceglie di **ripristinare l'annuncio precedente**, il sistema avvia un processo di caricamento per fornire un feedback visivo sulla ripresa dei dati. Sebbene i dati siano salvati localmente e il recupero sia immediato, un **indicatore di caricamento fittizio** viene mostrato per alcuni secondi prima di caricare la schermata.

Questa soluzione è basata sul principio della **coerenza con le aspettative dell'utente** [4]. In un contesto digitale, un ripristino istantaneo potrebbe apparire innaturale e creare confusione. L'indicatore di caricamento:

- Rafforza la percezione di un processo in corso, migliorando la trasparenza dell'operazione.
- Evita che l'utente si domandi se il recupero sia realmente avvenuto o se ci siano stati problemi tecnici.
- Contribuisce a una transizione più fluida tra stati dell'interfaccia.

Una volta completato il caricamento, il sistema presenta l'interfaccia con i dati precedentemente salvati, consentendo all'utente di riprendere il processo da dove era stato interrotto.

I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Gestione Annunci immobiliari

Gestione Proposte

In Trattativa	Nome	Email	Proposta	Controproposta	Azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	✓ ↗ X	
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	✓ ↗ X	
Luigi Verdi	luigi.verdi@example...	€230.000	-	✓ ↗ X	
Anna Neri	anna.neri@example...	€240.000	€245.000	✓ ↗ X	

Proposte Accettate	Nome	Email	Proposta	Controproposta
Mario Rossi	mario.rossi@example...	€240.000	€245.000	

click nuovo annuncio



I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Gestione Annunci immobiliari

Anuncio in sospeso rilevato

Il sistema ha riconosciuto un annuncio in sospeso. Vuoi ripristinare i dati oppure creare un nuovo annuncio?

In Trattativa	Nome	Email	Proposta	Controproposta	Azioni
Mario Rossi	mario.rossi@example...	€250.000	Nuovo annuncio	Ripristina dati	✓ ↗ X
Giulia Bianchi	giulia.bianchi@example...	€230.000	-	✓ ↗ X	
Luigi Verdi	luigi.verdi@example...	€230.000	-	✓ ↗ X	

Cockburn: extension B.2/B.3/B.4



I Miei Annunci | Le mie Proposte | Messaggi Promozionali | Team | Aldo CASA SRL

Crea il tuo Annuncio

2. Inserimento dei Dettagli Principali

Prezzo (€): 250.000 €	Superficie (m ²): 85 m ²	Numero di Stanze: 2
Numero Bagni: 3	Piano: Terzo Piano	
Classe Energetica: D (1,51 - 2,00 Epg)		

Cockburn: extension B.5

Figure 2.14: Mockup: estensione B della tabella di Cockburn del caso d'uso nuovo annuncio

Estensione C: Errore Durante la Pubblicazione dell'Annuncio

Nel caso in cui si verifichi un errore durante la pubblicazione dell'annuncio immobiliare, viene mostrato un popup informativo che comunica all'utente il problema riscontrato. Questo popup ha il compito di rassicurare l'utente che i dati inseriti non andranno persi, in quanto vengono salvati localmente, e invita a riprovare più tardi.

Gestione dell'Errore e Feedback Utente

Il popup presenta i seguenti elementi chiave:

- **Messaggio chiaro e rassicurante:** informa l'utente dell'errore senza tecnicismi, riducendo il senso di frustrazione.
- **Pulsante “Riprova”:** consente di tentare nuovamente la pubblicazione con un'animazione di caricamento, per segnalare che l'azione è in corso e mantenere un senso di controllo e progressione.
- **Pulsante “Monstra i Dettagli”:** offre la possibilità di visualizzare l'errore tecnico riscontrato. Questa funzione segue i principi di trasparenza e controllo dell'utente, utili sia per utenti avanzati che per eventuali tecnici di supporto che potrebbero risolvere il problema più rapidamente.

Principi di Design Applicati

L'implementazione di questa gestione degli errori si basa su diversi principi della UX e dell'usabilità:

- **Visibilità dello stato del sistema [1]:** l'animazione di caricamento fornisce un'indicazione chiara che il sistema sta lavorando sulla richiesta dell'utente.
- **Prevenzione degli errori [1]:** il salvataggio locale dei dati riduce la possibilità di perdere informazioni a causa di un problema temporaneo.
- **Fornire informazioni utili per il recupero dall'errore [1]:** il messaggio di errore è accompagnato da suggerimenti su come procedere e un'opzione per visualizzare i dettagli tecnici, utile per il supporto tecnico.

Queste soluzioni mirano a mantenere un'esperienza utente fluida e priva di frustrazione, minimizzando il disagio derivante da problemi tecnici imprevisti.

Indietro Crea il tuo Annuncio Pubblica

6. Anteprima dell'Annuncio

Appartamento in Vendita

Casa tra gli ulivi

Dettagli Immobile
Indirizzo: Via Roma, 123, Napoli
Superficie: 85 m²
Camere: 2 < > MacBook Pro 14" v

click pubblica nuovo annuncio

↓

Indietro Crea il tuo Annuncio Pubblica

6. Anteprima dell'Annuncio

Errore nella Pubblicazione

La conferma della pubblicazione non è andata a buon fine perché il sistema è offline.
Tuttavia, i dati dell'annuncio sono stati salvati in locale.

Monstra i Dettagli Riprova

Casa tra gli ulivi

Dettagli Immobile
Indirizzo: Via Roma, 123, Napoli

Cockburn: extension C.9/C.10

Figure 2.15: Mockup: estensione C della tabella di Cockburn del caso d'uso nuovo annuncio

Estensione D: Gestione degli Errori in Fase di Pubblicazione

Nel caso in cui, al momento della pubblicazione dell'annuncio immobiliare, il sistema rilevi errori in alcuni campi del form, viene attivato un meccanismo di gestione degli errori progettato per minimizzare la frustrazione dell'utente e facilitare la correzione delle informazioni errate.

Feedback Visivo e Navigazione Intelligente

Per evitare che l'utente debba individuare manualmente i campi errati, il sistema implementa un'animazione che lo riporta automaticamente al primo step contenente errori. Inoltre, sopra il titolo dello step viene visualizzato un messaggio chiaro che segnala la presenza di dati non corretti. Questo messaggio include un riepilogo dei campi che necessitano di correzione, garantendo così un'immediata comprensione del problema senza dover scorrere l'intero form.

Evidenziazione degli Errori nei Campi del Form

I campi contenenti errori vengono evidenziati attraverso i seguenti accorgimenti visivi:

- Il bordo del campo assume un colore rosso evidente, attirando immediatamente l'attenzione dell'utente.
- Al di sotto del campo appare un messaggio testuale che spiega chiaramente la natura dell'errore (ad esempio, "Inserire un numero valido per il prezzo" o "Il titolo dell'annuncio non può essere vuoto").

Questa strategia segue i principi di extiterror prevention e extiterror recovery proposti da Nielsen [1], permettendo all'utente di comprendere e correggere rapidamente gli errori.

Indicazione degli Step con Errori

Per garantire una visione d'insieme immediata dello stato del form, il sistema modifica dinamicamente il colore dei riquadri degli step nel percorso di navigazione:

- Gli step senza errori mantengono il colore verde, confermando all'utente che le informazioni inserite sono corrette.
- Gli step contenenti errori cambiano colore dal verde al rosso, segnalando visivamente dove è necessario intervenire, senza bisogno di ulteriori spiegazioni testuali.

Questa soluzione si basa sui principi dell' extitimmediate feedback e dell' extiterror mapping [4], riducendo il carico cognitivo e migliorando l'usabilità complessiva del sistema.

Esperienza Utente e Riduzione della Frustrazione

L'implementazione di questi meccanismi mira a ottimizzare l'esperienza dell'utente durante la compilazione del form, minimizzando la frustrazione derivante dalla correzione degli errori. La combinazione di feedback visivo, animazioni intuitive e segnalazioni mirate permette un'interazione fluida e priva di ostacoli, in linea con le best practice in ambito UX.

The mockup illustrates a user flow for creating a new advertisement. It starts with a preview step where a house image is shown with the caption "Casa tra gli ulivi". Below the image are details: "Dettagli Immobile", "Indirizzo: Via Roma, 123, Napoli", "Superficie: 85 m²", and a camera count of "2". A note indicates the image was taken with a "MacBook Pro 14" screen. An annotation "click pubblica nuovo annuncio" points to the "Pubblica" button at the top right of the preview screen. A downward arrow leads to the next step, which is an error page titled "Crea il tuo Annuncio". The error message "Errore nella pubblicazione" states: "Alcuni campi non sono corretti. Verifica e correggi i seguenti campi evidenziati in rosso:" followed by a bulleted list: "• Prezzo", "• Numero stanze", and "• Cap". The page is labeled "2. Inserimento dei Dettagli Principali". Fields for "Prezzo (€)" (000 €), "Superficie (m²)" (85 m²), and "Numero di Stanze" are shown with validation errors. Other fields include "Numero Bagni" (1), "Piano" (Terzo Piano), and "Classe Energetica" (A4 (<1 Epg)).

Figure 2.16: Mockup: estensione D della tabella di Cockburn del caso d'uso nuovo annuncio

2.6.2 Caso d'Uso: Attivazione e Disattivazione Notifiche

Il sistema prevede un'interfaccia dedicata alla gestione delle preferenze di notifica, pensata per permettere all'utente di curare la propria esperienza d'uso dell'applicazione, ricevendo news solo sugli argomenti di interesse.

In questa sezione andremo a esaminare le scelte effettuate durante la progettazione, l'impatto sull'esperienza utente e i prototipi generati alla fine dell'analisi.

Tipologie di Notifiche e Controllo Utente

Le notifiche sono suddivise in diverse categorie per permettere una personalizzazione granulare:

- **Annunci di nuovi immobili:** notifiche basate sulle ricerche dell'utente.
- **Risposte alle offerte:** aggiornamenti sulle interazioni con gli annunci pubblicati.
- **Messaggi promozionali:** comunicazioni di marketing e offerte esclusive.

L'utente può, in qualsiasi momento, disattivare le notifiche per una o più categorie, mantenendo un controllo totale sulla propria esperienza [4].

Gestione delle Notifiche e Conferma delle Modifiche

La gestione delle notifiche avviene principalmente attraverso la schermata delle notifiche, composta da:

- **Lista delle notifiche ricevute**, ognuna cliccabile per visualizzare i dettagli.
- **Barra laterale con le categorie di notifiche**, suddivise in:
 - **Categorie attive**, con notifiche attualmente abilitate.
 - **Categorie disattivate**, che non inviano più notifiche.

In cima alla barra laterale è presente un'icona che, se cliccata, apre una schermata popup intitolata "Attiva e Disattiva Notifiche". All'interno, ogni categoria è rappresentata da un toggle switch che indica lo stato attuale delle notifiche.

Feedback Visivo e Animazioni Intuitive

Per garantire un'interazione chiara e immediata, il sistema utilizza diverse tecniche di UX design:

- **Animazione di transizione:** quando un toggle viene modificato, la categoria si sposta visivamente tra la sezione attiva e quella disattiva, sfruttando il principio di **gestalt della continuità** [3] per rendere il cambiamento intuitivo.

- **Feedback visivo immediato:** l'utente percepisce immediatamente l'effetto dell'azione senza necessità di un testo esplicativo eccessivo.

