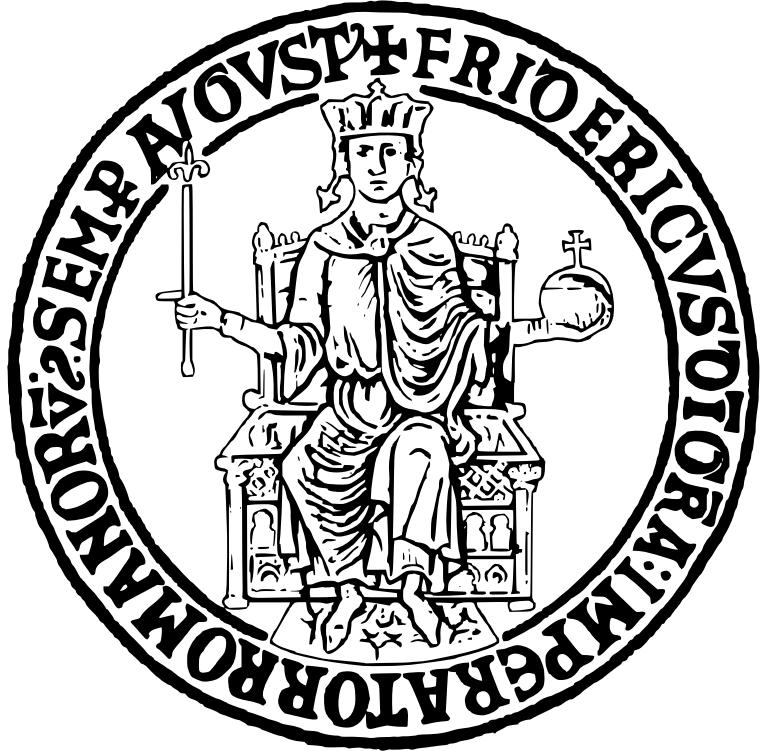


UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II



SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE
DELL'INFORMAZIONE

CORSO DI LAUREA TRIENNALE IN INFORMATICA

DietiDeals24

Autori

Salvatore Franzese N86003142
Claudio Riccio N86003291

Docenti

Prof. Sergio di Martino
Prof. Francesco Cutugno
Prof. Luigi Libero Lucio Starace
Prof. Sergio di Meglio

Anno Accademico 2023-2024

Questa pagina è stata intenzionalmente lasciata in bianco

Indice

1	Introduzione	5
2	Requisiti Software	6
2.1	Analisi dei Requisiti	6
2.1.1	Modellazione casi d'uso	6
2.1.1.1	Accesso alla piattaforma	6
2.1.1.2	Ricerca Aste	7
2.1.1.3	Partecipazione alle aste	8
2.1.1.4	Creazione aste	9
2.1.1.5	Personalizzazione profilo	10
2.1.2	Prototipazione visuale via Mock-up dell'interfaccia	11
2.1.2.1	Mock-up01: Accesso	11
2.1.2.2	Mock-up02: Registrazione	12
2.1.2.3	Mock-up03: Home	13
2.1.2.4	Mock-up04: Ricerca	14
2.1.2.5	Mock-up05: Ricerca per categoria	15
2.1.2.6	Mock-up06: Dettagli asta	16
2.1.2.7	Mock-up07: Profilo venditore	17
2.1.2.8	Mock-up08: Popup conferma offerta	18
2.1.2.9	Mock-up09: Popup scelta tipo di asta	18
2.1.2.10	Mock-up10: Creazione asta all'inglese	19
2.1.2.11	Mock-up11: Creazione asta al ribasso	20
2.1.2.12	Mock-up12: Notifiche	21
2.1.2.13	Mock-up13: Popup conferma creazione asta	22
2.1.3	Casi d'uso significativi	23
2.1.3.1	Creazione di un'asta all'inglese	23
2.1.3.2	Presenta offerta per asta all'inglese	25
2.1.4	Individuazione del target degli utenti	27
2.1.4.1	Personas 1: Sofia	27
2.1.4.2	Personas 2: Luca	28
2.1.4.3	Personas 3: Alessandro	28
2.1.5	Valutazione dell'usabilità a priori	29
2.1.6	Glossario	31
2.2	Specifiche dei requisiti	33
2.2.1	Entità	33
2.2.2	Class Diagram di Analisi	34

2.2.3	Legenda dei colori	34
2.2.3.1	Login/Registrazione	34
2.2.3.2	Ricerca	35
2.2.3.3	Partecipazione ad un'asta	36
2.2.3.4	Creazione asta	37
2.2.3.5	Personalizzazione profilo utente	38
2.2.3.6	Notifica	39
2.2.4	Sequence Diagram	40
2.2.4.1	Crea asta all'inglese	40
2.2.4.2	Presenta offerta per asta all'inglese	41
2.2.5	Statechart Diagram	42
2.2.5.1	Crea asta all'inglese	42
2.2.5.2	Presenta offerta per asta all'inglese	43
3	Design del Sistema	44
3.1	Analisi Architetturale	44
3.1.1	Database	45
3.1.2	JPA Hibernate	45
3.1.3	SpringBoot e Retrofit	45
3.1.4	Docker	45
3.1.5	Docker Compose	46
3.1.6	Amazon EC2	46
3.1.7	API REST	46
3.1.7.1	DietiUserAuth	46
3.1.7.2	DietiUser	47
3.1.7.3	Notification	47
3.1.7.4	EnglishAuction	47
3.1.7.5	DownwardAuction	47
3.1.7.6	Offer	48
3.1.7.7	Image	48
3.2	Class Diagram di Design	49
3.2.1	Entità	49
3.2.2	Login e Registrazione	50
3.2.3	Creazione asta al ribasso	51
3.2.4	Creazione asta all'inglese	52
3.2.5	Piazzamento offerta	53
3.2.6	Notifiche	54
3.2.7	Ricerca	55
3.2.8	Personalizzazione profilo	56
3.3	Sequence diagram di design	57
3.3.1	Crea asta all'inglese	57
3.3.2	Presenta offerta per asta all'inglese	58
4	Codice sorgente	59
4.1	Docker	59
4.1.1	Docker Compose	59
4.1.2	Dockerfile	59

4.2	GitHub	60
4.3	Report di qualità	61
5	Testing	62
5.1	Codice JUnit per Unit Testing	62
5.1.1	Metodo convertFieldsToMilliseconds	62
5.1.2	Metodo updateData	64
5.2	Valutazione dell'usabilità sul campo	65
5.2.1	Analytics	65
5.2.1.1	Google Analytics	65
5.2.1.2	Report	65
6	Contributori	68

Capitolo 1

Introduzione

DietiDeals24 è una piattaforma per la gestione di aste online. Il sistema consiste in un'applicazione mobile attraverso cui gli utenti possono fruire delle funzionalità del sistema in modo intuitivo, rapido e piacevole. Le principali funzionalità offerte da *DietiDeals24* sono indicate di seguito:

- Un utente può registrarsi mediante nome, cognome, email e password. Successivamente, una volta effettuato il login, può personalizzare il proprio profilo con una immagine, una breve biografia, l'area geografica e dei riferimenti ai propri social.
- Un utente può creare due tipologie di aste:
 - **Asta all'inglese:** caratterizzata da una base d'asta iniziale pubblica, specificata dal venditore al momento della creazione dell'asta, da un intervallo di tempo fisso per presentare nuove offerte (di default 1 ora), e da una soglia di rialzo (di default 10€). Gli acquirenti possono presentare un'offerta per il prezzo corrente. Quando viene presentata un'offerta, il timer per la presentazione di nuove offerte viene resettato. Quando il timer raggiunge lo zero senza che siano presentate nuove offerte, l'ultima offerta si aggiudica il bene/servizio in vendita, e il venditore e gli acquirenti che hanno partecipato all'asta visualizzano una notifica.
 - **Asta al ribasso:** caratterizzata da un prezzo iniziale elevato specificato dal venditore, da un timer per il decremento del prezzo (di default 1 ora), da un importo, in €, per ciascun decremento, e da un prezzo minimo (segreto) a cui vendere il prodotto/servizio. Il prodotto/servizio sarà in vendita al prezzo iniziale stabilito dal venditore. Al raggiungimento del timer, il prezzo verrà decrementato dell'importo previsto. Il primo compratore a presentare un'offerta si aggiudica il prodotto/servizio. Se il prezzo viene decrementato fino a raggiungere il prezzo minimo segreto, l'asta viene considerata fallita e il venditore visualizza una notifica.

Capitolo 2

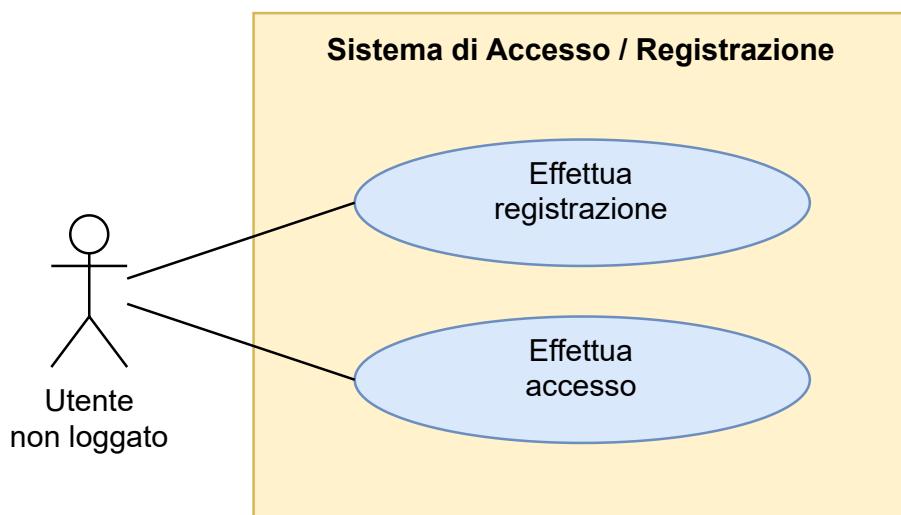
Requisiti Software

2.1 Analisi dei Requisiti

2.1.1 Modellazione casi d'uso

Di seguito verranno mostrati gli **Use Case Diagram** al fine di rappresentare le interazioni tra gli **utenti** (attori) e le **funzionalità** del sistema (casi d'uso).

2.1.1.1 Accesso alla piattaforma



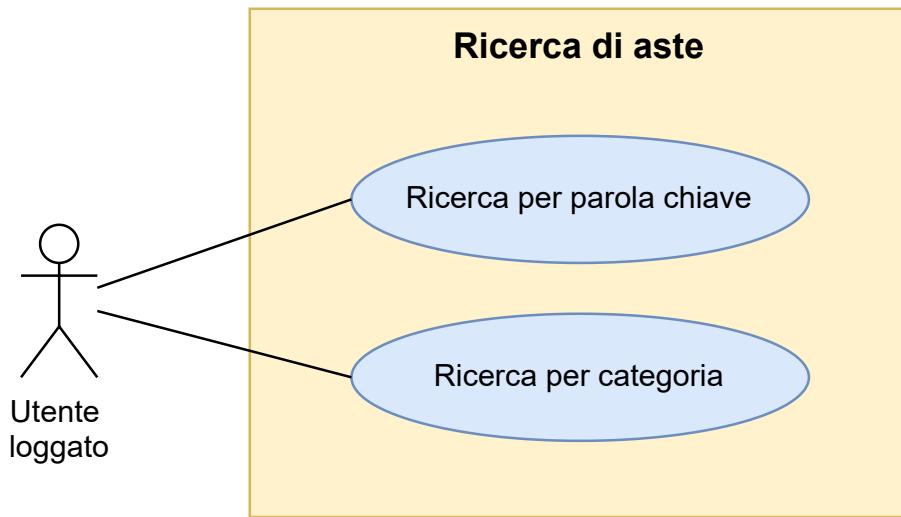
E' possibile **accedere** a DietiDeals24 solo dopo essersi registrati. La **registrazione** avviene inserendo:

- Nome;
- Cognome;
- Email;
- Password.

L'accesso avviene inserendo:

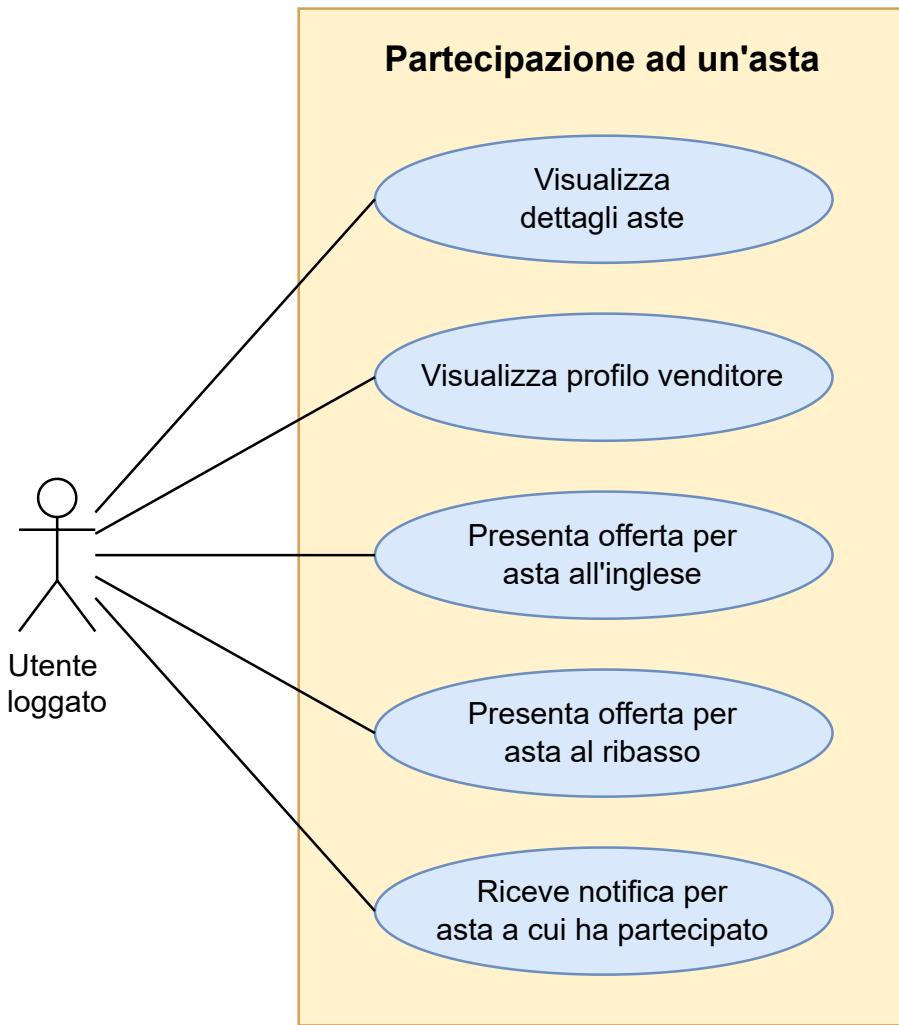
- Email;
- Password.

2.1.1.2 Ricerca Aste



La **ricerca** di un'asta può avvenire tramite l'inserimento di una **parola chiave** o filtrando le aste esistenti per **categoria**.

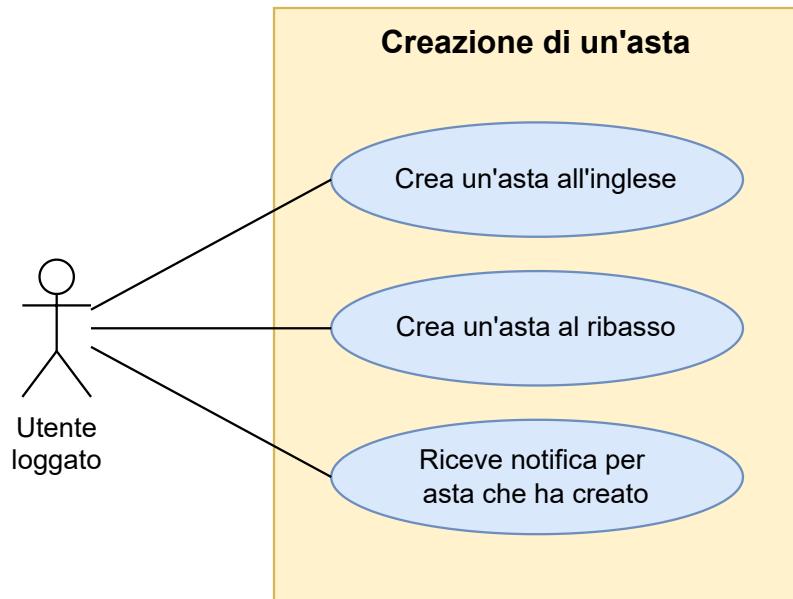
2.1.1.3 Partecipazione alle aste



Questo Use Case descrive le azioni inerenti alla partecipazione ad un'asta che un'utente loggato può effettuare. In particolare:

- **Visualizza dettagli aste:** permette di visualizzare tutte le informazioni relative all'asta inserita dall'utente che ha creato l'asta;
- **Visualizza profilo venditore:** permette all'utente, che sta visualizzando un'asta, di visualizzare le informazioni dell'utente che l'ha creata;
- **Presenta offerta per asta all'inglese;**
- **Presenta offerta per asta al ribasso;**
- **Riceve notifica per asta a cui ha partecipato:** alla conclusione di un'asta, i partecipanti riceveranno una notifica; lo stato della notifica può essere "PERSA" oppure "VINTA".

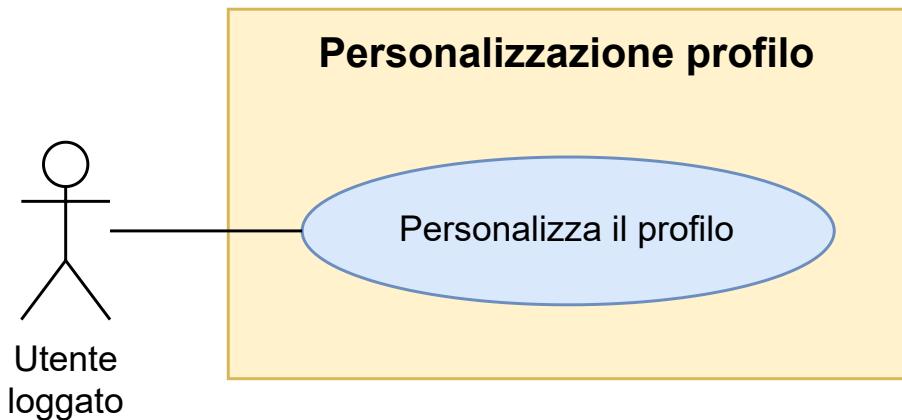
2.1.1.4 Creazione aste



Questo Use Case descrive le azioni inerenti alla creazione di un'asta che un utente loggato può effettuare. In particolare:

- **Crea un'asta all'inglese**, caratterizzata da:
 - Titolo;
 - Descrizione;
 - Categoria;
 - Immagine;
 - Prezzo iniziale;
 - Importo di rialzo;
 - Durata.
- **Crea un'asta al ribasso**, caratterizzata da:
 - Titolo;
 - Descrizione;
 - Categoria;
 - Immagine;
 - Prezzo iniziale;
 - Importo di decremento;
 - Intervallo per il decremento;
 - Prezzo minimo.
- **Riceve notifica per asta che ha creato**: alla conclusione di un'asta, il creatore di essa riceverà una notifica; lo stato può essere "FALLITA" oppure "CONCLUSUA".

2.1.1.5 Personalizzazione profilo



Un utente loggato può **personalizzare** il proprio profilo, modificando:

- Immagine di profilo;
- Nome;
- Cognome;
- Password;
- Breve biografia;
- Area geografica;
- Link ai propri profili social.

2.1.2 Prototipazione visuale via Mock-up dell'interfaccia

Di seguito verrano riportati i mock-up necessari alla comprensione dei *casi d'uso*, la cui prototipazione visuale completa è accessibile al seguente [link](#).

Nota: il design dell'applicativo finale potrebbe aver subito delle variazioni.