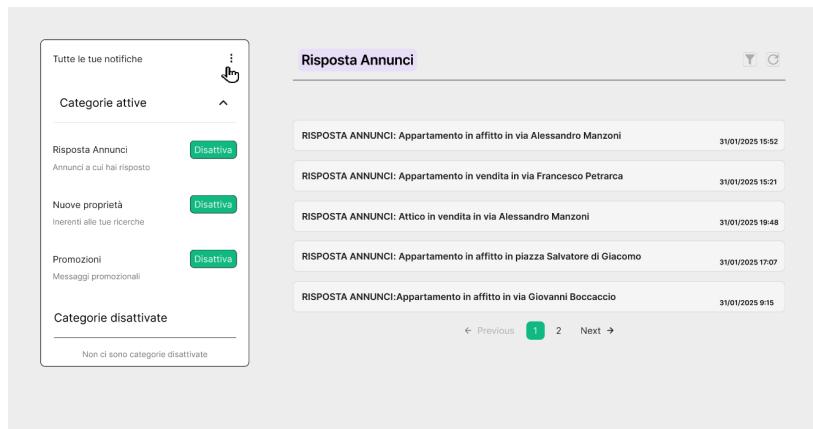
Conferma e Implicazioni della Disattivazione

Per evitare errori accidentali e garantire consapevolezza delle conseguenze, la disattivazione di una categoria di notifiche è accompagnata da:

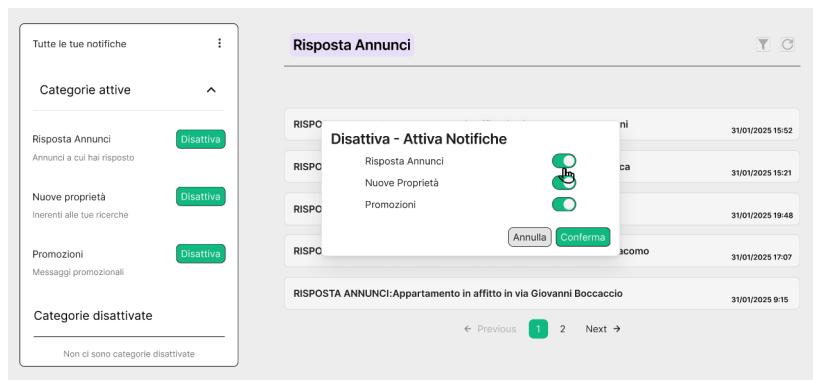
- Un popup di conferma che informa l'utente che, durante il periodo in cui le notifiche sono disattivate, le notifiche non potranno essere recuperate [5].
- Un ulteriore messaggio di avviso prima della conferma definitiva, in linea con le **heuristiche di usabilità di Nielsen** [1] per la prevenzione degli errori.

Solo dopo la conferma finale, il sistema applica le modifiche alle preferenze dell'utente, garantendo un'interazione consapevole e trasparente.

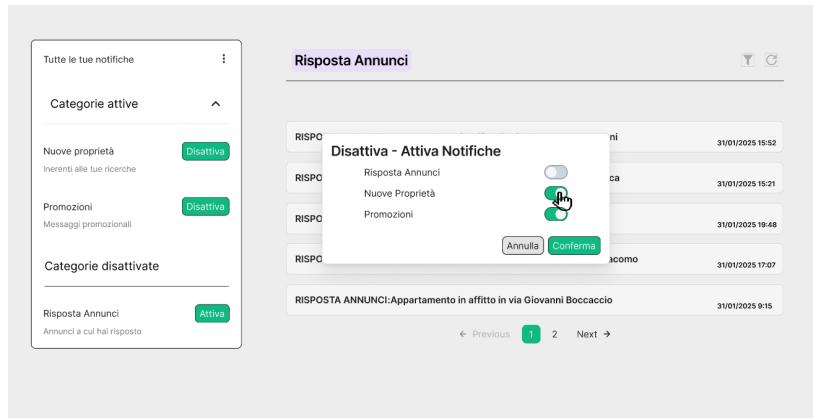
Nel prototipo questi comportamenti sono stati modellati con un pulsante, tuttavia nell'applicazione finale è stato deciso di sostituirlo con un menù contestuale.



Cockburn: step 1/2/3

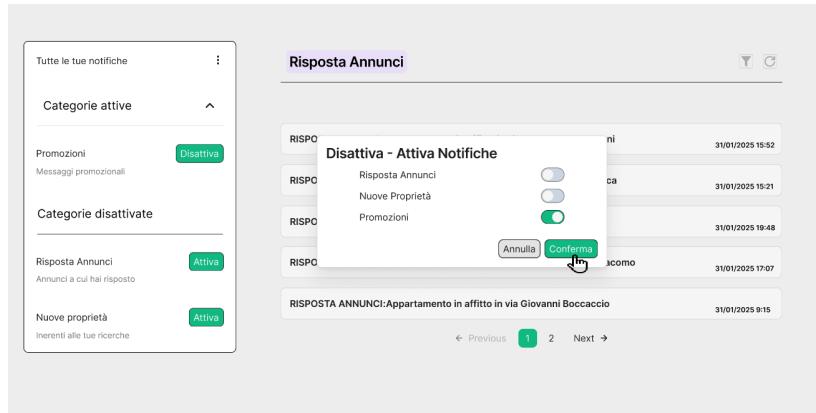


Cockburn: step 4

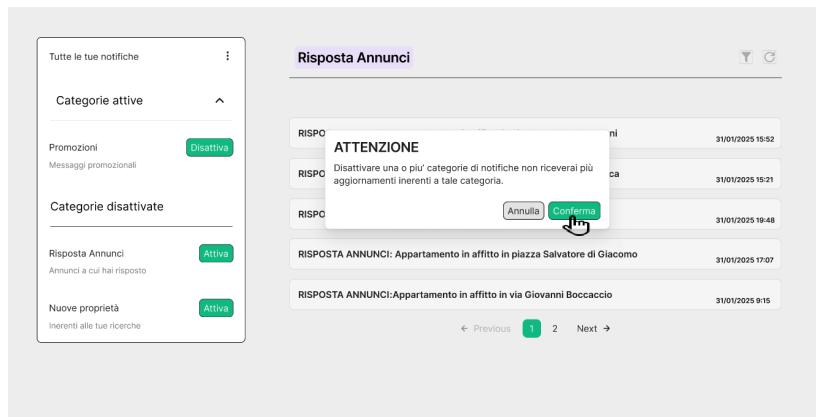


Cockburn: step 4

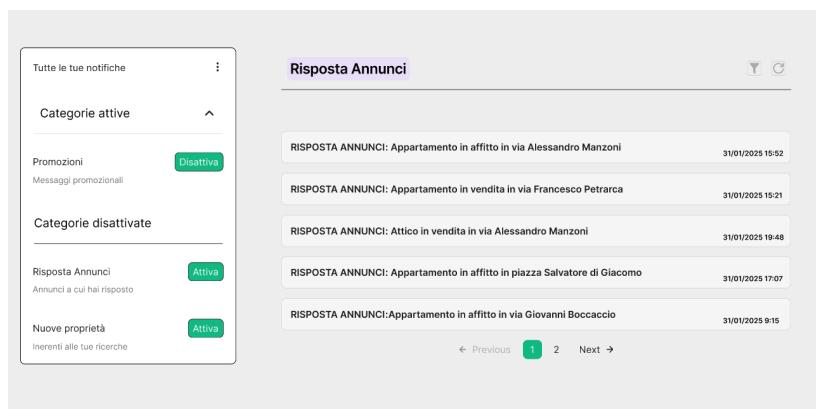
Figure 2.17: Mockup: scenario principale della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica



Cockburn: step 5



Cockburn: step 6/7



Conckburn: step 8/9/10

Figure 2.18: Parte 2 mockup: scenario principale della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica

Estensione A: Disattivazione Notifiche dalla Visualizzazione di una Notifica

Per offrire un maggiore controllo sulla gestione delle notifiche senza interrompere l'esperienza utente, il sistema permette di disattivare una categoria direttamente dalla visualizzazione di una notifica specifica. Questa variante è progettata per garantire una modifica consapevole delle preferenze, evitando azioni impulsive che potrebbero compromettere la ricezione di informazioni rilevanti.

Interfaccia e Comportamento del Bottone

Alla fine del testo di ogni notifica, se la relativa categoria è attiva, è presente un pulsante neutro con la dicitura “Disattiva notifiche”. Accanto al pulsante, un testo in grigio informa l'utente della funzione del pulsante, evitando ambiguità. L'utilizzo di colori non accesi e di un design discreto segue il principio della **gerarchia visiva** [6], scoraggiando azioni impulsive che potrebbero portare alla perdita involontaria di notifiche future.

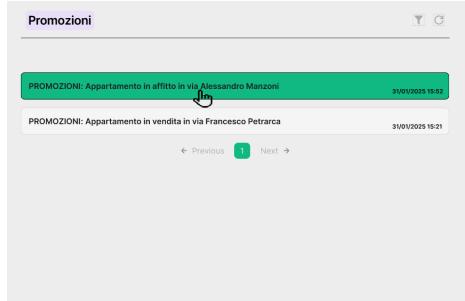
Modifica dello Stato e Feedback Visivo

Quando l'utente clicca sul pulsante, il testo del pulsante cambia colore, diventando rosso, e il messaggio a fianco si aggiorna per sottolineare che la categoria di notifiche è stata disattivata. Questo utilizza il principio della **salienza visiva** [1], enfatizzando il cambiamento e rendendo immediatamente chiara la conseguenza dell'azione.

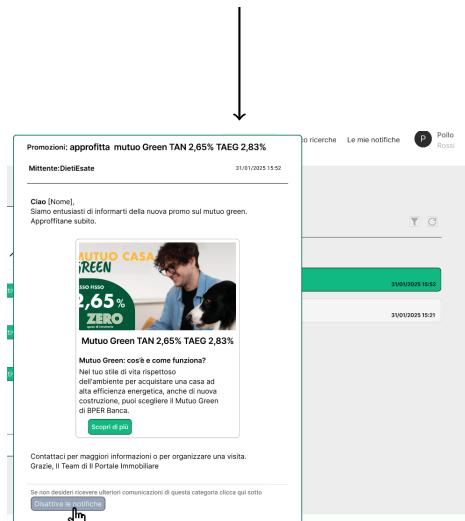
Incentivo alla Riattivazione

Una volta disattivata una categoria tramite questa modalità, viene visualizzato un secondo pulsante con una call-to-action mirata per incentivare la riattivazione delle notifiche. Il design e il posizionamento del pulsante sfruttano il **principio dell'affordance** [2], rendendo chiaro che l'utente ha la possibilità di tornare indietro sulla sua decisione in modo semplice e immediato.

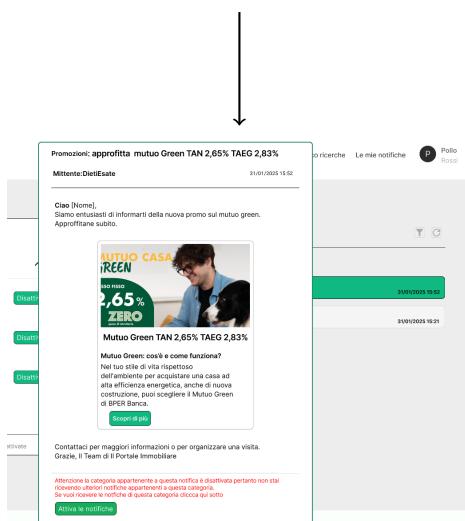
Questa estensione si integra perfettamente con il modello generale di gestione delle notifiche, garantendo un'interazione fluida e coerente con le esigenze dell'utente. Nel caso in cui l'utente scelga di riattivare la categoria delle notifiche direttamente da una notifica, si passa all'**Estensione F**, che approfondisce questa modalità di gestione a partire dalle notifiche disattivate.



Cockburn: Extension A.2/A.3



Cockburn: Extension A.4



Cockburn: extension A.5

Figure 2.19: Mockup: estensione A della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica

Estensione B: Conferma Senza Modifiche

Nel caso in cui l'utente apra il popup "Attiva e Disattiva Notifiche" e prema il pulsante di conferma senza apportare alcuna modifica allo stato delle categorie di notifica, il sistema segue un comportamento mirato a garantire un'interazione fluida e priva di frizioni inutili.

Gestione dell'Interazione

Per evitare interruzioni superflue, il sistema rileva automaticamente che non sono state effettuate modifiche e si limita a chiudere il popup senza richiedere ulteriori conferme. Questo approccio è in linea con le **euristiche di usabilità di Nielsen** [1], in particolare il principio di **minimizzazione del carico cognitivo**, che riduce il numero di azioni necessarie per completare un'operazione priva di effetti.

Feedback Visivo

Sebbene non venga mostrato alcun messaggio di doppia conferma, il sistema utilizza piccoli accorgimenti di UX design per rendere l'interazione chiara:

- **Animazione di chiusura fluida:** il popup si chiude con una breve transizione, migliorando la percezione di un'interazione naturale.
- **Assenza di notifiche di conferma:** poiché nessuna modifica è stata apportata, il sistema evita messaggi ridondanti che potrebbero confondere l'utente.
- **Chiarezza dello stato iniziale:** i toggle switch rimangono invariati, rafforzando la prevedibilità del sistema.