2.1.2.1 Mock-up01: Accesso

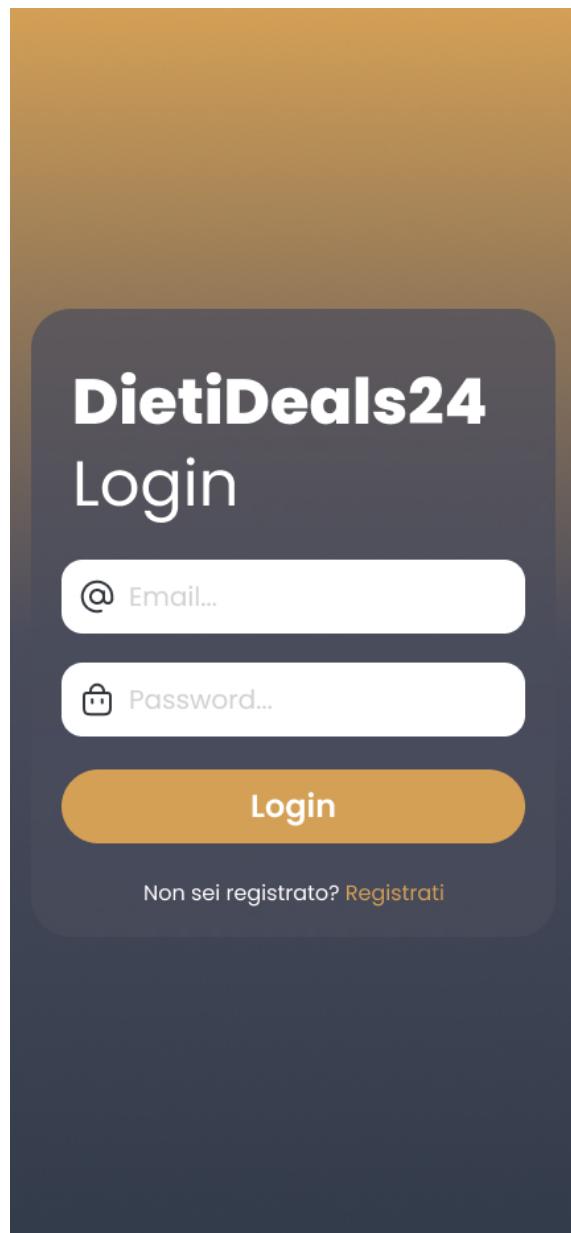


Figura 2.1: Mock-up01 Accesso

2.1.2.2 Mock-up02: Registrazione

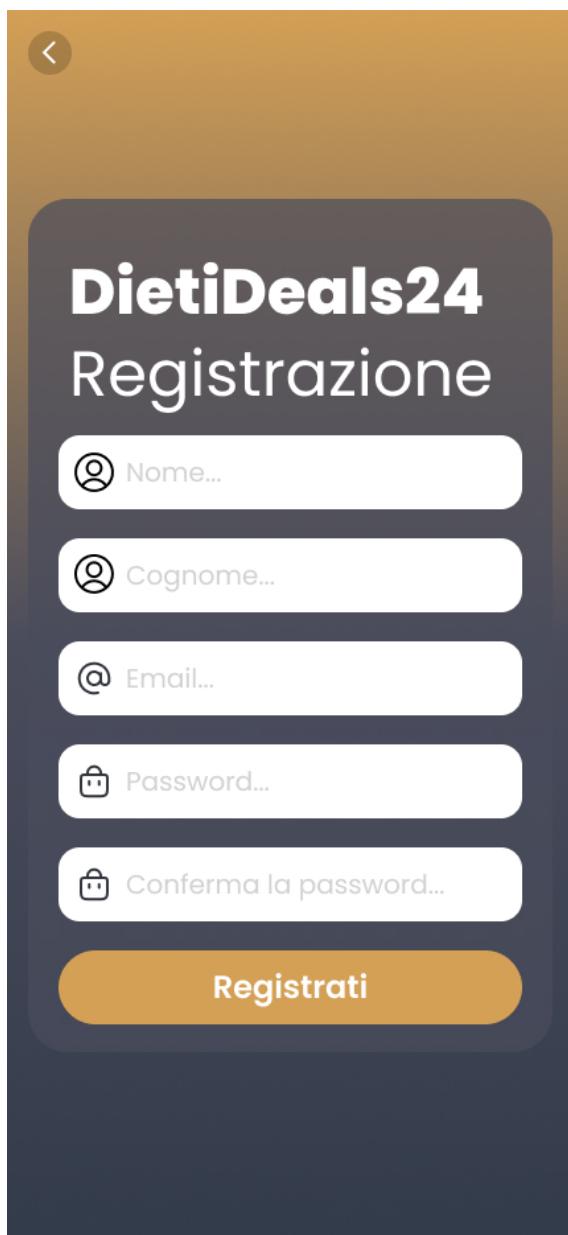


Figura 2.2: Mock-up02 Registrazione

2.1.2.3 Mock-up03: Home

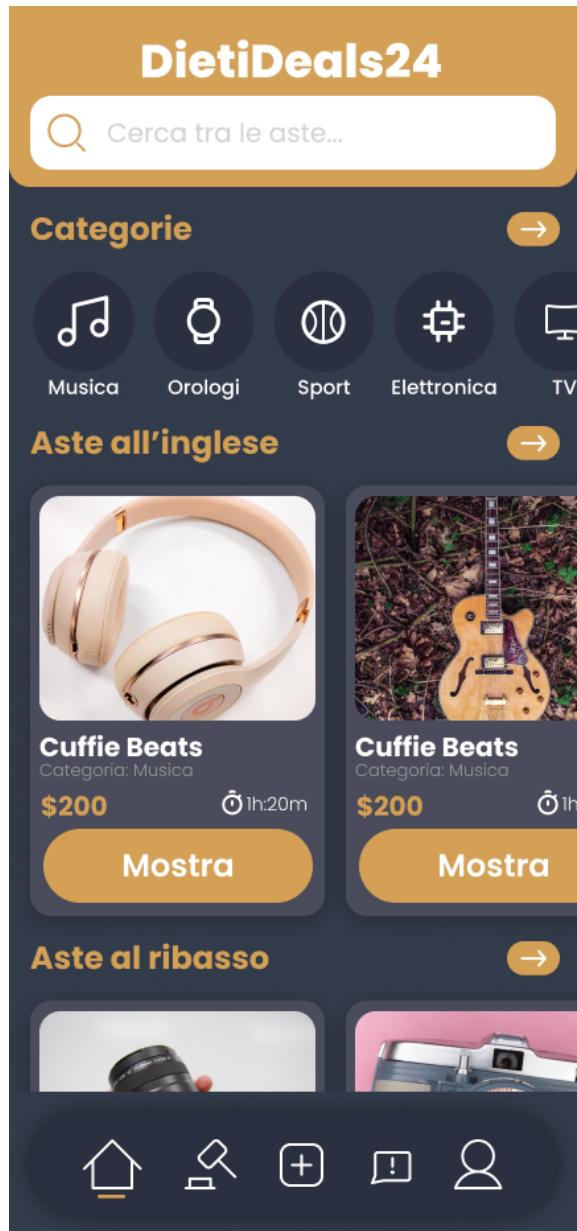


Figura 2.3: Mock-up03 Home

2.1.2.4 Mock-up04: Ricerca

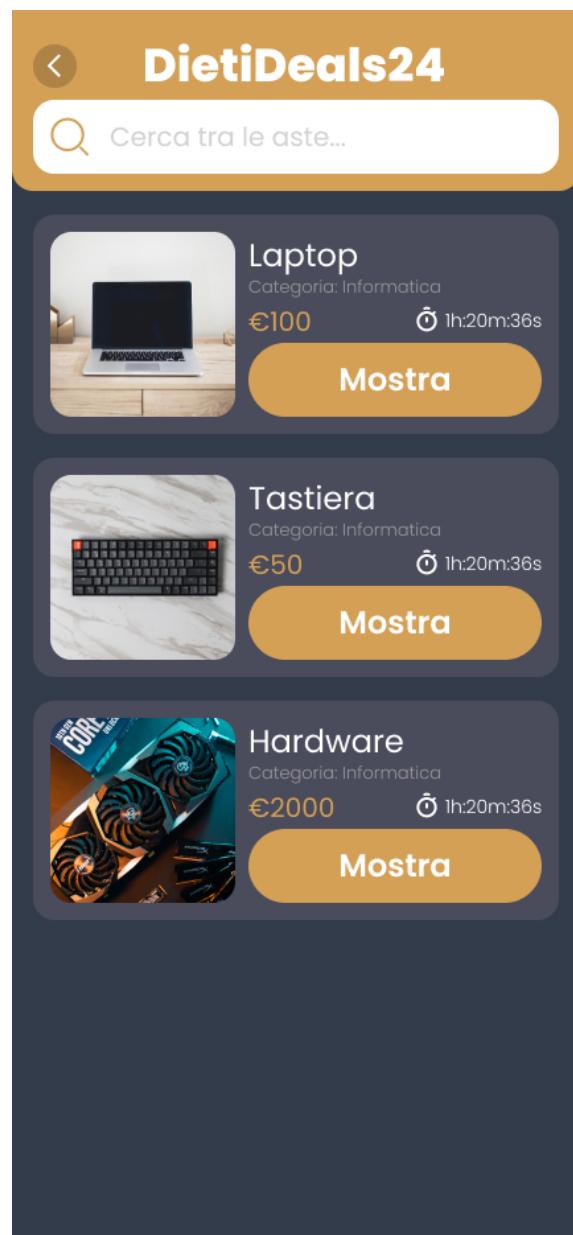


Figura 2.4: Mock-up04 Ricerca

2.1.2.5 Mock-up05: Ricerca per categoria

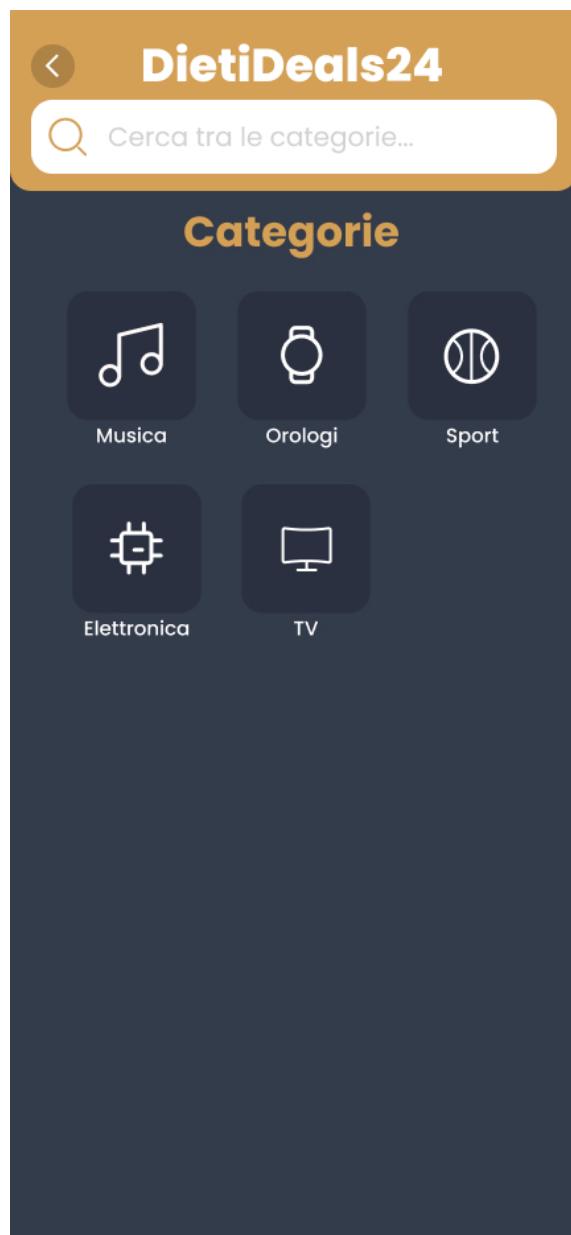


Figura 2.5: Mock-up05 Ricerca per categoria

2.1.2.6 Mock-up06: Dettagli asta

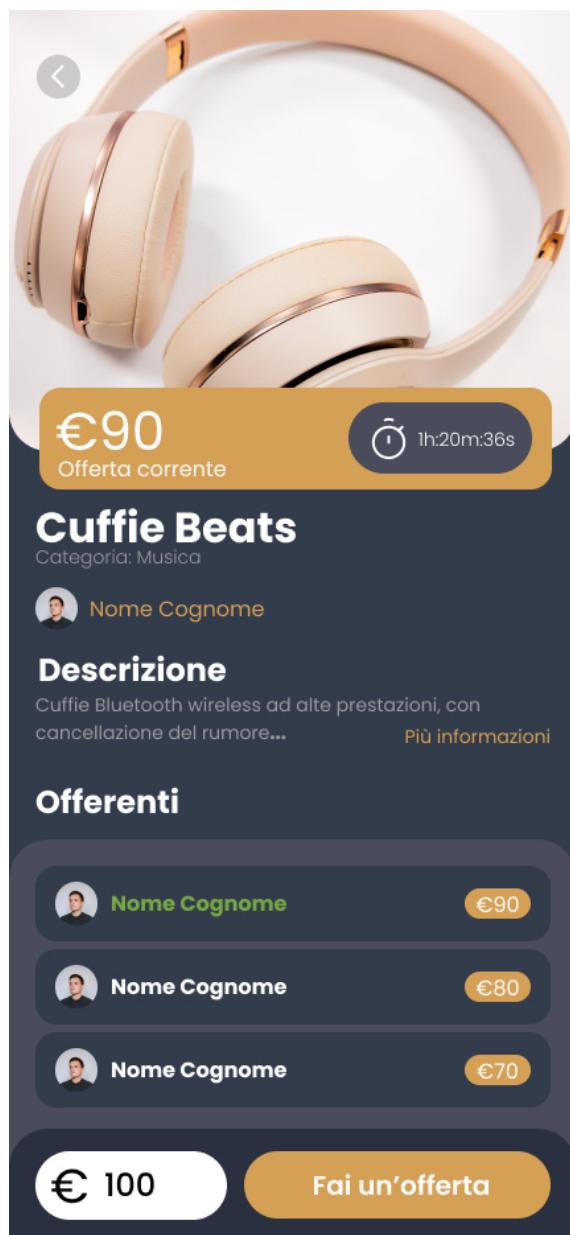


Figura 2.6: Mock-up06 Dettagli asta

2.1.2.7 Mock-up07: Profilo venditore



Figura 2.7: Mock-up07 Profilo venditore

2.1.2.8 Mock-up08: Popup conferma offerta



Figura 2.8: Mock-up08 Popup conferma offerta

2.1.2.9 Mock-up09: Popup scelta tipo di asta



Figura 2.9: Mock-up09 Popup scelta tipo di asta

2.1.2.10 Mock-up10: Creazione asta all'inglese



Figura 2.10: Mock-up10 Creazione asta all'inglese

2.1.2.11 Mock-up11: Creazione asta al ribasso



Figura 2.11: Mock-up11 Creazione asta al ribasso

2.1.2.12 Mock-up12: Notifiche

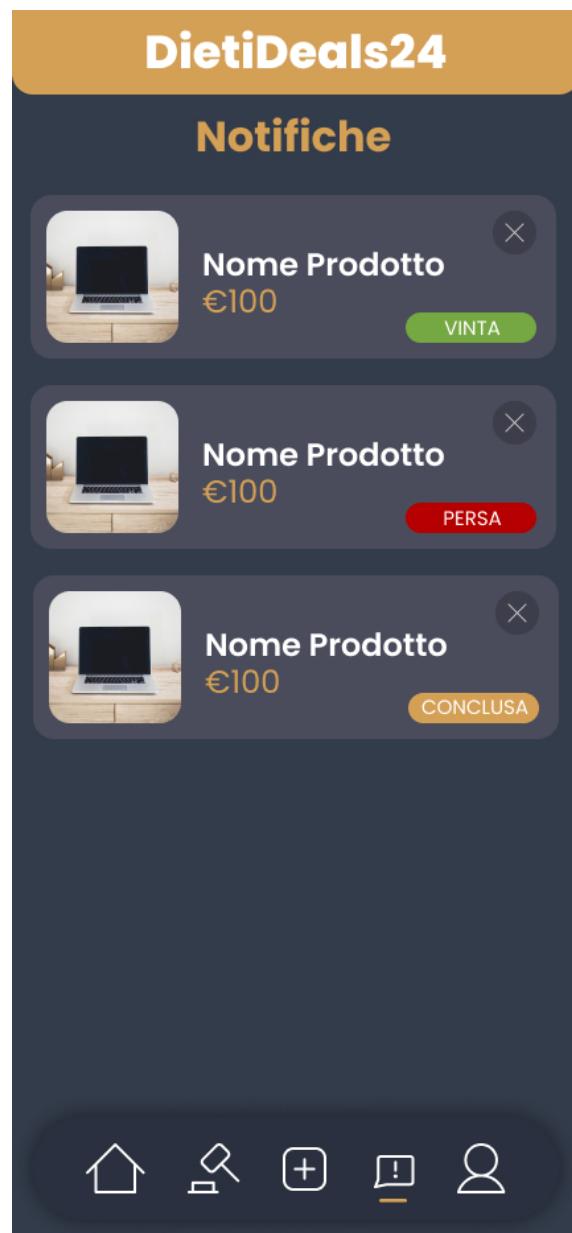


Figura 2.12: Mock-up12 Notifiche

2.1.2.13 Mock-up13: Popup conferma creazione asta

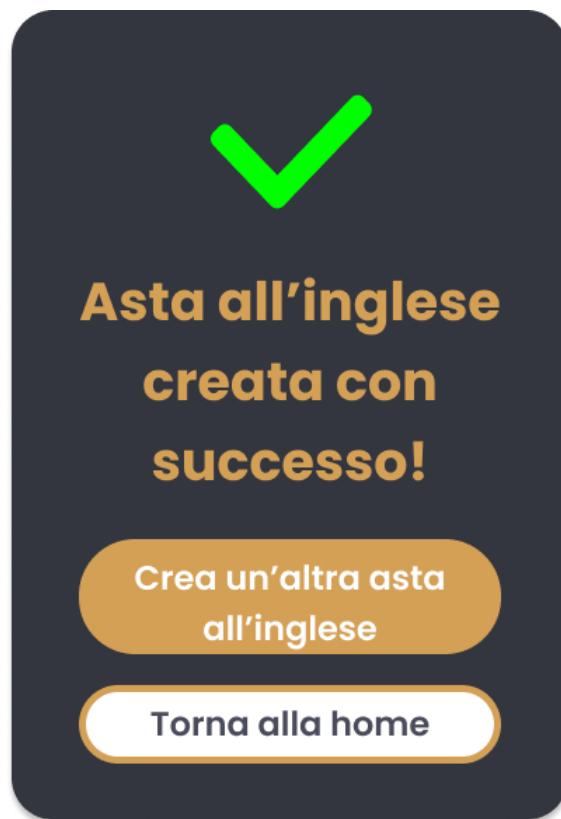


Figura 2.13: Mock-up13 Popup conferma creazione asta

2.1.3 Casi d'uso significativi

Di seguito una descrizione testuale dettagliata di 2 casi d'uso significativi: *Crea un'asta all'inglese* e *Presenta offerta per asta all'inglese*, con l'obiettivo di specificare in ogni aspetto l'interazione tra attore e sistema.