Questo comportamento segue il principio della **coerenza e prevedibilità** [2], assicurando che il sistema reagisca in modo intuitivo alle azioni dell'utente senza introdurre complessità non necessarie.

Estensione C: Modifica Rapida dello Stato delle Notifiche dalla Barra Laterale

Per migliorare l'accessibilità e rendere la gestione delle notifiche più immediata, il sistema offre un metodo alternativo per attivare e disattivare le notifiche direttamente dalla barra laterale. Questa estensione elimina la necessità di accedere alla schermata dedicata, offrendo un controllo contestuale più rapido ed efficace.

Interfaccia e Interazione

Ogni categoria di notifica presente nella barra laterale include un pulsante contestuale che consente di modificarne lo stato. Il pulsante ha due possibili stati:

- **Attiva**, se la categoria è attualmente disabilitata.
- **Disattiva**, se la categoria è attualmente abilitata.

Quando l'utente interagisce con il pulsante:

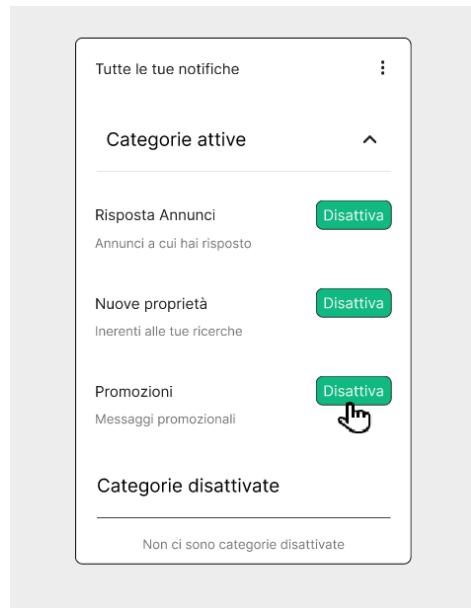
- Se clicca su **Disattiva**, appare un popup di conferma che informa sulle implicazioni della disattivazione, in linea con i principi di prevenzione degli errori di Nielsen [1].
- Se l'utente conferma, il sistema esegue una transizione animata per spostare la categoria dalla sezione delle notifiche attive a quella delle notifiche disattivate.
- Il pulsante cambia stato, diventando **Attiva**, in modo da riflettere visivamente la modifica e garantire un feedback immediato.

Feedback Visivo e UX Design

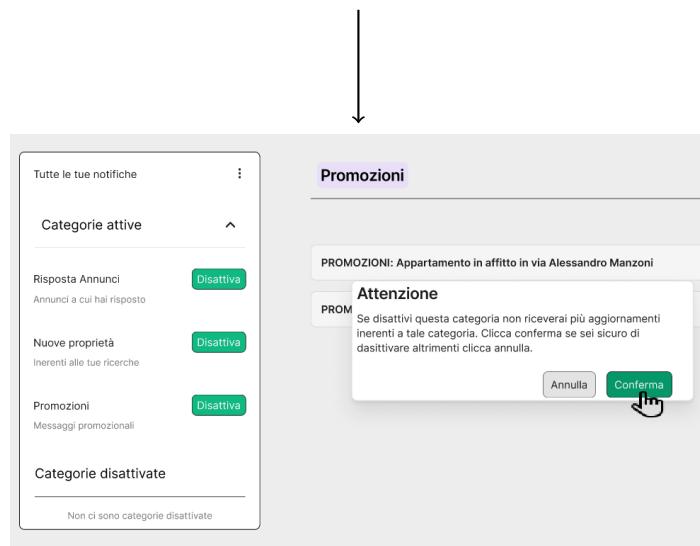
Per rendere il cambiamento chiaro e intuitivo, il sistema implementa le seguenti tecniche di UX design:

- **Popup di conferma**: viene visualizzato prima di procedere con la modifica, seguendo le euristiche di usabilità per la prevenzione degli errori [1].
- **Animazione di transizione**: la categoria viene spostata visivamente tra le sezioni della barra laterale, applicando il principio della **gestalt della continuità** [3] per rendere il cambiamento più naturale.
- **Aggiornamento dello stato del pulsante**: il pulsante cambia dinamicamente per riflettere lo stato attuale della categoria, riducendo l'ambiguità e migliorando la prevedibilità dell'interazione.

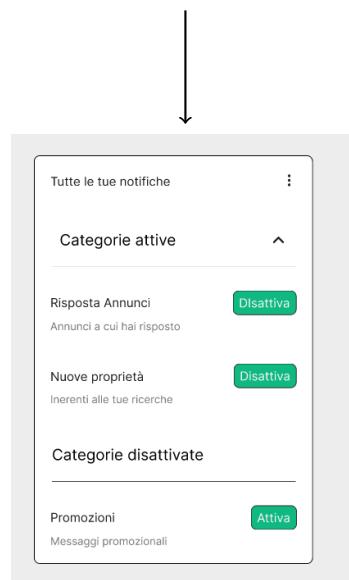
Questa estensione garantisce un'esperienza utente più fluida e immediata, riducendo il numero di passaggi necessari per la gestione delle notifiche senza compromettere la chiarezza e il controllo dell'utente.



Cockburn: Extension C.2



Cockburn: step 6/7



Cockburn: step 8/9/10

Estensione D: Modifica Rapida dello Stato delle Notifiche con Attivazione Immediata

Questa variante rappresenta l'approccio duale dell'**Estensione C**, semplificando ulteriormente l'attivazione delle notifiche. L'obiettivo è ridurre i passaggi necessari per riattivare una categoria disattivata, mantenendo comunque un controllo chiaro sulla disattivazione.

Interfaccia e Interazione

Analogamente all'**Estensione C**, ogni categoria nella barra laterale dispone di un pulsante contestuale per modificarne lo stato. Il pulsante può assumere due stati:

- **Attiva**, se la categoria è attualmente disabilitata.
- **Disattiva**, se la categoria è attualmente abilitata.

Le interazioni dell'utente variano a seconda dell'azione eseguita:

- **Disattivazione di una categoria:**
 - Al clic su **Disattiva**, appare un popup di conferma che informa l'utente sulle conseguenze della scelta, prevenendo azioni accidentali in linea con i principi di Nielsen [1].
 - Se confermata, la categoria viene spostata nella sezione delle notifiche disattivate tramite un'animazione di transizione.
 - Il pulsante cambia stato, diventando **Attiva**, fornendo un feedback visivo chiaro sulla modifica.
- **Attivazione di una categoria:**
 - Al clic su **Attiva**, il sistema aggiorna immediatamente lo stato della notifica senza richiedere una conferma esplicita.
 - La categoria viene spostata nella sezione delle notifiche attive con un'animazione fluida, applicando il principio della **gestalt della continuità** [3].
 - Il pulsante cambia stato in **Disattiva**, rendendo la modifica evidente e intuitiva.

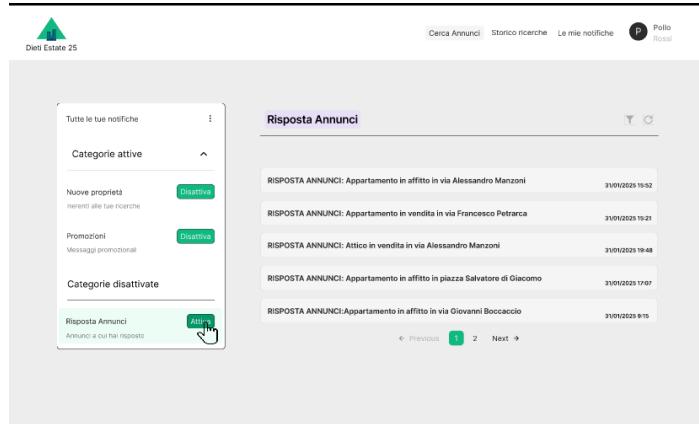
Feedback Visivo e UX Design

L'esperienza utente è ottimizzata tramite tecniche di design che garantiscono chiarezza e immediatezza:

- **Popup di conferma per la disattivazione:** aiuta a prevenire errori e rende consapevole l'utente delle conseguenze della scelta [1].

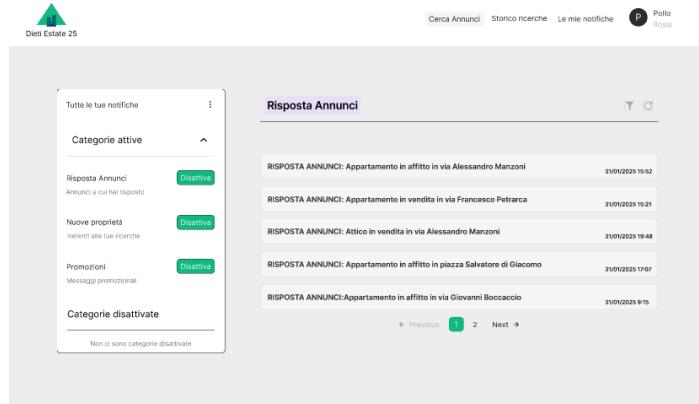
- **Animazione di transizione:** assicura una continuità visiva fluida nello spostamento delle categorie, migliorando la percezione del cambiamento [3].
- **Aggiornamento immediato dello stato del pulsante:** il cambio di testo e colore riflette lo stato corrente della categoria, riducendo l'ambiguità e migliorando la prevedibilità dell'interazione.

Questa estensione semplifica l'attivazione delle notifiche, eliminando il passaggio della conferma e migliorando la fluidità dell'interazione, senza compromettere il controllo dell'utente sulla gestione delle proprie preferenze.



© 2025 - Solo a scopo didattico

Cockburn: Extension D.2



© 2025 - Solo a scopo didattico

Cockburn: step 8/9/10

Figure 2.21: Mockup: estensione D della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica

Estensione E: Errore Durante la Modifica dello Stato delle Categorie di Notifica

Nel caso in cui si verifichi un errore durante il tentativo di modifica dello stato di una categoria di notifica, viene visualizzato un messaggio di errore sotto forma di un popup informativo. Il messaggio informa l'utente che la modifica non è riuscita e lo invita a riprovare più tardi, senza salvare le modifiche effettuate.

Gestione dell'Errore e Feedback Utente

Il popup di errore presenta i seguenti elementi chiave:

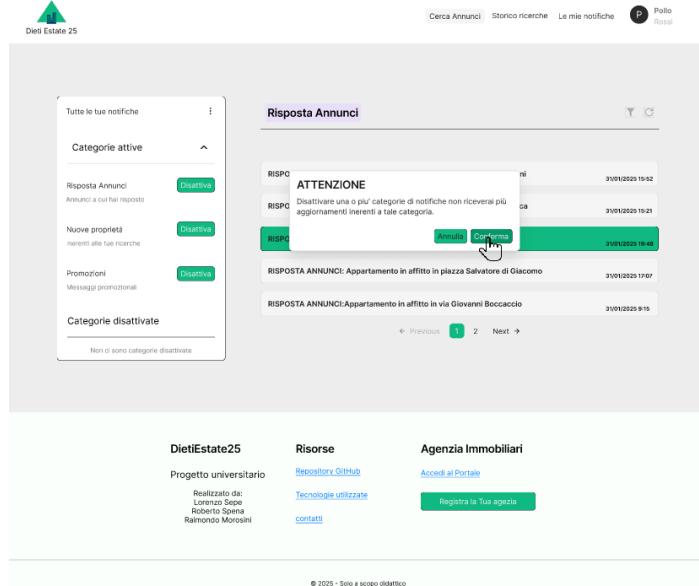
- **Messaggio chiaro e informativo:** il messaggio comunica all'utente che l'operazione non è andata a buon fine, senza entrare in dettagli tecnici, per ridurre il rischio di frustrazione. L'utente viene anche informato che l'operazione non è stata completata e che le modifiche non sono state salvate.
- **Pulsante “Ok”:** consente all'utente di chiudere il popup e tornare all'interfaccia principale. Il pulsante di conferma è chiaro e consente di riprendere l'interazione senza indugi.

Principi di Design Applicati

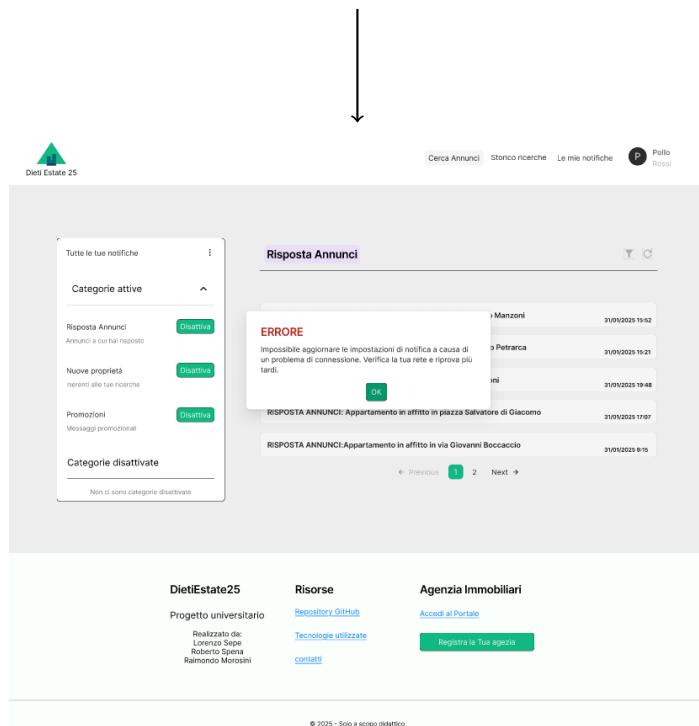
L'approccio di gestione dell'errore in questa estensione si fonda sui seguenti principi di UX e usabilità:

- **Visibilità dello stato del sistema [1]:** l'errore è comunicato all'utente attraverso un popup che evidenzia chiaramente che il sistema non è riuscito a completare l'operazione.
- **Prevenzione degli errori [1]:** sebbene l'errore non possa essere evitato completamente, il sistema offre un feedback immediato e comprensibile, impedendo all'utente di rimanere confuso o incerto sullo stato dell'operazione.
- **Semplicità e chiarezza [1]:** il messaggio di errore è semplice e diretto, senza sovraccaricare l'utente con informazioni tecniche. L'invito a riprovare più tardi mantiene il flusso di lavoro semplice e lineare.
- **Controllo dell'utente [2]:** l'utente ha il pieno controllo sulla gestione dell'errore, poiché il popup permette di chiudere facilmente l'interfaccia e riprendere l'attività, mantenendo un'esperienza utente fluida.

Questa soluzione di gestione dell'errore è progettata per garantire un'esperienza utente chiara e senza frustrazioni, minimizzando il disagio derivante da errori tecnici imprevisti e offrendo un percorso semplice per riprendere l'interazione.



Cockburn: step 7



Cockburn: Extension E.7/E.8

Figure 2.22: Mockup: estensione E della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica

Estensione F: Riattivazione Notifiche dalle Notifiche Disattivate

Questa estensione consente all'utente di riattivare una categoria di notifiche direttamente da una notifica precedentemente ricevuta e appartenente a una categoria disattivata. L'obiettivo è fornire un meccanismo immediato per ripristinare le notifiche quando l'utente si rende conto della loro utilità.

Indicazione dello Stato e Call-to-Action

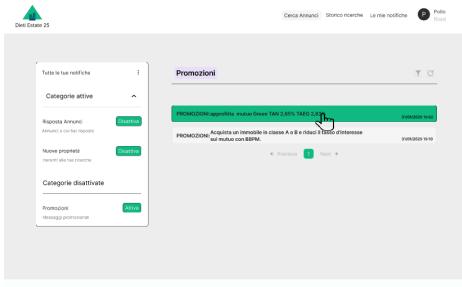
Quando una notifica proviene da una categoria disattivata, il sistema mostra un messaggio di avviso evidenziato che informa l'utente che non riceverà più aggiornamenti simili. Il pulsante associato cambia stato e diventa un invito all'azione con il testo "Riattiva notifiche". Questo sfrutta il **principio della reversibilità** [4], permettendo all'utente di annullare la decisione precedente senza difficoltà.

Feedback Visivo e Conferma della Riattivazione

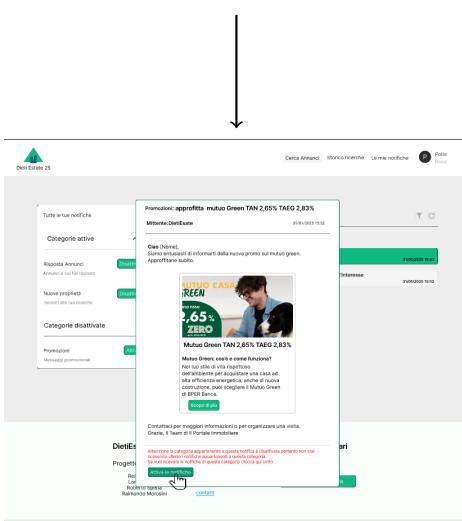
Alla pressione del pulsante, il testo cambia colore in verde e il messaggio informativo si aggiorna, confermando che le notifiche per quella categoria sono state riattivate. Il sistema può fornire un'ulteriore conferma con un breve messaggio di notifica o una vibrazione del dispositivo per enfatizzare l'azione completata.

Coerenza con il Modello di Gestione Notifiche

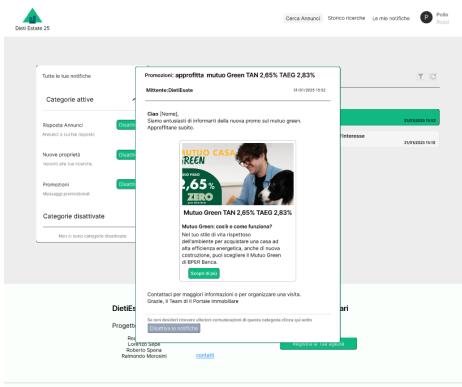
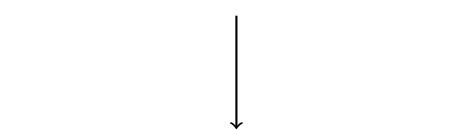
Questa estensione rafforza la coerenza dell'interfaccia di gestione delle notifiche, mantenendo le scelte dell'utente sempre modificabili e promuovendo un'interazione trasparente e prevedibile. L'utente può così gestire le notifiche senza dover accedere necessariamente alla schermata delle impostazioni, riducendo il carico cognitivo e migliorando l'usabilità complessiva del sistema.



Cockburn: Extension F.2



Cockburn: Extension F.3



Cockburn: Extension F.4/F.5

Figure 2.23: Mockup: estensione F della tabella di Cockburn del caso d'uso disattiva/attiva categoria notifica

2.6.3 Caso d'Uso: Fare una controproposta di un offerta

TODO: Inserire descrizione

Gestione Annunci immobiliari

Aldo Baglio

Titolo	Copertina	Prezzo	Contratto	Proposte
Panoramic Attic in Monteverde Vecchio		€680/mese	Affitto	10

Gestione Proposte

In Trattativa

Nome	Email	Proposta	Controproposta	Azioni
Mario Rossi	mario.rossi@example...	€240.000	-	
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	
Luigi Verdi	luigi.verdi@example...	€230.000	-	
Anna Neri	anna.neri@example...	€240.000	-	

Proposte Accettate

Nome	Email	Proposta	Controproposta
Mario Rossi	mario.rossi@example...	€240.000	€245.000
Giulia Bianchi	giulia.bianchi@example...	€250.000	-

Cockburn: step 1/2/3/4



Gestione Annunci immobiliari

Aldo Baglio

Titolo	Copertina	Prezzo	Contratto	Proposte
Panoramic Attic in Monteverde Vecchio		€680/mese	Affitto	10

Gestione Proposte

In Trattativa

Nome	Email	Proposta	Controproposta	Azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	
Giulia Bir	giulia.bir@example...	-	-	
Luigi Verdi	luigi.verdi@example...	-	-	
Anna Neri	anna.neri@example...	-	-	

Proposte

Proposta	Controproposta
Offerta di Mario Rossi	€240.000

Inserisci proposta

€ 250.000

Invia controproposta

Cockburn: step 5/6/7



Gestione Annunci immobiliari

Aldo Baglio

Titolo	Copertina	Prezzo	Contratto	Proposte
Panoramic Attic in Monteverde Vecchio		€680/mese	Affitto	10

Gestione Proposte

In Trattativa

Nome	Email	Proposta	Controproposta	Azioni
Mario Rossi	mario.rossi@example...	€240.000	€245.000	
Giulia Bir	giulia.bir@example...	-	-	
Luigi Verdi	luigi.verdi@example...	-	-	
Anna Neri	anna.neri@example...	-	-	

Proposte

Proposta	Controproposta
FORM CONTROPROPOSTA	-

CONFERMA OPERAZIONE

Sei sicuro di effettuare questa controproposta

Annula Conferma

66

Invia controproposta

Cockburn: step 8/9/10

The screenshot displays a web-based application for managing real estate listings and offers. At the top, there's a header with the logo of "Dieti Estate 25 Portale Agenzia" and navigation links for "I Miei Annunci", "Le mie Proposte", "Messaggi Promozionali", "Team", and "Aldo Baglio". Below the header, the main content area is divided into two main sections:

Gestione Annunci immobiliari

This section shows a single listing for a "Panoramic Attic in Monteverde Vecchio" with a small thumbnail image, a price of €680/mese, and a status of "Affitto" (Rental). There are 10 offers listed, each with a checkmark, a pencil icon, and a delete icon.

Copertina	Prezzo	Contratto	Proposte
	€680/mese	Affitto	10

Gestione Proposte

This section is divided into three categories of proposals:

- In Trattativa**: Proposals under negotiation (indicated by a yellow circle).
- Proposte Accettate**: Accepted proposals (indicated by a green circle).
- Proposte Rifiutate**: Rejected proposals (indicated by a red circle).

Each category contains a table with columns for Name, Email, Proposta, and Controproposta. The accepted proposals table includes a column for azioni (actions), which includes a checkmark, a pencil icon, and a delete icon.

Nome	Email	Proposta	Controproposta	azioni
Mario Rossi	mario.rossi@example...	€240.000	€250.000	
Giulia Bianchi	giulia.bianchi@example...	€250.000	-	
Luigi Verdi	luigi.verdi@example...	€230.000	-	
Anna Neri	anna.neri@example...	€240.000	-	

Nome	Email	Proposta	Controproposta
Mario Rossi	mario.rossi@example...	€240.000	€245.000
Giulia Bianchi	giulia.bianchi@example...	€250.000	-
Luigi Verdi	luigi.verdi@example...	€230.000	-
Anna Neri	anna.neri@example...	€240.000	€245.000

Nome	Email	Proposta	Controproposta

Figure 2.25:

Cockburn: step 6.A



Cockburn: step 7.A

Figure 2.26: Mockup: Extension A della tabella di Cockburn del caso d'uso: Fare una controproposta a un'offerta.

Cockburn: step 7.B

Cockburn: step 8.B

Figure 2.27: Mockup: Extension B della tabella di Cockburn del caso d'uso: Fare una controproposta a un'offerta.

Cockburn: step 7.C/8.C

Cockburn: step 9.C

Figure 2.28: Mockup: Extension C della tabella di Cockburn del caso d'uso: Fare una controproposta a un'offerta.