2.1.3.1 Creazione di un'asta all'inglese

Use Case #1	Crea di un'asta all'inglese		
Goal in Context	Un utente loggato crea un'asta all'inglese descrivendo l'oggetto che vuole mettere in vendita, specificando i parametri di vendita a suo piacimento		
Preconditions	L'utente ha effettuato correttamente il login		
Success End Conditions	L'asta all'inglese viene creata correttamente		
Actor	Utente loggato		
Trigger	Utente clicca su "+" nella barra di navigazione		
Description	Step n°	Utente loggato	Sistema
	1	Clicca sul bottone "+" nella barra di navigazione	
	2		Mostra Mockup-09
	3	Clicca su "Asta all'inglese"	
	4		Mostra Mockup-10
	5	Sceglie un'immagine	
	6	Inserisce il nome	
	7	Inserisce la categoria	
	8	Inserisce il prezzo iniziale	
	9	Inserisce il timer	
	10	Inserisce la soglia di rialzo	
	11	Inserisce la descrizione	
	12	Clicca su "Crea"	
	13		Mostra Mockup-13

	Step n°	Utente loggato	Sistema
Extension A: L'utente torna indietro	In qualsiasi step	Torna indietro	
	1.A		Mostra Mock-up03
Extension B: L'utente non inserisce un campo obbligatorio	Step n°	Utente loggato	Sistema
	Da 6 a 11	Clicca su "Crea"	
	1.B		Colora di rosso il campo vuoto
	2.B		Mostra il testo di suggerimento " <i>Questo campo è obbligatorio</i> "
Extension C: Internet non disponibile	Step n°	Utente loggato	Sistema
	Da 5	Clicca "Crea"	
	1.C		Mostra popup " <i>Internet non disponibile</i> "
Subvariation A: L'utente non sceglie un'immagine	Step n°	Utente loggato	Sistema
	12.A	Clicca su "Crea"	
	13.A		Mostra Mockup-13

2.1.3.2 Presenta offerta per asta all'inglese

Use Case #2	Presenta offerta per asta all'inglese		
Goal in Context	Un utente loggato fa un'offerta per accaparrarsi il prodotto messo all'asta		
Preconditions	L'utente ha effettuato correttamente il login		
Success End Conditions	Viene piazzata correttamente l'offerta		
Primary Actor	Utente loggato		
Trigger	Utente clicca sul pulsante "Fai un'offerta"		
Description	Step n°	Utente loggato	System
	1	Preme il pulsante "Mostra" di un'asta all'inglese su Mock-up03	
	2		Mostra Mock-up06 con tutte le informazioni dell'asta all'inglese
	3	Preme il pulsante "Fai un'offerta" del Mock-up06	
	4		Mostra Mock-up08
	5	Preme il pulsante "Conferma" del Mock-up08	
	6		Mostra Mock-up06 con le informazioni aggiornate sugli offerenti

Extension A: L'utente torna indietro	Step n°	Utente loggato	Sistema
	3.A	Torna indietro	
	4.A		Mostra Mock-up03
Extension B: L'utente annulla l'offerta	Step n°	Utente loggato	Sistema
	5.B	Preme il pulsante "Annulla" del Mock-up08	
	6.B		Mostra Mock-up06
Subvariation A: L'utente cerca un'asta prima di piazzare l'offerta	Step n°	Utente loggato	Sistema
	1.A	Clicca su barra di ricerca	
	2.A	Scrive parola chiave per ricerca	
	3.A		Mostra Mock-up04
	Riprende il main scenario da Step n° 1		
Subvariation B: L'utente filtra per categoria prima di piazzare l'offerta	Step n°	Utente loggato	Sistema
	1.B	Clicca su "->" della sezione Categorie	
	2.B		Mostra Mock-up05
	3.B	Clicca sulla categoria interessata	
	4.B		Mostra Mock-up04 con le aste della categoria selezionata
	Riprende il main scenario da Step n° 1		

2.1.4 Individuazione del target degli utenti

Dall'analisi dei requisiti funzionali, abbiamo identificato due target principali di utenti:

- I **venditori**, che vogliono espandere il loro bacino di utenza per dare più visibilità ai loro prodotti e ai propri profili social;
- Gli **acquirenti**, che sono alla ricerca di nuovi affari ed oggetti interessanti da vincere all'asta.

Oltre questi due target, abbiamo pensato che potesse esistere anche una parte di utenza che avremmo potuto attirare grazie ad un design estetico appagante. Dall'individuazione di questi possibili target, abbiamo definito le **Personas** riportate di seguito.

2.1.4.1 Personas 1: Sofia



"Cerca l'Unicità, Riempì la Vita di Piccole Meraviglie"

ANNI 28
PROFESSIONE Progettista grafica freelance
RESIDENZA Napoli

NOME
Sofia

BIOGRAFIA
Sofia è una progettista grafica appassionata di tecnologia e design. Cerca affari unici e prodotti di qualità attraverso aste online per arricchire la sua collezione di gadget tecnologici e articoli di design. Desidera espandere i suoi orizzonti attraverso altre app per avere la possibilità di aggiudicarsi nuovi oggetti. Vorrebbe un'alternativa alle app già presenti sul mercato che le consenta di ricercare e navigare facilmente tra le aste, per scovare nuovi prodotti interessanti.

OBIETTIVI

- Aggiudicarsi almeno un oggetto raro o di design attraverso un'asta
- Arricchire la sua collezione con pezzi unici e vivere esperienze di shopping emozionanti

PERSONALITÀ

INNOVATIVO:	★★★★★
AVVENTUROSO:	★★
ORGANIZZATO:	★★★★★
CREATIVO:	★★★★★
SOCEVOLE:	★★★★★

2.1.4.2 Personas 2: Luca



"Vivi all'Avanguardia,
Sperimenta il Mondo
con Stile"

ANNI	30
PROFESSIONE	Sviluppatore di software e app
RESIDENZA	Roma

NOME
Luca

BIOGRAFIA

Luca è uno sviluppatore di software appassionato di design moderno e tecnologia all'avanguardia. Ha una vasta esperienza nella creazione di app mobile e desktop con un focus particolare sul Material Design di Google. Utilizza la sua esperienza tecnica per creare applicazioni innovative e intuitive che soddisfano le esigenze dei clienti. Apprezza le app ben progettate che offrono un'esperienza utente fluida e piacevole. Contribuisce attivamente alla community di sviluppatori, condividendo conoscenze e collaborando a progetti open source.

OBIETTIVI

- Contribuire con feedback al miglioramento dell'esperienza utente
- Essere parte attiva nella comunità, aiutando a perfezionare l'app attraverso il suo interesse per il design moderno

PERSONALITÀ

INNOVATIVO:	★★★★★
AVVENTUROSO:	★★★★★
ORGANIZZATO:	★★★★★
CREATIVO:	★★★★★
SOCEVOLE:	★★★★★

2.1.4.3 Personas 3: Alessandro



"Vendi con Passione,
Guida il Futuro del
Business"

ANNI	35
PROFESSIONE	Imprenditore nel settore dell'e-commerce
RESIDENZA	Milano

NOME
Alessandro

BIOGRAFIA

Alessandro è un imprenditore esperto nel settore dell'e-commerce, specializzato in vendite di prodotti elettronici e informatici. Ha una presenza online consolidata e gestisce il suo negozio web. È appassionato di nuove tecnologie e desidera espandere il suo business partecipando a aste online. Cerca modi per poter espandere il suo bacino di utenza, non rinunciando alla visibilità del suo negozio web. Apprezzerebbe la possibilità di inserire link ai suoi riferimenti già esistenti.

OBIETTIVI

- Aumentare il volume di vendite
- Sfruttare la piattaforma per raggiungere nuovi acquirenti e ampliare la visibilità del suo negozio

PERSONALITÀ

INNOVATIVO:	★★★★★
AVVENTUROSO:	★★★★★
ORGANIZZATO:	★★★★★
CREATIVO:	★★★★★
SOCEVOLE:	★★★★★

2.1.5 Valutazione dell'usabilità a priori

*L'usabilità di un prodotto è il grado con cui esso può essere usato da specificati utenti per raggiungere specificati obiettivi con efficacia, efficienza e soddisfazione in uno specificato contesto d'uso.*¹

- L'**efficacia** viene definita come la accuratezza e completezza con cui gli utenti raggiungono specificati obiettivi. Essa considera pertanto il “livello di precisione” con cui l’utente riesce a raggiungere i suoi scopi, misurato in qualche modo numericamente.
- L'**efficienza** è definita come “la quantità di risorse spese in relazione all’accuracy e alla completezza con cui gli utenti raggiungono gli obiettivi”. Tali risorse potranno essere di natura differente secondo le situazioni, e potranno anch’esse essere quantificate. Per esempio: il tempo impiegato per ottenere un determinato risultato, il numero di tasti da premere per realizzare una certa funzione, il numero di operazioni di un certo tipo da effettuare, ecc.
- La **soddisfazione**, infine, è definita – in modo in effetti un po’ contorto - come “la libertà dal disagio e l’attitudine positiva verso l’uso del prodotto”.

Il percorso scelto in fase di progettazione è stato quello dell’ibridazione: abbiamo considerato prodotti già esistenti sul mercato che avessero funzionalità simili a quelle identificate nella fase di analisi dei requisiti (**Vinted**, **Wallapop**, **Ebay**) e abbiamo fuso quelle che ritenevamo essere gli aspetti di design migliori da ciascuno dei prodotti. L’analisi dei prodotti presenti sul mercato ha influenzato anche la scelta della paletta colori, in quanto abbiamo cercato di usare un colore dominante non utilizzato dalle app più famose. Analizzando la **ruota di Plutchik**, abbiamo scelto come colore predominante il giallo, facendo riferimento al significato di "interesse", con l’idea di generare interesse nell’utente nei confronti dei prodotti presenti nell’app.

Per la valutazione dell’usabilità a priori ci rifacciamo alle **euristiche di Jakob Nielsen** e alle **regole d’oro di Ben Shneiderman**. In particolare:

- **Coerenza a tutti i costi:** l’interfaccia è stata pensata con una specifica paletta colori coerente per ogni schermata, con l’aggiunta dei colori rosso e verde, universalmente riconosciuti per indicare rispettivamente errori e successi.
- **Offrire riscontri informativi:** ogni interfaccia ha un’etichetta in cima alla pagina che permette all’utente di capire rapidamente dove è stato portato dall’azione che ha effettuato.
- **Dialogo con gli utilizzatori:** sequenze di azioni come la creazione di un’asta o il piazzamento di un’offerta sono delimitate da popup di conferma di volere iniziare quella determinata azione e popup di conclusione che descrivono lo stato della terminazione.

¹“the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (ISO 9241-11:1998).

- **Aiutare gli utenti a riconoscere gli errori, diagnosticarli e correggerli:** in caso di inserimento erroneo in determinati campi, verrà segnalato in rosso il campo non inserito correttamente e verranno fornite istruzioni sul corretto inserimento richiesto.
- **Ridurre il carico della memoria a breve termine:** tutti i form sono adattati ad essere visualizzati su una singola schermata.
- **Libertà e controllo da parte degli utenti:** su ogni interfaccia (escludendo quelle raggiungibili dalla barra di navigazione) è presente un tasto esplicito per tornare alla schermata precedente.
- **Design minimalista ed estetico:** il design è basato sul Material Design, con l'obiettivo di essere esteticamente pulito e intuitivo, puntando a qualsiasi tipo di utente.
- **Consistenza e standard:** sono state utilizzate icone autoesplicative e largamente utilizzate per specifiche funzioni.