2.6.4 Caso d'Uso: Registra nuovo agente

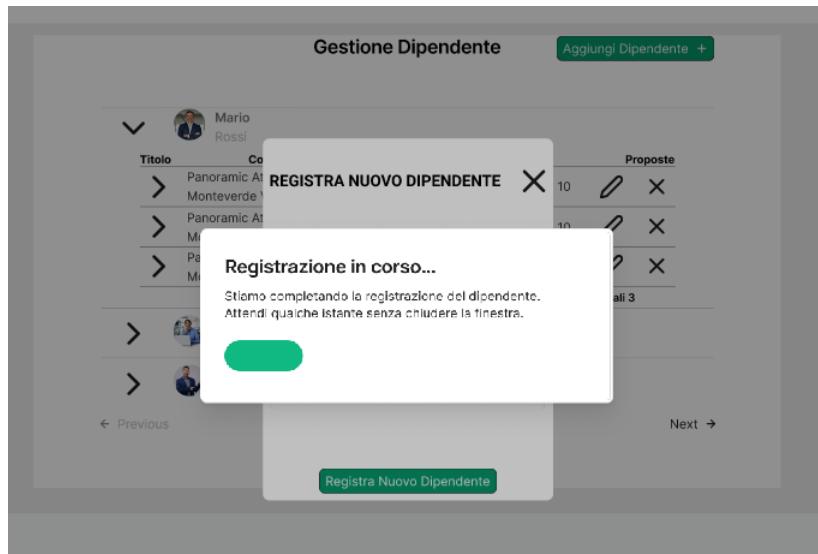
TODO: Inserire descrizione

Cockburn: step 1

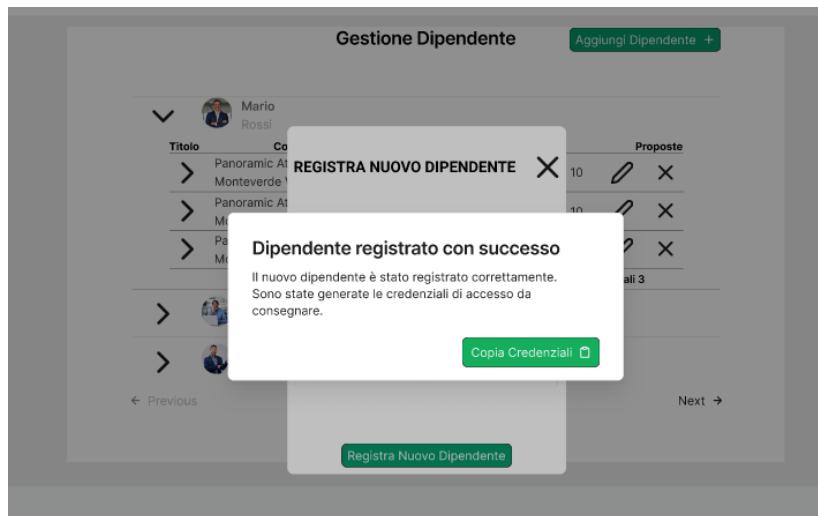


Cockburn: step 2/3/4

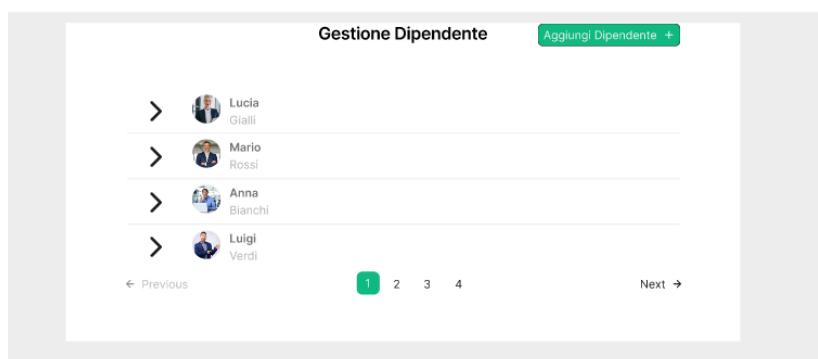




Cockburn: step 6/7/9



Cockburn: step 10



Cockburn: step 11

Gestione Dipendente Aggiungi Dipendente +

Mario Rossi	Titolo	Cop
> Panoramic Atti Monteverde V		
> Panoramic Atti Monteverde V		
> Panoramic Atti Monteverde V		
Anna Bianchi	Titolo	Cop
> Luigi Verdi		

← Previous Next →

REGISTRA NUOVO DIPENDENTE X

Nome:

Cognome:

Ruolo:

Registra Nuovo Dipendente →

Cockburn: step 4.A



Gestione Dipendente Aggiungi Dipendente +

Mario Rossi	Titolo	Cop
> Panoramic Atti Monteverde V		
> Panoramic Atti Monteverde V		
> Panoramic Atti Monteverde V		
Anna Bianchi	Titolo	Cop
> Luigi Verdi		

← Previous Next →

REGISTRA NUOVO DIPENDENTE X

Nome:

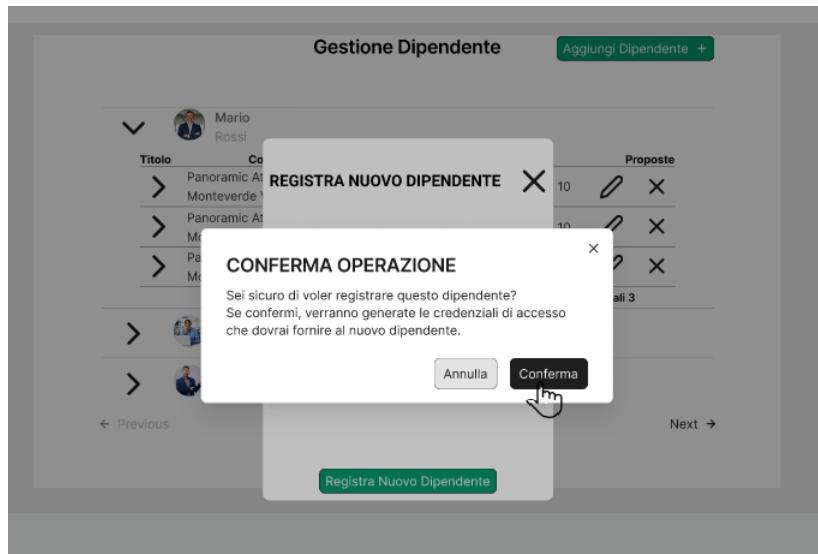
Cognome:

Ruolo:

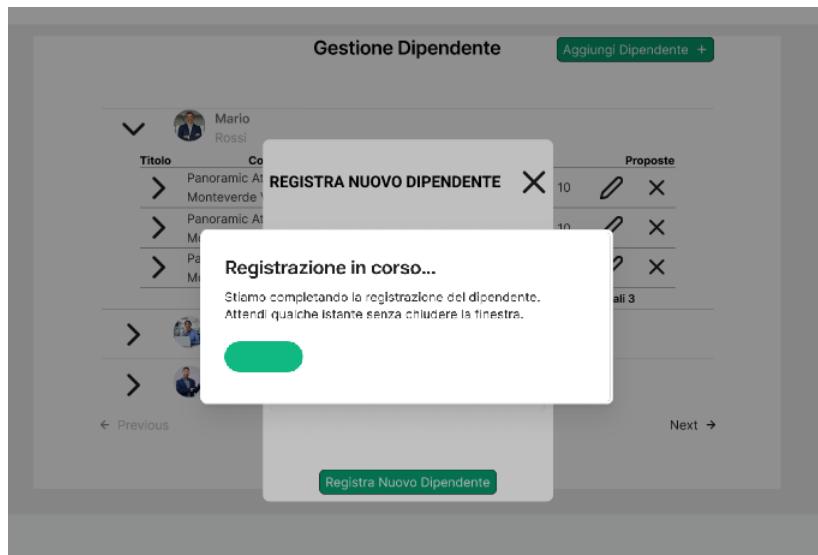
Registra Nuovo Dipendente →

Cockburn: step 5.A

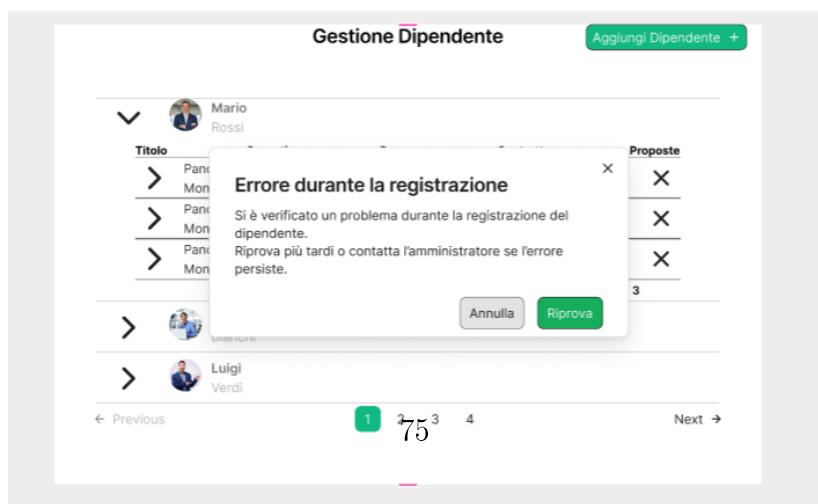
Figure 2.31: Mockup: Extension A della tabella di Cockburn del caso d'uso: Registra nuovo agente.



Cockburn: step 6.D



Cockburn: step 7.D



Chapter 3

Design del Sistema

3.1 Descrizione dell'architettura proposta

L'architettura del nostro server è stata progettata per rispondere alle esigenze richieste dal cliente, dove è stata prioritizzata la velocità di sviluppo e deployment senza sacrificare robustezza e sicurezza. Abbiamo adottato un'architettura monolitica basata su una struttura a livelli, che facilita la separazione delle responsabilità e garantisce una maggiore manutenibilità nel tempo.

In particolare, il sistema è suddiviso nei seguenti strati:

- Controllers: Responsabili della gestione delle richieste HTTP, questo livello si occupa di ricevere le chiamate dai client e di instradarle verso il livello successivo.
- Services: Qui risiede la logica di business. Il livello Service elabora le richieste provenienti dai Controllers, gestendo le regole e i processi applicativi.
- Repositories: Questo strato è dedicato alla persistenza dei dati, interagendo con il database attraverso l'uso di Hibernate e JPA per mappare le entità e gestire le operazioni di lettura/scrittura.

La scelta di un'architettura monolitica si giustifica con la necessità di ridurre la complessità iniziale e non rallentare lo sviluppo, mantenendo comunque una struttura modulare che potrà essere facilmente evoluta nel tempo – ad esempio, trasformando parti del sistema in microservizi qualora il progetto dovesse espandersi in futuro.

3.2 Descrizione e motivazione delle scelte tecnologiche adottate

Le tecnologie selezionate per lo sviluppo del backend sono state scelte in base a una combinazione di familiarità e facilità d'uso, con l'obiettivo di accelerare il processo di implementazione e garantire al contempo solidità e qualità del codice. In particolare:

- Java con Spring Boot: Abbiamo optato per questo framework grazie alle sue numerose configurazioni predefinite e utili estensioni come Lombok, che ci permettono di concentrarci sulla logica di business anziché su complesse configurazioni di sistema o codice ripetuto. Spring Boot consente di avviare rapidamente un'applicazione web robusta e scalabile.
- Hibernate: Per la gestione della persistenza, Hibernate si rivela una scelta efficace, in quanto astrae le operazioni di accesso al database e riduce significativamente il lavoro manuale nella scrittura di query SQL. Questo approccio rende il codice più leggibile e facilmente mantenibile.
- JWT (JSON Web Token): Per garantire un'autenticazione stateless e sicura, abbiamo deciso di utilizzare i JWT. Questa soluzione permette di gestire le sessioni degli utenti in maniera efficiente, differenziando eventuali ruoli o permessi e riducendo il carico sul server.
- Swagger: L'adozione di Swagger per la documentazione delle REST API agevola notevolmente il testing e la verifica degli endpoint, creando al contempo una documentazione ricca di informazioni utili per condividere informazioni critiche
- .

3.3 Descrizione dello schema per la persistenza dati

La gestione della persistenza dei dati è centrale per il nostro backend e si basa su un modello che sfrutta le potenzialità di Hibernate. PostgreSQL, per la loro affidabilità e diffusione nell'ecosistema Java.

Hibernate si occupa di:

Mappare le entità: Le classi Java rappresentano le entità del dominio, le quali vengono automaticamente correlate alle tabelle del database. Gestire le relazioni: Le associazioni fra le entità (uno-a-uno, uno-a-molti, molti-a-molti) sono gestite in modo trasparente, riducendo la complessità nella scrittura delle query. Ottimizzare le operazioni di lettura e scrittura: Grazie al supporto per caching e gestione delle transazioni, Hibernate semplifica l'interazione con il DBMS e contribuisce a mantenere il sistema performante e affidabile.

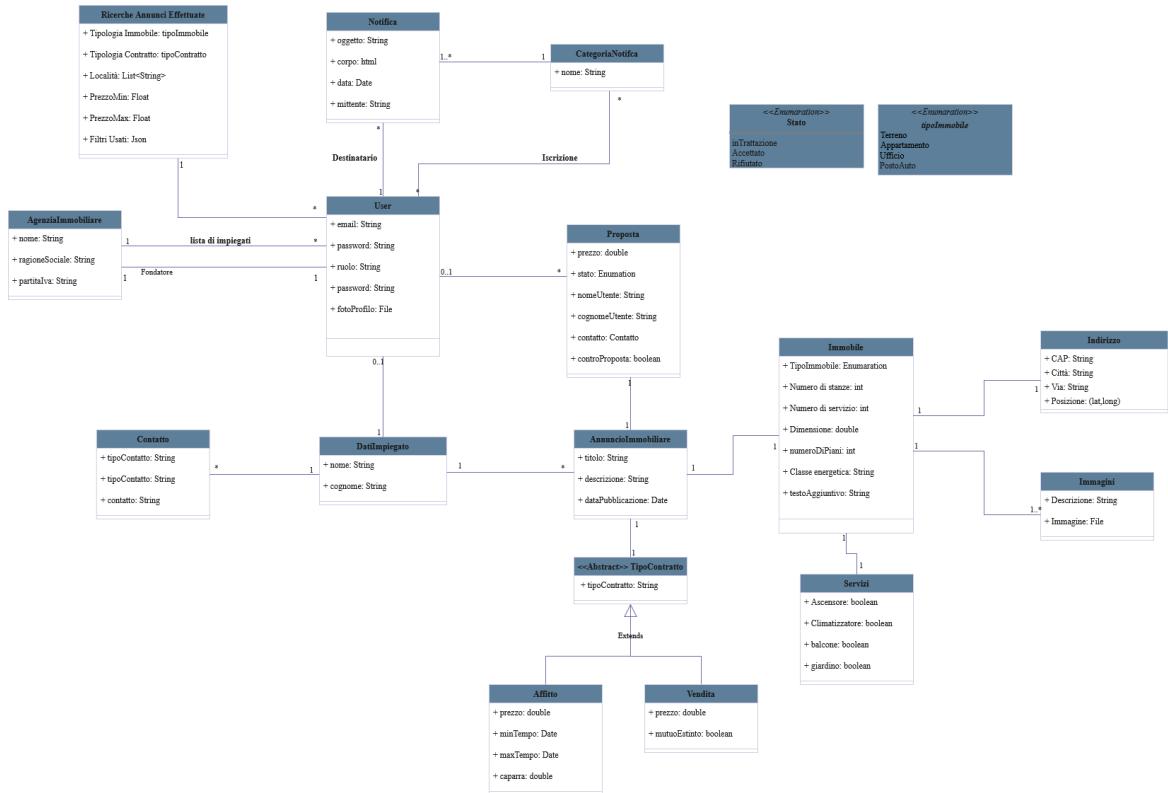


Figure 3.1: Diagramma delle classi usate

Chapter 4

Processo di sviluppo

Chapter 5

Testing e valutazione dell’usabilità.

Questo capitolo è suddiviso in due sezioni principali. Nella prima sezione vengono illustrate le strategie adottate per l’esecuzione dei test unitari utilizzando il framework xUnit, con particolare attenzione alla definizione delle classi di equivalenza, alla copertura dei percorsi logici e all’individuazione dei casi limite. Nella seconda sezione vengono presentate le valutazioni sull’usabilità del sistema, con analisi dei test condotti sugli utenti finali e delle metodologie adottate per misurare l’efficacia, l’efficienza e la soddisfazione nell’utilizzo dell’applicazione.

5.1 Progettazione e implementazione dei test unitari

Per la fase di unit testing sono stati selezionati i seguenti metodi da sottoporre a verifica:

- NewDipendenteResponse addDipendente(DipendenteRequest request, String aliasAgenzia)
- List<ContattoResponse> aggiungiContatto(ContattoRequest request)
- String changePassword(String oldPassword, String newPassword, String confirmPassword)
- List<User> utentiInteressati(CriteriDiRicercaUtenti request)

5.1.1 addDipendente(DipendenteRequest request, String aliasAgenzia)

Il metodo *addDipendente*, appartenente alla classe UserService, ha la responsabilità di aggiungere al sistema un nuovo dipendente, che può essere un agente o un manager. Per questo metodo è stata adottata una strategia di **Black Box** Testing, poiché l’obiettivo principale è verificare il comportamento del sistema in base agli stati e ai valori dei parametri in ingresso, senza considerare la logica interna dell’implementazione.

In particolare, si è ritenuto fondamentale analizzare le diverse condizioni in cui un dipendente viene correttamente registrato o rifiutato dal sistema.

A tal fine, è stata effettuata una suddivisione dei parametri di input in **classi di equivalenza**, al fine di individuare i casi validi e non validi da sottoporre a test. Di seguito si riportano le tabelle che documentano tale suddivisione:

ID Classe	Descrizione	Valore esempio	Validità
DRN1	L'insieme delle stringhe non vuote/lunghezza zero	request.nome==Roberto;	Valido
DRN2	L'insieme delle stringhe null/lunghezza zero	request.nome==" "	Non Valido

Table 5.1: Parametro DipendenteRequest.nome

ID Classe	Descrizione	Valore esempio	Validità
DRC1	L'insieme delle stringhe non vuote/lunghezza zero	request.cognome==Morosini;	Valido
DRC2	L'insieme delle stringhe null/lunghezza zero	request.cognome==null	Non Valido

Table 5.2: Parametro DipendenteRequest.cognome

ID Classe	Descrizione	Valore esempio	Validità
DRR1	Stringa di valore "AGENT"	request.ruolo=="AGENT";	Valido
DRR2	Stringa di valore "MANAGER"	request.ruolo=="MANAGER"	Valido
DRR3	tutti i valori non corrispondenti ad AGENT oppure a MANAGER	request.ruolo=="ADMIN"	Non Valido
DRR4	stringa null/lunghezza zero	request.ruolo==" "	Non Valido

Table 5.3: Parametro DipendenteRequest.ruolo

ID Classe	Descrizione	Valore esempio	Validità
AA1	L'insieme delle stringhe non null e non vuoti	aliasAgenzia==RobyImmobili	Valida
AA2	La stringa è null oppure è vuota	aliasAgenzia==null	Non valido

Table 5.4: Parametro aliasAgenzia

Per quanto riguarda la strategia di combinazione dei valori, è stato scelto l'approccio **R-WECT (Reduced Weak Equivalence Class Testing)**. Questa tecnica è considerata particolarmente robusta perché garantisce:

- il test di tutte le classi non valide, ognuna combinata con valori validi per gli altri parametri;
- il test di tutte le classi valide per ciascun parametro.

Infine, vengono riportati sia il codice sorgente dei test implementati sia l'evidenza del loro esito positivo, a conferma della correttezza delle funzionalità verificate

```

1 /**
2 * Test che copre le classi valide: DRN1 (Valida); DRC1 (Valida);
3 *          DRR1 (Valida); AA1 (Valida);
4 */
5 @Test
6 void addDipendenteTestAddAgente(){
7
8     DipendenteRequest dipendente = new DipendenteRequest("Roberto", "Spena", "AGENT");
9     String dominioAgenzia = "RobyImmobili";
10
11    when(userRepository.countByEmail("roberto.spena@robyimmobili.
12 .dietestate.com")).thenReturn(0);
13
14    Authority fakeAuthority = new Authority();
15    fakeAuthority.setAuthorityName(AuthorityName.AGENT);
16    when(authorityRepository.findByIdByName(AuthorityName.
17 AGENT)).thenReturn(Optional.of(fakeAuthority));
18
19    User fakeUser = new User();

```

```

17 fakeUser.setId(1); //id fittizio
18 DatiImpiegato fakeDati = new DatiImpiegato();
19 when(userRepository.save(Mockito.any(User.class))).thenReturn(fakeUser);
20 when(datiImpiegatoRepository.save(Mockito.any(DatiImpiegato.class))).thenReturn(fakeDati);
21
22 NewDipendeteResponse response = userService.addDipendete(
23     dipendente, dominioAgenzia);
24
25 //Controllo email generata
26 String emailPrevista = "roberto.spena@robyimmobili.
27     dietiestate.com";
28 assertEquals(emailPrevista, response.getUser().getEmail(), "Email non generata correttamente");
29
30 //Controllo password generata
31 String password = response.getPassword();
32 assertEquals(12, password.length(), "La password deve essere
33     lunga 12 caratteri");
34 assertTrue(
35     password.matches(".*[a-zA-Z].*"),
36     "La stringa deve contenere almeno una lettera");
37
38 assertTrue(
39     password.matches(".*[!@#$%^&*()\\\\\\-_+=\\\\\\\\[\\\\\\\\]{})\:,.<>?].*"),
40     "La stringa deve contenere almeno un carattere speciale");
41
42 assertTrue(
43     password.matches(".*[0-9].*"),
44     "La stringa deve contenere almeno una cifra");
45
46 //Controllo nomeVisualizzato
47 String nomeVisualizzato = response.getUser().
48     getNomeVisualizzato();
49 String nomeVisualizzatoPrevisto = "Agente Roberto S.";
50 assertEquals(nomeVisualizzato, nomeVisualizzatoPrevisto);
51
52 //Controllo foto profilo

```

```

50     String fotoProfilo = response.getUser().getUrlFotoProfilo();
51     assertTrue(fotoProfilo.matches("^https://dummyimage.com/.*"))
52         , "Url foto profilo non valido");

```

```

1 /**
2  * Test che copre le classi valide: DRN1 (Valida); DRC1 (
3  * Valida); DRR2(Valida); AA1(Valida);
4 */
5 @Test
6 void addDipendenteTestAddManager(){
7
8     DipendenteRequest dipendente = new DipendenteRequest("Raimondo", "Morosini", "MANAGER");
9     String dominioAgenzia = "RaimondoImmobili";
10
11    when(userRepository.countByEmail("raimondo.
12        morosini@raimondoimmobili.dietiestate.com")).thenReturn(0);
13
14    Authority fakeAuthority = new Authority();
15    fakeAuthority.setAuthorityName(AuthorityName.MANAGER);
16    when(authorityRepository.findByAuthorityName(
17        AuthorityName.MANAGER)).thenReturn(Optional.of(
18        fakeAuthority));
19
20    User fakeUser = new User();
21    fakeUser.setId(1); //id fittizio
22    DatiImpiegato fakeDati = new DatiImpiegato();
23    when(userRepository.save(Mockito.any(User.class))).thenReturn(fakeUser);
24    when(datiImpiegatoRepository.save(Mockito.any(
25        DatiImpiegato.class))).thenReturn(fakeDati);
26
27    NewDipendenteResponse response = userService.addDipendente(
28        dipendente, dominioAgenzia);
29
30    //Controllo email generata
31    String emailPrevista = "raimondo.
32        morosini@raimondoimmobili.dietiestate.com";
33    assertEquals(emailPrevista, response.getUser().getEmail()
34        , "Email non generata correttamente");

```

```

27
28     //Controllo password generata
29     String password = response.getPassword();
30     assertEquals(12, password.length(), "La password deve
31         essere lunga 12 caratteri");
32     assertTrue(
33         password.matches(".*[a-zA-Z].*"),
34         "La stringa deve contenere almeno una lettera"
35     );
36     assertTrue(
37         password.matches(".*[!@#$%^&*()\\\\\\/-_
38             =+\\\\\\\\[\\\\\\\\]{})|:,.<>?].*"),
39         "La stringa deve contenere almeno un carattere speciale"
40     );
41     assertTrue(
42         password.matches(".*[0-9].*"),
43         "La stringa deve contenere almeno una cifra"
44     );
45
46     //Controllo nomeVisualizzato
47     String nomeVisualizzato = response.getUser().
48         getNomeVisualizzato();
49     String nomeVisualizzatoPrevisto = "Manager Raimondo M.";
50     assertEquals(nomeVisualizzato, nomeVisualizzatoPrevisto)
51         ;
52 }

```

```

1 /**
2  * Test per coprire la classe nome non valida: DRN1 (Non
3  * Valida); DRC1 (Valida); DRR2 (Valida); AA1 (Valida);
4  */
5 @Test
6 void addDipendenteTestNomeNonValido(){
7
8     DipendenteRequest dipendente = new DipendenteRequest(
9         null, "Seppe", "MANAGER");