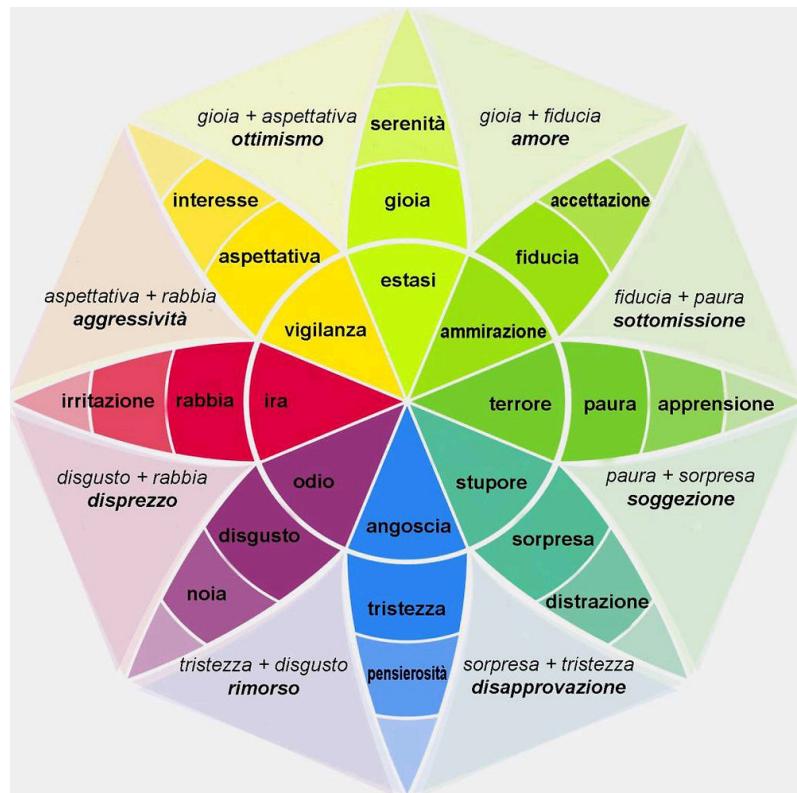


Figura 2.14: Ruota di Plutchik

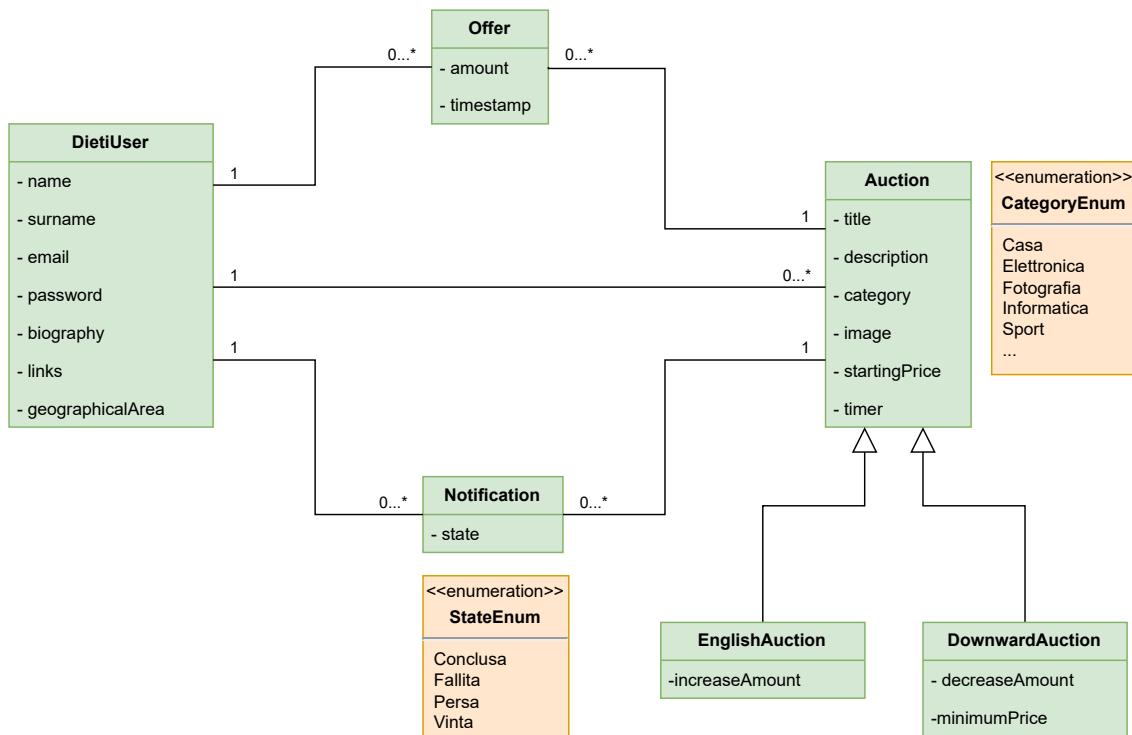
2.1.6 Glossario

Termino	Descrizione
Mock-up	Rappresentazione visiva di un'interfaccia utente senza funzionalità, utile per la progettazione e la valutazione preliminare
Use Case Diagram	Diagramma che illustra interazioni tra utenti e sistema, enfatizzando gli scenari d'uso principali
Tabelle di Cockburn	Strumento per descrivere casi d'uso dettagliando azioni attore-sistema, sviluppato da Alistair Cockburn
Springboot	Framework per lo sviluppo rapido di applicazioni Java, basato su Spring e configurazione automatica
Framework	Insieme strutturato di librerie, moduli e linee guida che forniscono un'architettura di base per lo sviluppo di software, facilitando la creazione di applicazioni attraverso l'uso di codice riutilizzabile e pratiche di programmazione consolidate
JWT	Token JSON Web per autenticazione e trasmissione sicura di informazioni tra client e server
JPA	API Java per la persistenza dei dati in database relazionali, semplificando l'interazione con il database attraverso oggetti Java e mapping degli oggetti alle tabelle del database (ORM)
ORM	Object-Relational Mapping, tecnica di programmazione per mappare oggetti di un'applicazione ai record di un database relazionale, semplificando l'interazione tra il codice dell'applicazione e il database
PostgreSQL	Sistema di gestione di database relazionale open-source, ampiamente utilizzato per l'affidabilità e le funzionalità avanzate
Database	Un sistema per la memorizzazione e la gestione dei dati in modo organizzato, che consente l'accesso e la manipolazione efficiente delle informazioni da parte di un'applicazione.
Android	Sistema operativo mobile sviluppato da Google per dispositivi touchscreen
Retrofit	Libreria per le chiamate di rete in applicazioni Android, che semplifica l'interazione con API REST
Architettura a tre livelli	Suddivisione di un'applicazione in presentazione, logica di business e dati per facilitare la manutenzione e la scalabilità

Testing	Attività per verificare la correttezza e la qualità del software attraverso prove sistematiche e controllate
JUnit	Framework per il testing unitario in Java, offre strumenti per automatizzare i test e verificarne l'integrità
Black Box	Approccio al testing che si concentra sul comportamento esterno del software, senza conoscere la struttura interna
White Box	Approccio al testing che esamina la struttura interna del software, inclusi codice e logica di implementazione
Docker	Piattaforma per lo sviluppo, il deployment e l'esecuzione di applicazioni in contenitori, garantendo portabilità e isolamento
Docker Compose	Strumento che semplifica la gestione di applicazioni Docker multi-container, consentendo la definizione e l'avvio di ambienti complessi con un singolo file di configurazione.
Container	Un'unità di software che contiene tutti gli elementi necessari per eseguire un'applicazione, inclusi codice, runtime, librerie e dipendenze, isolati dagli altri processi del sistema operativo per garantire la portabilità e la coerenza dell'ambiente di esecuzione.
API REST	Protocollo standard per la comunicazione e lo scambio di dati tra software, permettendo l'interazione e l'accesso alle funzionalità di un'applicazione da parte di altre applicazioni o componenti.
API REST	Interfaccia per l'accesso a risorse tramite HTTP, seguendo i principi REST per la scalabilità e l'interoperabilità
Client	Il componente che invia richieste a un server per ottenere informazioni o servizi, come ad esempio un browser web che richiede pagine da un server web.
Server	Il componente che fornisce risorse, servizi o informazioni ai client in risposta alle loro richieste, come ad esempio un server web che invia pagine web ai browser dei client.
Backend	La parte di un'applicazione che gestisce le funzioni di elaborazione e accesso ai dati, solitamente non visibile agli utenti finali, ma che fornisce funzionalità e dati al frontend
HTTP	Protocollo di comunicazione utilizzato per il trasferimento di dati sul web, basato sul modello richiesta-risposta
AWS	Amazon Web Services, piattaforma di servizi cloud offerti da Amazon per storage, calcolo, networking e altro
EC2	Servizio di hosting di istanze di server virtuali su AWS, permettendo la scalabilità e la gestione flessibile delle risorse

2.2 Specifica dei requisiti

2.2.1 Entità



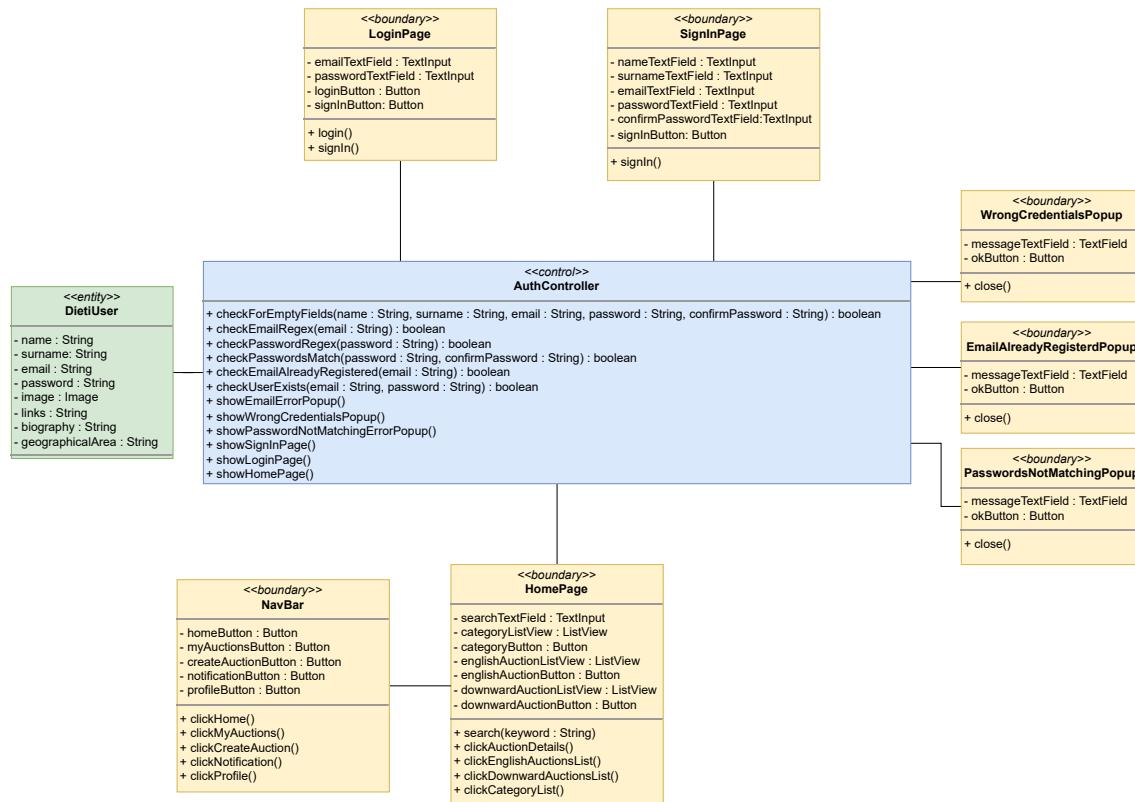
2.2.2 Class Diagram di Analisi

Qui riportiamo i Class Diagram di Analisi di DietiDeals24.