```

```

8     String dominioAgenzia = null;
9
10    Exception ex = assertThrows(IllegalArgumentException.
11        class, () -> userService.addDipendente(dipendente,
12        dominioAgenzia));
13
14    assertTrue(ex.getMessage().contains("Il parametro nome è
15        vuoto."));
16
17 }

```

```

1 /**
2 * Test per coprire la classe cognome non valida: DRN1 (
3     Valida); DRC1 (Non Valida); DRR2(Valida); AA1(Valida);
4 */
5 @Test
6 void addDipendenteTestCognomeNonValido(){
7
8     DipendenteRequest dipendente = new DipendenteRequest("Lorenzo", "", "MANAGER");
9     String dominioAgenzia = "LorenzoImmobili";
10
11    Exception ex = assertThrows(IllegalArgumentException.
12        class, () -> userService.addDipendente(dipendente,
13        dominioAgenzia));
14
15    assertTrue(ex.getMessage().contains("Il parametro
16        cognome è vuoto."));
17 }

```

```

1 /**
2 * Test per coprire la classe ruolo non valida: DRN1 (Valida)
3     ); DRC1 (Valida); DRR2(Non Valida); AA1(Valida);
4 */
5 @Test
6 void addDipendenteTestRuoloNonValido(){
7
8     DipendenteRequest dipendente = new DipendenteRequest("", "Sepe", "ADMIN");
9     String dominioAgenzia = "LorenzoImmobili";
10
11    Exception ex = assertThrows(IllegalArgumentException.
12        class, () -> userService.addDipendente(dipendente,
13        dominioAgenzia));
14
15    assertTrue(ex.getMessage().contains("Il parametro
16        ruolo è vuoto."));
17 }

```

```

    dominioAgenzia));
11
12     assertTrue(ex.getMessage().contains("Ruolo non valido:"));
13 }

```

```

1 /**
2 * Test per coprire la classe ruolo non esistente: DRN1 (Valida); DRC1 (Valida); DRR4 (Non Valida); AA1 (Valida);
3 */
4 @Test
5 void addDipendenteTestRuoloInesistente(){
6
7     DipendenteRequest dipendente = new DipendenteRequest("", "Sepe", "");
8     String dominioAgenzia = "LorenzoImmobili";
9
10    Exception ex = assertThrows(IllegalArgumentException.class, () -> userService.addDipendente(dipendente, dominioAgenzia));
11
12    assertTrue(ex.getMessage().contains("Ruolo inesistente"));
13 }

```

```

1 /**
2 * Test per coprire la classe aliasAgenzia non valida: DRN1 (Valida); DRC1 (Valida); DRR2 (Valida); AA1 (Non Valida);
3 */
4 @Test
5 void addDipendenteTestDominioAgenziaNonValido(){
6
7     DipendenteRequest dipendente = new DipendenteRequest("Lorenzo", "Sepe", "MANAGER");
8     String dominioAgenzia = null;
9
10    Exception ex = assertThrows(IllegalArgumentException.class, () -> userService.addDipendente(dipendente, dominioAgenzia));
11
12    assertTrue(ex.getMessage().contains("Il parametro aliasAgenzia è vuoto."));

```

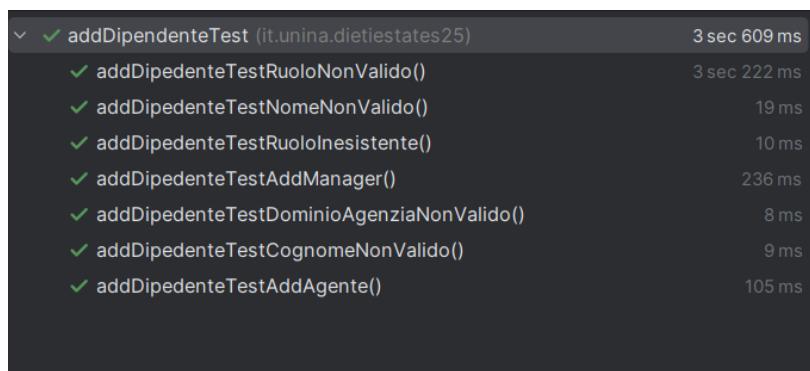


Figure 5.1: Screen che riporta l'esito positivo dei test

5.1.2 utentiInteressati(CriteriDiRicercaUtenti request)

Il metodo `utentiInteressati`, appartenente alla classe `RicercaAnnunciEffettuatiService`, ha la responsabilità di individuare tutti gli utenti potenzialmente interessati in base alle ricerche che essi hanno precedentemente effettuato. Questo metodo viene invocato quando il sistema deve inviare una notifica e necessita di filtrare i destinatari realmente pertinenti. Il parametro di ingresso, `CriteriDiRicercaUtenti`, contiene i campi relativi alle caratteristiche di un immobile. Se i criteri passati al metodo corrispondono a quelli di una ricerca salvata da un utente nel sistema, tale utente viene incluso nella lista dei destinatari risultanti.

Per la progettazione dei test unitari è stata adottata una **strategia di tipo black-box**, analogamente a quanto fatto per il metodo `addDipendente`. Tuttavia, la motivazione alla base di questa scelta è differente: mentre nel test di `addDipendente` l'obiettivo è verificare che il dipendente aggiunto possieda i valori corretti in funzione della richiesta di input, nel caso di `utentiInteressati` l'attenzione è rivolta al comportamento del metodo rispetto ai criteri di ricerca forniti. In particolare, si verifica che i valori non validi (ad esempio campi vuoti o non significativi) presenti nel parametro `CriteriDiRicercaUtenti` vengano opportunamente impostati a `null`, in modo da escluderli dal processo di matching con le ricerche salvate. Questo controllo è fondamentale per garantire che il sistema notifichi esclusivamente gli utenti realmente interessati.

Di seguito si riportano le tabelle che documentano tale suddivisione:

ID Classe	Descrizione	Valore esempio	Validità
A1	L'insieme delle stringhe che corrispondono a una località italiana	request.areaDiInteresse==Napoli	Valido
A2	L'insieme delle stringhe nulle/lunghezza zero	request.areaDiInteresse==null	Valido
A3	L'insieme delle stringhe che non corrispondono a una località italiana	request.areaDiInteresse==città inesistente	Valido

Table 5.5: Parametro request.areaDiInteresse

ID Classe	Descrizione	Valore esempio	Validità
C1	Un valore dell'enumerazione (AFFITTO,VENDITA)	request.tipoDiContrattoDiInteresse == AFFITTO;	Valido
C2	valore null	request.tipoDiContrattoDiInteresse == null	Valido

Table 5.6: Parametro request.tipoDiContrattoDiInteresse

ID Classe	Descrizione	Valore esempio	Validità
I1	Un valore dell'enumerazione	request.tipoDiImmobileDiInteresse == APPARTAMENTO;	Valido
I2	valore null	request.tipoDiContrattoDiInteresse == null	Valido

Table 5.7: Parametro request.tipoDiImmobileDiInteresse

ID Classe	Descrizione	Valore esempio	Validità
BMI1	qualsiasi numero positivo	request.budgetMin==100	Valido
BMI2	qualsiasi numero negativo	request.budgetMin== -100	Valido
BMI3	valore null	request.budgetMin==null	Valido

Table 5.8: Parametro request.budgetMin

ID Classe	Descrizione	Valore esempio	Validità
BMA1	qualsiasi numero positivo	request.budgetMan==600	Valido
BMA2	qualsiasi numero negativo	request.budgetMax== -600	Valido
BMA3	valore null	request.budgetMin==null	Valido

Table 5.9: Parametro request.budgetMax

ID Classe	Descrizione	Valore esempio	Validità
IG1	qualsiasi intero positivo	request.intervalloGiorni==10	Valida
II2	intero negativo	request.intervalloGiorni== -1	Valido
II3	Valore null	request.intervalloGiorni==null	Valido

Table 5.10: Parametro request.intervalloGiorniStoricoRicerche

Prima di ogni test vengono eseguiti i settaggi di default della request in modo che venga modificato solo il campo di interesse i ogni test. Mentre dopo eseguito ogni test viene verificato che alla repository, che interroga il db ed esegue il processo di corrispondenza, vengono passati i parametri che sono impostati nella request. Di seguito mostriamo il codice di tale settaggio.

```

1 @BeforeEach
2 void setup() {
3     ricercaAnnunciEffettuataService = new
4         RicercaAnnunciEffettuataService(
5             mockRicercaAnnunciEffettuataRepository, MockUserRepository
6             , new HashSet<String>());
7     request = CriteriDiRicercaUtenti.builder()
8         .intervalloGiorniStoricoRicerca(10)
9         .tipoDiContrattoDiInteresse(TipoContratto.AFFITTO)
10        .tipologiaDiImmobileDiInteresse(TipologiaImmobile.
11            APPARTAMENTO)
12        .budgetMin(BigDecimal.valueOf(100))
13        .budgetMax(BigDecimal.valueOf(600))
14        .areaDiInteresse("Napoli")
15        .build();
16    }

```

```

14
15 @AfterEach
16 void verifyRepositorySave() {
17     verify(mockRicercaAnnunciEffettuataRepository).
18         trovaUtentiPerCriteri(
19             eq(request.getBudgetMin()),
20             eq(request.getBudgetMax()),
21             eq(request.getAreaDiInteresse()),
22             eq(request.getTipoDiContrattoDiInteresse()),
23             eq(request.getTipologiaDiImmobileDiInteresse()),
24             any(LocalDateTime.class) // qualsiasi valore va bene
25         );
26
27 }

```

La suddivisione in classi di equivalenza dei parametri di questo metodo non presenta classi non valide, poiché, come spiegato, i campi con valori non validi vengono impostati a un valore specifico, come ad esempio `null`. La strategia utilizzata per la combinazione dei test consiste nel verificare tutte le classi valide con parametri validi e, successivamente, nell'eseguire un test per ciascun caso in cui un solo campo assume un valore non valido mentre tutti gli altri rimangono validi.

```

1 /**
2  * test con area di interesse Italia non deve modificare
3  * nulla tranne l'area di interesse che deve settarla a null
4 */
5 @Test
6 void areaDiInteresseItaliaShouldSetNullAreaDiInteresse() {
7     request.setAreaDiInteresse("Italia");
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10    assertTrue(request.getAreaDiInteresse() == null);
11 }

```

```

1 /**
2  * test che controlla che se il budget max è minore del
3  * budget min, li scambia
4 */
5 @Test
6 void budgetMaxLessThanBudgetMinShouldSwapThem() {
7     request.setBudgetMin(BigDecimal.valueOf(600));
8     request.setBudgetMax(BigDecimal.valueOf(100));

```

```
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10    assertTrue(request.getBudgetMin().equals(BigDecimal.
11                  valueOf(100)));
12    assertTrue(request.getBudgetMax().equals(BigDecimal.
13                  valueOf(600)));
14 }
```

```
1 /**
2 * Test che controlla che se la città non esiste, imposta a
3     null il campo area di interesse della request
4 */
5 @Test
6 void invalidCityShouldSetNullAreaDiInteresse() {
7     request.setAreaDiInteresse("Città Inesistente");
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10    assertTrue(request.getAreaDiInteresse()==null);
11 }
```

```
1 /**
2 * test che controlla che se budget min è negativo, lo
3     imposta a null
4 */
5 @Test
6 void negativeBudgetMinShouldSetNullBudgetMin() {
7     request.setBudgetMin(BigDecimal.valueOf(-100));
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10
11    assertTrue(request.getBudgetMin()==null);
12 }
```

```
1 /**
2 * test che controlla che se budget max è negativo, lo
3     imposta a null
4 */
5 @Test
6 void negativeBudgetMaxShouldSetNullBudgetMax() {
7     request.setBudgetMax(BigDecimal.valueOf(-100));
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10 }
```

```

8     assertTrue(request.getBudgetMax() == null);
9 }
```

```

1 /**
2 * test che controlla se l' intervallo di giorni è negativo,
3 * lo imposta setta a 7
4 */
5 @Test
6 void deltaDaysLessThanZeroShouldSet7() {
7     request.setIntervalloGiorniStoricoRicerca(0);
8     ricercaAnnunciEffettuataService.utentiInteressati(
9         request);
10    assertTrue(request.getIntervalloGiorniStoricoRicerca()
11        == 7);
12 }
```

✓ CriteriDiRicercaTest (it.unina.dietestates25)	3 sec 334 ms
✓ invalidCityShouldSetNullAreaDiInteresse()	3 sec 116 ms
✓ negativeBudgetMinShouldSetNullBudgetMin()	47 ms
✓ negativeBudgetMaxShouldSetNullBudgetMax()	52 ms
✓ areaDiInteresseItaliaShouldSetNullAreaDiInteresse()	31 ms
✓ validRequestShouldNotChangeAnything()	21 ms
✓ deltaDaysLessThanZeroShouldSet7()	20 ms
✓ budgetMaxLessThanBudgetMinShouldSwapThem()	20 ms
✓ allNullParametersShouldSetDaysTo7()	27 ms

Figure 5.2: Screen che riporta l'esito positivo dei test

5.2 Test di usabilità

I **test di usabilità** rappresentano una fase essenziale nel processo di sviluppo di un’interfaccia utente, consentendo di valutare l’efficacia, l’efficienza e la soddisfazione dell’utente nell’interazione con il sistema. In particolare, l’uso di mockup interattivi offre la possibilità di raccogliere feedback sulle scelte di design prima ancora della fase di sviluppo, riducendo i costi di eventuali revisioni e migliorando la qualità dell’esperienza utente.

L’obiettivo principale di questi test è identificare eventuali problemi di navigazione, ambiguità nelle interazioni o difficoltà nella comprensione delle funzionalità, al fine di ottimizzare l’interfaccia prima del rilascio definitivo.

1. Tempo medio per completare un Task

$$T_{\text{medio}} = \frac{\sum_{i=1}^n T_i}{n} \quad (5.1)$$

dove T_i è il tempo impiegato dall'utente i per completare il task e n è il numero totale di utenti.

2. Tasso di completamento di un Task

$$T_{\text{completamento}} = \frac{U_{\text{completati}}}{U_{\text{totali}}} \times 100 \quad (5.2)$$

dove $U_{\text{completati}}$ è il numero di utenti che hanno completato il task e U_{totali} è il numero totale di utenti che hanno provato il task.

3. Errore Medio per Task

$$E_{\text{medio}} = \frac{\sum_{i=1}^n E_i}{n} \quad (5.3)$$

dove E_i è il numero di errori commessi dall'utente i durante il task.

4. Customer Satisfaction Score

$$CSAT = \frac{\sum_{i=1}^n S_i}{n} \times 100 \quad (5.4)$$

dove S_i è il punteggio di soddisfazione dato dall'utente i e n è il numero totale di risposte raccolte.

5. System Usability Scale (SUS)

Il **System Usability Scale (SUS)** è un metodo standardizzato, introdotto da **John Brooke** nel 1986, utilizzato per misurare l'usabilità di un prodotto attraverso un questionario composto da **10 affermazioni**. Gli utenti rispondono utilizzando una **scala Likert a 5 punti**, esprimendo il loro grado di accordo o disaccordo. Il punteggio complessivo, che varia da 0 a 100, fornisce una misura quantitativa dell'usabilità percepita, consentendo di confrontare i risultati con benchmark consolidati.

Questionario relativo al mockup "Creazione nuovo annuncio"

Di seguito presentiamo il questionario utilizzato per valutare il mockup del processo di creazione di un annuncio immobiliare. Ogni domanda prevede cinque opzioni di risposta:

- Per niente d'accordo
- Poco d'accordo
- Né d'accordo né in disaccordo
- Abbastanza d'accordo
- Molto d'accordo

Ai fini del calcolo del punteggio **SUS**, alle risposte viene assegnato un valore da **1** a **5**, dove la prima opzione corrisponde a 1 punto e l'ultima a 5 punti.

1. La compilazione dei campi divisi in step ha reso il processo di creazione più intuitivo e meno stancante.

Scopo: Verificare se la suddivisione del form in più passaggi migliora l'esperienza utente, evitando un sovraccarico cognitivo e rendendo la compilazione più fluida.

2. Il bottone “Avanti” è posizionato in modo poco visibile e difficile da individuare.

Scopo: Valutare la visibilità e l'intuitività del pulsante che consente di procedere nella compilazione, elemento essenziale per garantire una navigazione chiara e senza interruzioni.

3. I campi a scelta (non liberi) mi hanno aiutato a capire meglio il tipo di informazioni richieste e a velocizzare la compilazione.

Scopo: Analizzare se l'uso di menu a tendina o opzioni predefinite aiuta a ridurre incertezze, errori e tempi di compilazione rispetto ai campi di testo libero.

4. Il numero di campi richiesti è eccessivo o insufficiente, rendendo il form poco bilanciato.

Scopo: Ottenere un feedback sulla quantità di informazioni richieste, bilanciando completezza e semplicità d'uso, evitando di rendere il processo troppo lungo o complesso.

5. I colori utilizzati sono gradevoli e non affaticano la lettura.

Scopo: Valutare se la combinazione di colori scelta favorisce una buona leggibilità e un'esperienza visiva piacevole, senza creare affaticamento visivo.

6. L'importazione e la gestione delle foto dell'annuncio risultano poco intuitive.

Scopo: esaminare la semplicità e l'efficacia del meccanismo di caricamento delle immagini, una funzionalità chiave nella creazione di annunci immobiliari.

7. Trovo utile l'anteprima dell'annuncio prima di confermare la pubblicazione.

Scopo: Capire se l'anteprima fornisce valore aggiunto agli utenti, permettendo loro di controllare e correggere eventuali errori prima della pubblicazione definitiva.

8. L'avviso in caso di un annuncio in sospeso è poco chiaro o inutile.

Scopo: Testare la comprensibilità e l'efficacia del messaggio di avviso per evitare che l'utente perda dati o crei annunci duplicati.

9. In caso di errore (compilazione errata o parziale), i messaggi di errore sono posizionati bene e mi aiutano a capire rapidamente dove ho sbagliato.

Scopo: Valutare la chiarezza dei messaggi di errore e il loro posizionamento, per garantire che l'utente possa correggere facilmente eventuali problemi.

10. Nel complesso, non sono soddisfatto dell'esperienza di creazione di un annuncio.

Scopo: Raccogliere un'indicazione generale sul livello di soddisfazione dell'utente rispetto all'intero processo, fornendo una misura qualitativa dell'usabilità percepita.

Risultato SUS ottenuto attraverso i feedback raccolti

Il System Usability Scale (SUS) viene calcolato seguendo questi passaggi:

1. Assegnazione punteggio

Ogni domanda viene valutata su una scala da 1 a 5:

- 1 = Per niente d'accordo
- 2 = Poco d'accordo
- 3 = Né d'accordo né in disaccordo
- 4 = Abbastanza d'accordo
- 5 = Molto d'accordo

Le 10 domande del questionario SUS sono di due tipi:

- **Domande dispari (1, 3, 5, 7, 9):** indicano usabilità positiva
- **Domande pari (2, 4, 6, 8, 10):** indicano usabilità negativa

2. Calcolo del punteggio per ogni domanda

- **Per le domande dispari:** Punteggio=(Risposta-1)
- **Per le domande pari:** Punteggio=(5-Risposta)

3. Sommiamo tutti i punteggi ottenuti

Otteniamo un punteggio complessivo che va da 0 a 40.

4. Moltiplichiamo per 2,5: SUS=(somma dei punteggi)×2.5.

Il punteggio finale sarà compreso tra 0 e 100, ma non rappresenta una percentuale.

Interpretazione del punteggio SUS

- **Sopra 80** → Ottima usabilità
- **Tra 70 e 80** → Buona usabilità
- **Tra 50 e 70** → Accettabile, ma migliorabile
- **Sotto 50** → Problemi di usabilità significativi

Il questionario è stato compilato da quattro agenti immobiliari, i principali attori di questo caso d'uso. Ciascun agente proviene da un'agenzia immobiliare diversa e appartiene a una fascia d'età differente, al fine di garantire un campione più eterogeneo e realistico.

Utente 1:

- **Risposte a domande dispari**
 - Domanda 1: molto d'accordo = 5-1 = 4
 - Domanda 3: molto d'accordo = 5-1 = 4
 - Domanda 5: molto d'accordo = 5-1 = 4
 - Domanda 7: molto d'accordo = 5-1 = 4
 - Domanda 9: molto d'accordo = 5-1 = 4
- **Risposte a domande pari**
 - Domanda 2: per niente d'accordo = 5-1 = 4
 - Domanda 4: poco d'accordo = 5-2 = 3
 - Domanda 6: poco d'accordo = 5-2 = 3
 - Domanda 8: per niente d'accordo = 5-1 = 4
 - Domanda 10: per niente d'accordo = 5-1 = 4

- Totale punteggio utente 1: $38*2.5 = 95$

Utente 2:

- Risposte a domande dispari

- Domanda 1: molto d'accordo = 5-1 = 4
- Domanda 3: molto d'accordo = 5-1 = 4
- Domanda 5: molto d'accordo = 5-1 = 4
- Domanda 7: abbastanza d'accordo = 4-1 = 3
- Domanda 9: molto d'accordo = 5-1 = 4

- Risposte a domande pari

- Domanda 2: per niente d'accordo = 5-1 = 4
- Domanda 4: per niente d'accordo = 5-1 = 4
- Domanda 6: poco d'accordo = 5-2 = 3
- Domanda 8:né d'accordo né in disaccordo = 5-3 = 2
- Domanda 10:per niente d'accordo = 5-1 = 4

- Totale punteggio utente 2: $36*2.5 = 90$

Utente 3:

- Risposte a domande dispari

- Domanda 1:abbastanza d'accordo = 4-1 = 3
- Domanda 3: molto d'accordo = 5-1 = 4
- Domanda 5: molto d'accordo = 5-1 = 4
- Domanda 7: molto d'accordo = 5-1 = 4
- Domanda 9: molto d'accordo = 5-1 = 4

- Risposte a domande pari

- Domanda 2: poco d'accordo = 5-2 = 3
- Domanda 4: poco d'accordo= 5-2 = 3
- Domanda 6: per niente d'accordo = 5-1 = 4
- Domanda 8: poco d'accordo = 5-2 = 3
- Domanda 10: per niente d'accordo = 5-1 = 4

- Totale punteggio utente 3: $36*2.5 = 90$

Utente 4:

- Risposte a domande dispari

- Domanda 1: molto d'accordo = 5-1 = 4
- Domanda 3: molto d'accordo = 5-1 = 4
- Domanda 5: né d'accordo né in dissacordo = 3-1 = 2
- Domanda 7: molto d'accordo = 5-1 = 4
- Domanda 9: molto d'accordo = 5-1 = 4

- Risposte a domande pari

- Domanda 2: poco d'accordo = 5-2 = 3
- Domanda 4: né 'accordo né in disaccordo = 5-3 = 2
- Domanda 6: per niente d'accordo = 5-1 = 4
- Domanda 8: per niente d'accordo = 5-1 = 4
- Domanda 10: per niente d'accordo = 5-1 = 4

- Totale punteggio utente 4: $35*2.5 = 87.5$

Media punteggio SUS = $95+90+90+87.5/4 = 90.62$. Il punteggio è nettamente superiore a 80 il che indica un **ottima usabilità**.

Chapter 6

Fonti

Bibliography

- [1] Nielsen, J. (1995). *10 Heuristics for User Interface Design*. Nielsen Norman Group. Disponibile su: <https://www.nngroup.com/articles/ten-usability-heuristics/>.
- [2] Norman, D. A. (1988). *The Psychology of Everyday Things*. Basic Books. (Riedizione: *The Design of Everyday Things*, 2013).
- [3] Miller, G. A. (1956). *The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information*. Psychological Review, 63(2), 81-97.
- [4] Shneiderman, B., & Plaisant, C. (2004). *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (4^a ed.). Pearson.
- [5] Wickens, C. D. (2008). *Multiple Resources and Mental Workload*. Human Factors, 50(3), 449-455.
- [6] Pieters, R., & Wedel, M. (2004). *Attention Capture in Advertising: Brand, Pictorial, and Text-Size Effects*. Journal of Marketing, 68(2), 36-50.