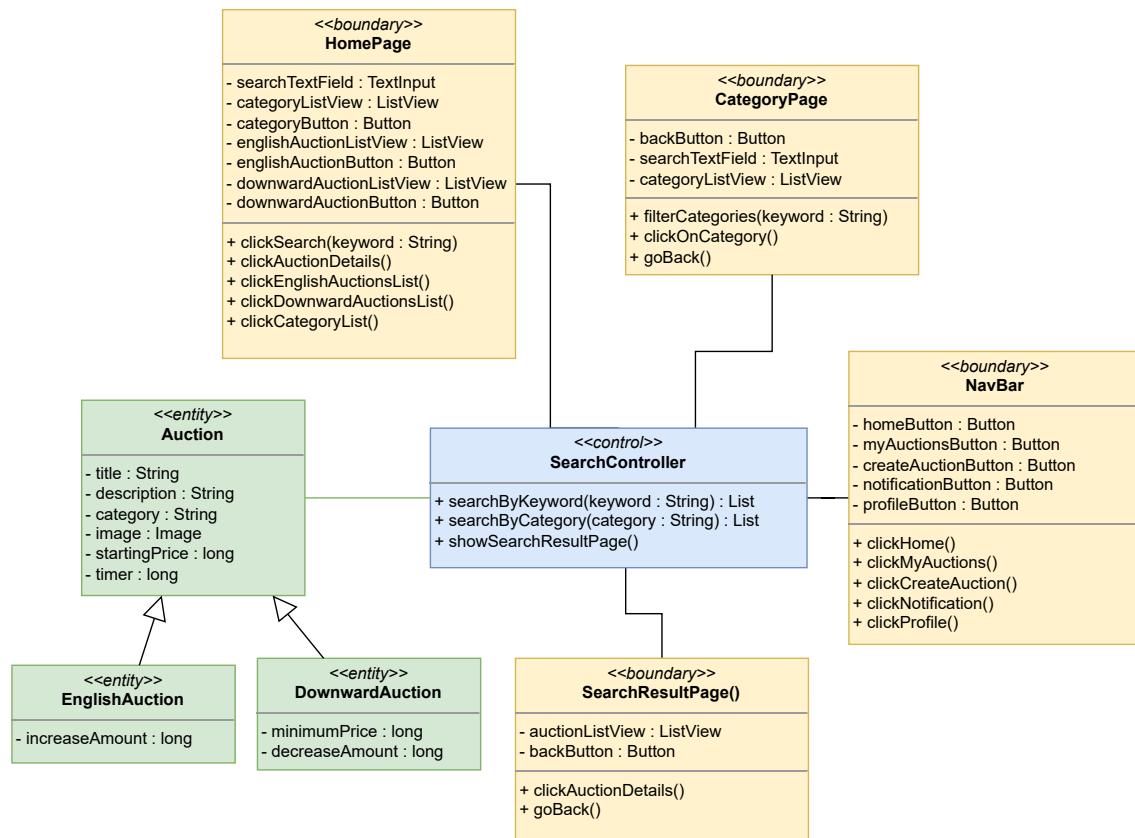
2.2.3 Legenda dei colori



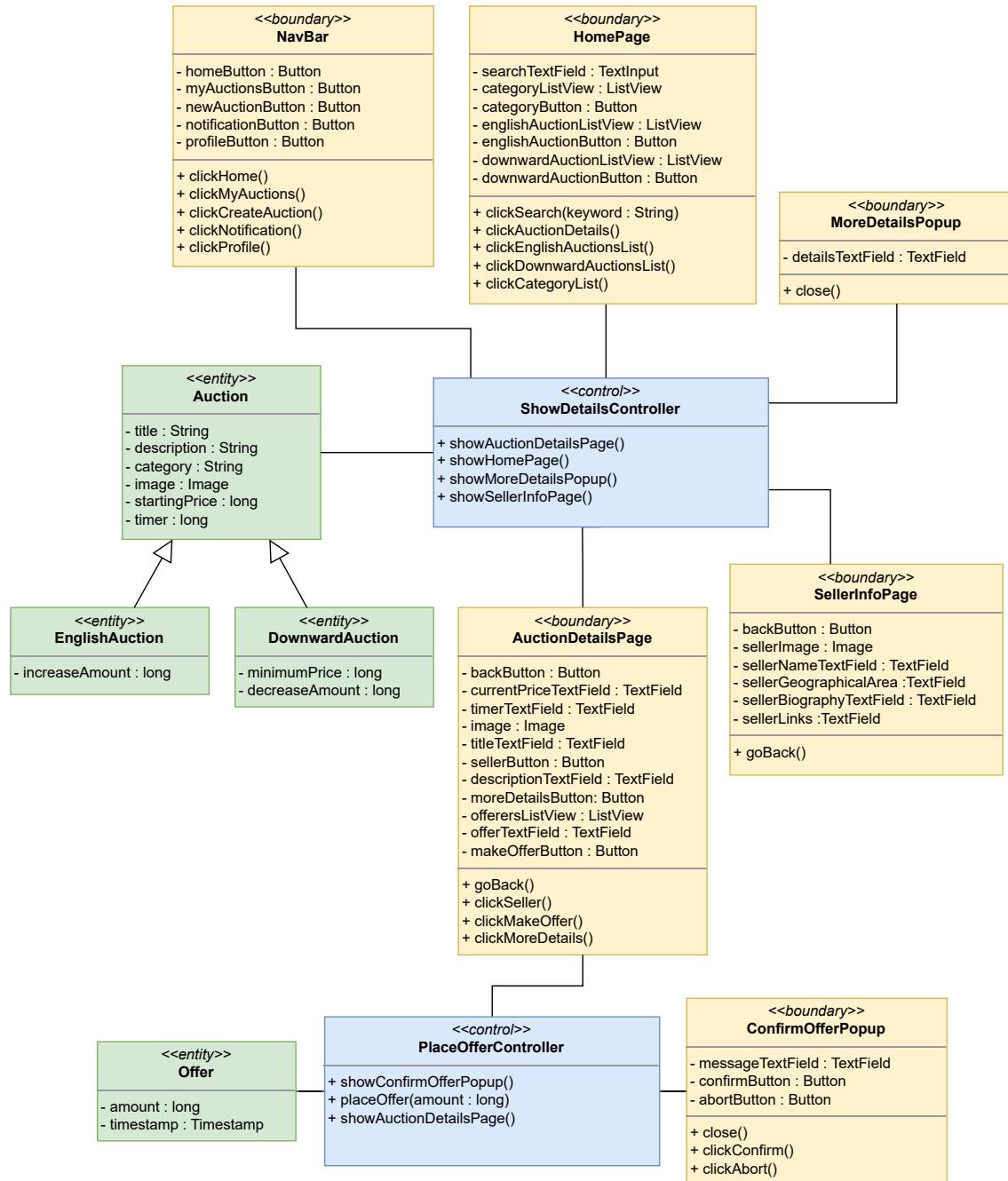
2.2.3.1 Login/Registrazione



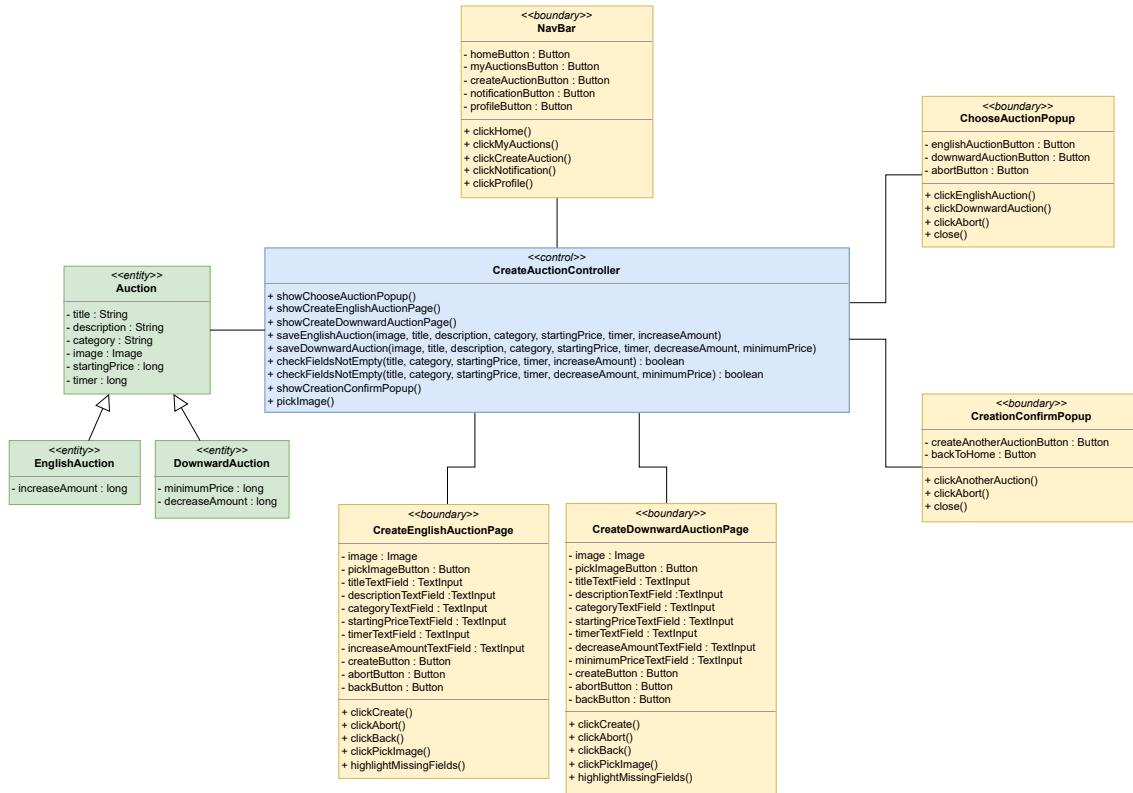
2.2.3.2 Ricerca



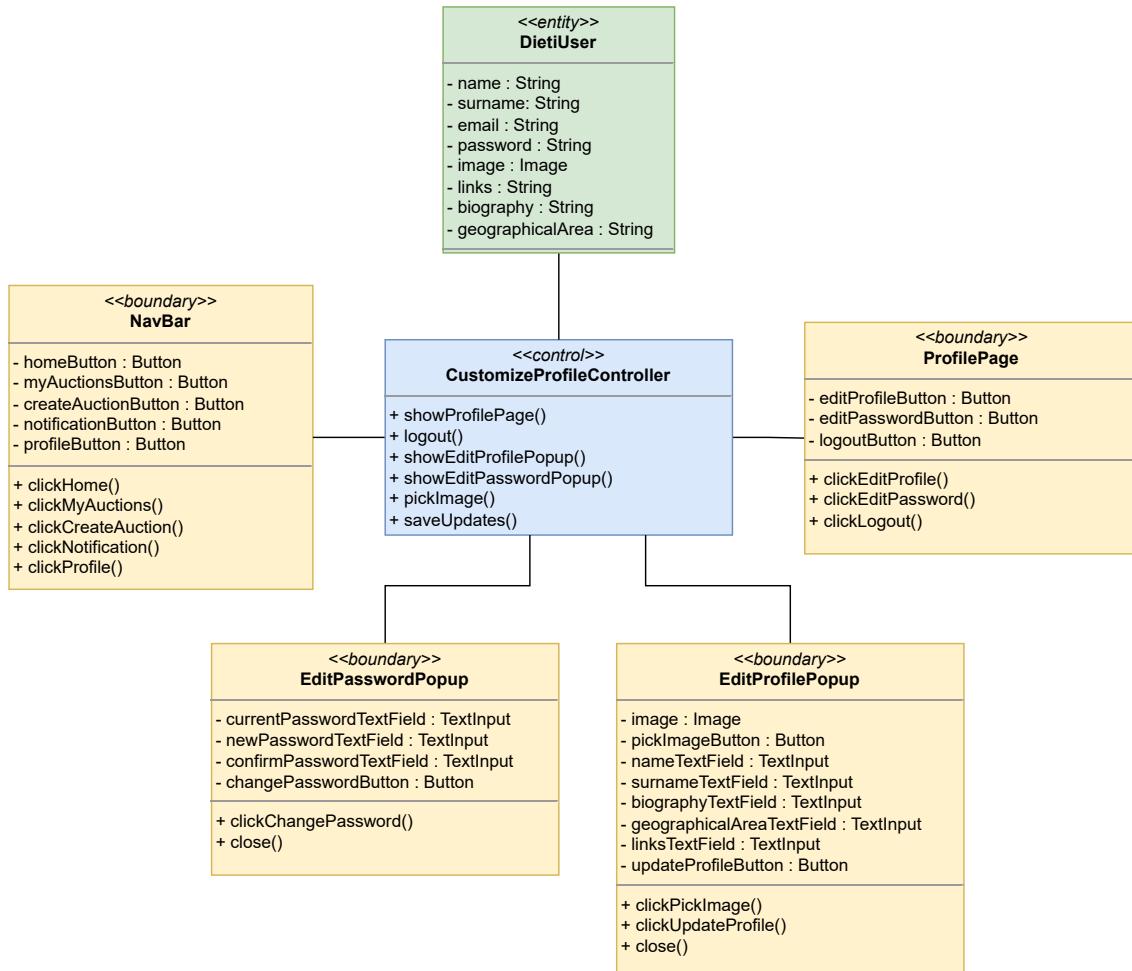
2.2.3.3 Partecipazione ad un'asta



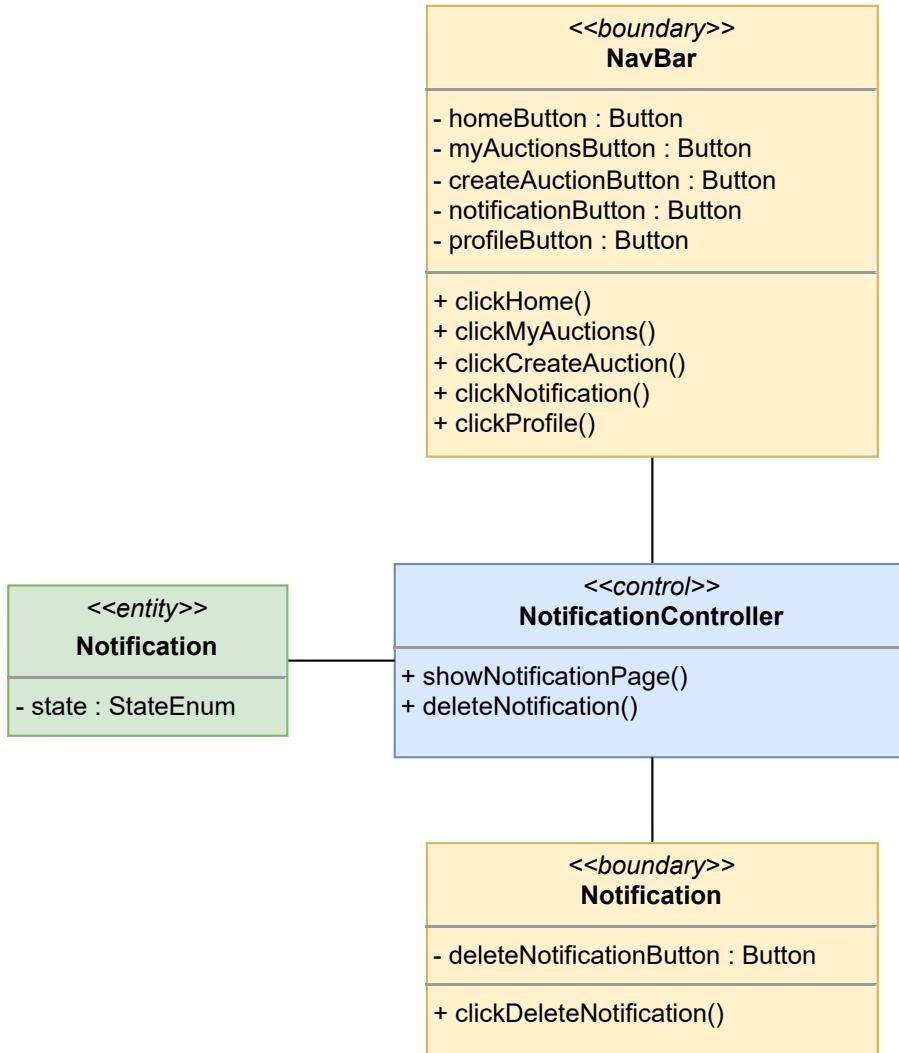
2.2.3.4 Creazione asta



2.2.3.5 Personalizzazione profilo utente



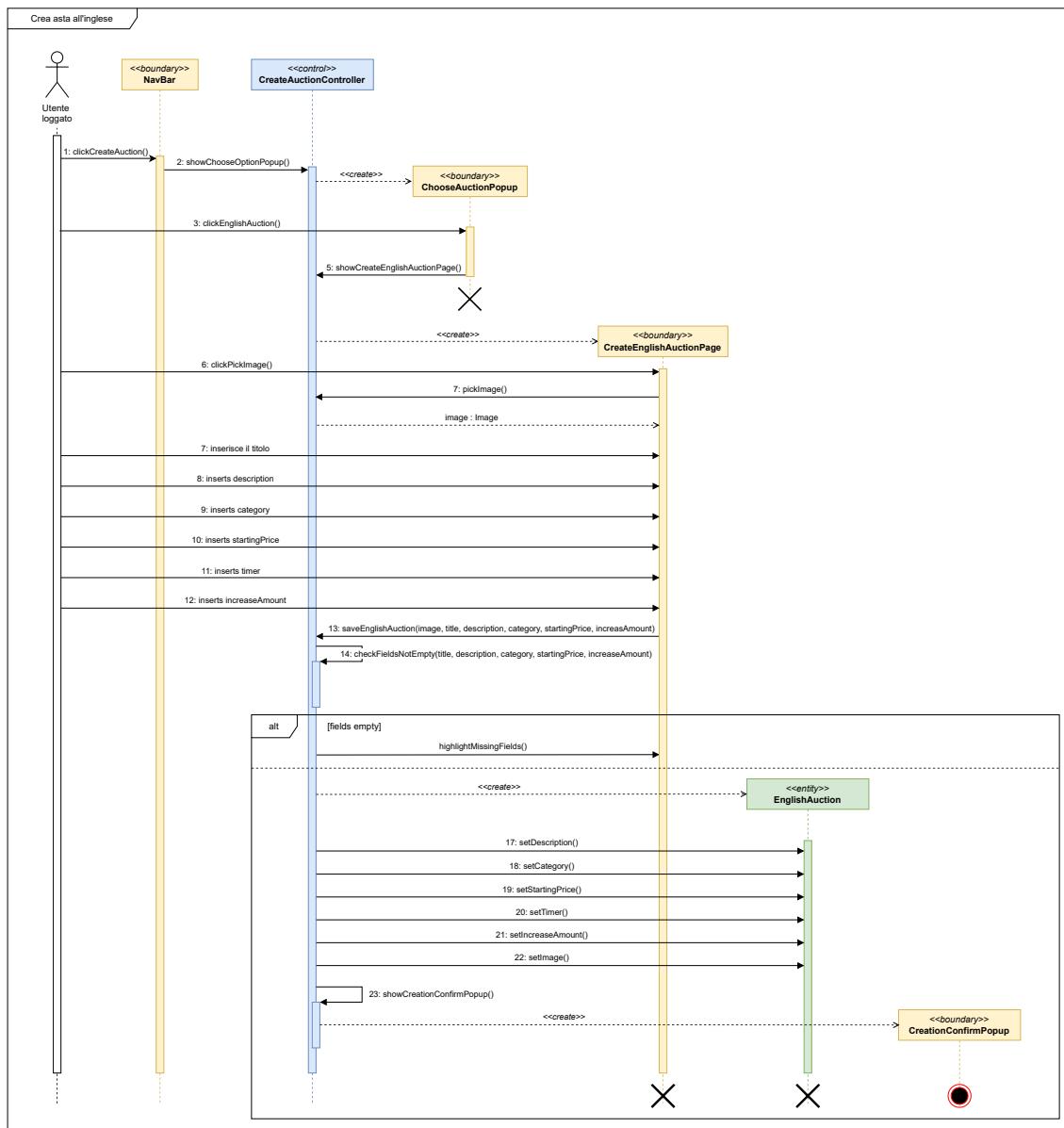
2.2.3.6 Notifica



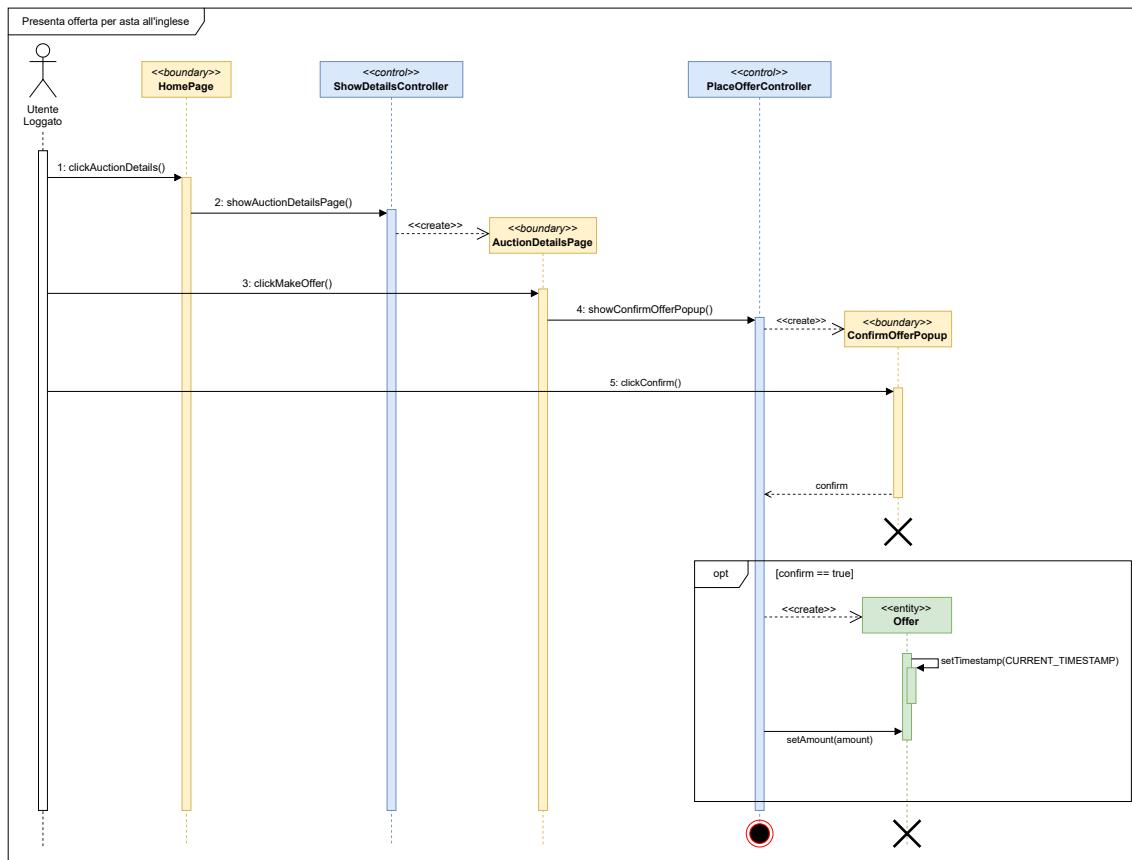
2.2.4 Sequence Diagram

Qui riportiamo i Sequence Diagram di Analisi per le funzionalità "*Crea asta all'inglese*" e "*Presenta offerta per asta all'inglese*".

2.2.4.1 Crea asta all'inglese



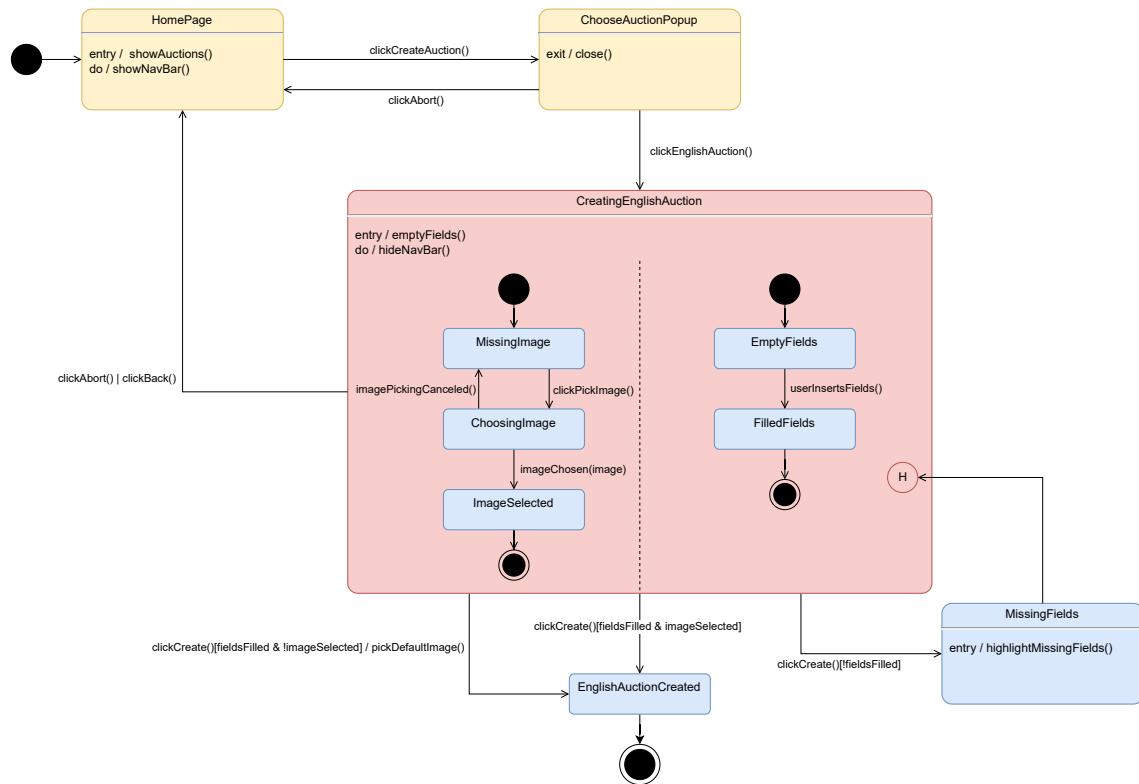
2.2.4.2 Presenta offerta per asta all'inglese



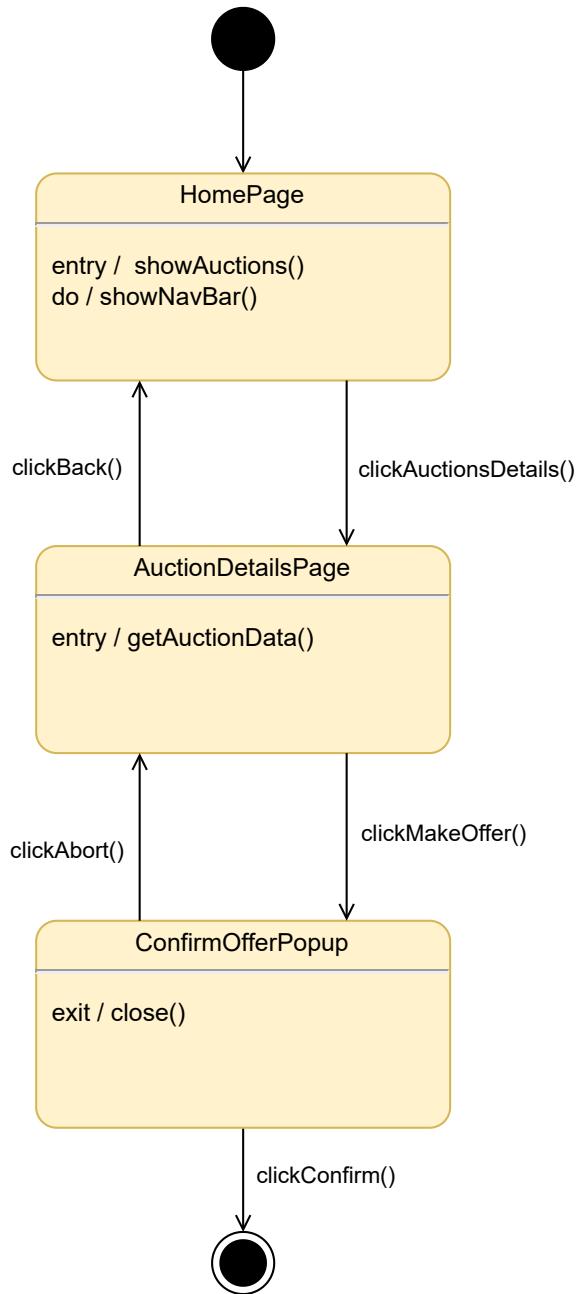
2.2.5 Statechart Diagram

Qui riportiamo gli Statechart Diagram di Analisi per le funzionalità "*Crea asta all'inglese*" e "*Presenta offerta per asta all'inglese*".

2.2.5.1 Crea asta all'inglese



2.2.5.2 Presenta offerta per asta all'inglese

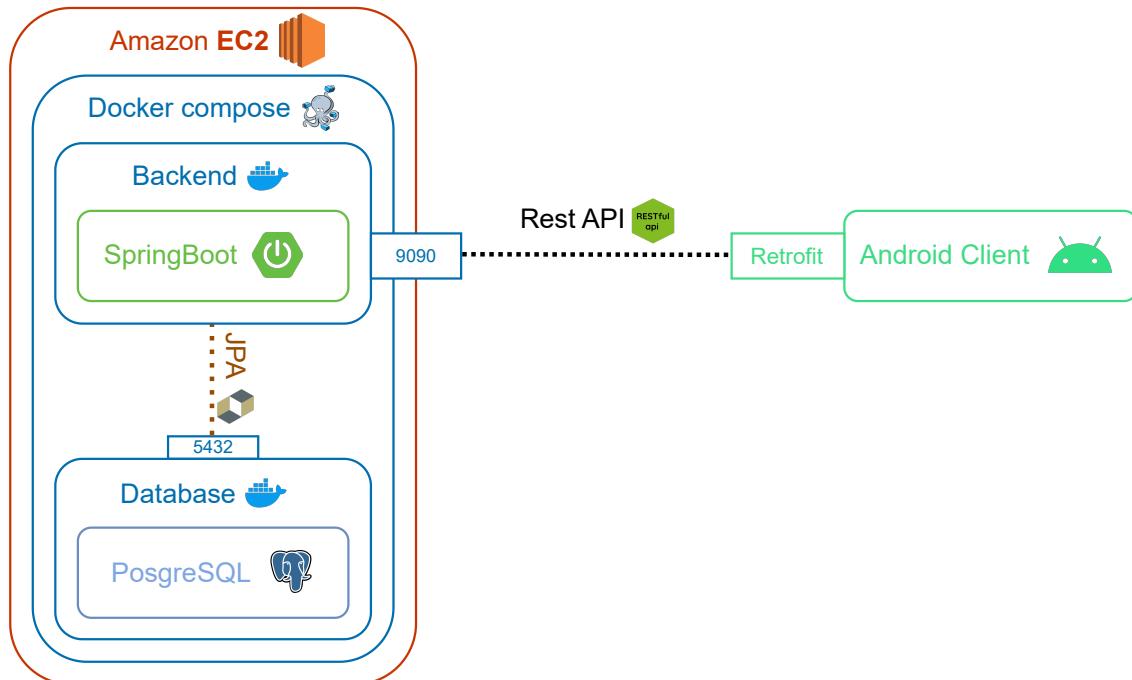


Capitolo 3

Design del Sistema

3.1 Analisi Architetturale

Per lo sviluppo di DietiDeals24 abbiamo adottato una Architettura a Tre Livelli .



Dal basso verso l'alto, abbiamo:

- Il Database, costituito da un Container Docker contenente un'immagine di PostgreSQL;
- Il Backend, costituito da un Container Docker contenente l'applicativo Java sviluppato utilizzando il framework SpringBoot.
- Il Client, costituito da una applicazione Android.

Il Backend attinge al Database mediante JPA Hibernate, interrogando la porta 5432. Il Client comunica con il Backend mediante API REST; le richieste HTTP vengono

inviate tramite Retrofit alla porta 9090. Il deployment del Backend e del Database avviene mediante Docker Compose su Amazon EC2.

3.1.1 Database

Per il Database abbiamo deciso di utilizzare un DBMS relazionale, ossia PostgreSQL. Abbiamo ritenuto che un DBMS relazionale fosse la tipologia più appropriata per l'applicativo richiesto.

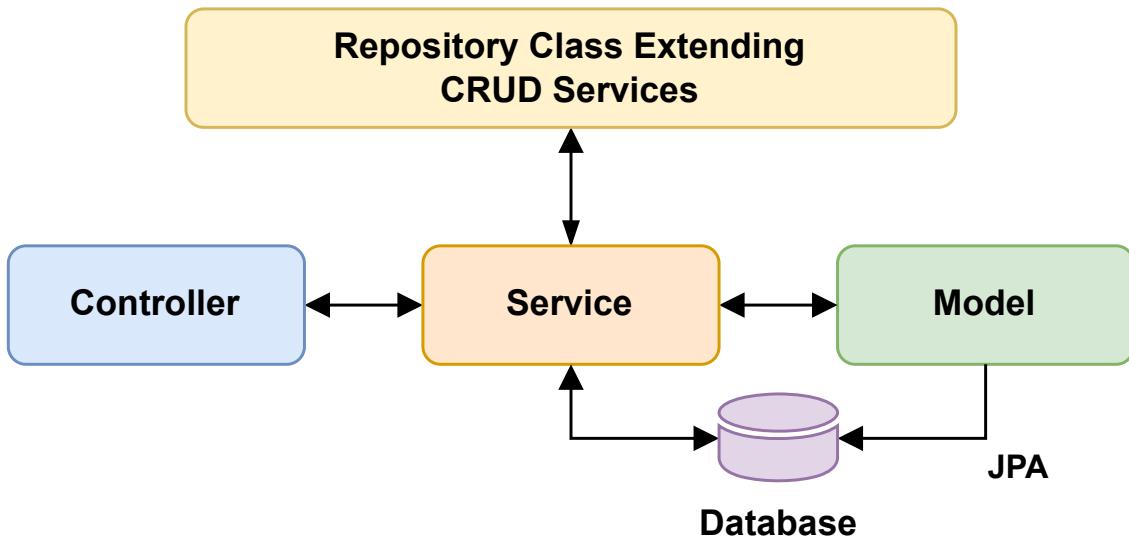
3.1.2 JPA Hibernate

JPA Hibernate è stato utilizzato per semplificare l'esecuzione di query dal Backend al Database. Esso permette di creare rapidamente un Database mediante annotazioni sulle classi che compongono il model e semplifica la creazione di query utilizzando semplicemente metodi denominati con keyword nel giusto ordine.

3.1.3 SpringBoot e Retrofit

Abbiamo scelto di utilizzare SpringBoot perché è un framework che supporta lo sviluppo di applicazioni web RESTful e che, mediante l'uso di annotazioni, consente uno sviluppo rapido e ordinato.

Per formare le richieste HTTP da inviare al nostro Backend abbiamo usato Retrofit. Retrofit è un client HTTP per Android e Java che, tramite l'uso di interfacce e annotazioni, consente di creare richieste RESTful che si integrano facilmente con SpringBoot.



3.1.4 Docker

Docker è uno strumento open-source per la creazione, personalizzazione ed esecuzione di Container. Con esso abbiamo la garanzia che la nostra applicazione sia lanciabile nel giro di pochi secondi e che sia eseguibile su qualsiasi hardware capace

di eseguire Container Docker. Questo ha reso, insieme all'uso di Docker Compose, facile e immediato il rilascio dell'applicazione in cloud mediante Amazon EC2.

3.1.5 Docker Compose

Docker Compose è uno strumento che permette di gestire in un unico file in formato YAML più Container Docker insieme. In particolare, lo abbiamo usato per il deployment automatico del Backend e del Database.

3.1.6 Amazon EC2

L'applicazione è stata rilasciata su Amazon EC2, che è l'istanza Cloud più flessibile e che ci forniva tutte le funzionalità di cui necessitavamo, ossia un'ambiente Linux su cui lanciare i nostri Container Docker e il bilanciamento automatico delle prestazioni in base al carico di lavoro.

3.1.7 API REST

REST (REpresentational State Transfer) è uno stile architettonico basato su un insieme di *principi di progettazione* (ne sono 6) e *linee guida* per sistemi che utilizzano Internet come mezzo di comunicazione. L'architettura REST si basa su HTTP e sul concetto di risorsa, ognuna delle quali è identificata da un URI. Le risorse sono rappresentate in formato testuale (poiché HTTP è di tipo testuale), spesso utilizzando i formati come JSON (JavaScript Object Notation) o XML (eXtensible Markup Language).

La manipolazione delle risorse avviene tramite i metodi HTTP:

- **POST** (Create): creare una nuova risorsa.
- **GET** (Read/Retrive): leggere o recuperare i dati di una risorsa.
- **PUT** (Update): aggiornare o modificare una risorsa esistente.
- **DELETE** (Delete): eliminare una risorsa.

Una **API REST** è un'API conforme ai principi e alle linee guida di REST. Di seguito verranno riportate le API REST sviluppate per l'applicazione DietiDeals24.

3.1.7.1 DietiUserAuth

Verbo	URI HTTP	Protetta?	Cosa fa?
POST	/auth/register	No	Permette ad un utente di registrarsi
POST	/auth/login	No	Permette ad un utente di effettuare l'accesso

Tabella 3.1: API REST per l'autenticazione

3.1.7.2 DietiUser

Verbo	URI HTTP	Protetta?	Cosa fa?
GET	/users/{id}	Si	Permette di recuperare un utente in base al suo id
GET	/users/email/{id}	Si	Permette di recuperare un utente in base alla sua email
POST	/users/{id}/profile-picture	Si	Permette di caricare l'immagine del profilo dell'utente
POST	/users/{id}	Si	Permette di aggiornare i dati dell'utente
POST	/users/{id}/password	Si	Permette di aggiornare la password dell'utente

Tabella 3.2: API REST per il DietiUser

3.1.7.3 Notification

Verbo	URI HTTP	Protetta?	Cosa fa?
GET	/notifications/receiver/{id}	Si	Permette di recuperare tutte le notifiche di un utente in base al suo id
GET	/notifications/receiver/{id}/push	Si	Permette di recuperare tutte le notifiche push non ancora inviate ad un utente in base al suo id
DELETE	/notifications/{id}	Si	Elimina una notifica in base al suo id

Tabella 3.3: API REST per le notifiche

3.1.7.4 EnglishAuction

Verbo	URI HTTP	Protetta?	Cosa fa?
POST	/english-auctions/create	Si	Permette di creare un'asta all'inglese
POST	/english-auctions/{id}/image	Si	Permette di caricare un'immagine per un'asta all'inglese in base al suo id
GET	/english-auctions	Si	Permette di recuperare tutte le aste all'inglese
GET	/english-auctions/first-six	Si	Permette di recuperare le prime 6 aste all'inglese
GET	/english-auctions/category/{category}	Si	Permette di recuperare le aste all'inglese in base ad una categoria
GET	/english-auctions/{id}	Si	Permette di recuperare un'asta all'inglese in base al suo id
GET	/english-auctions/search/{} /english-auctions/search/{keyword}	Si	Permette di recuperare delle aste all'inglese in base al nome e/o descrizione
GET	/english-auctions/owner/{id}	Si	Permette di recuperare tutte le aste all'inglese di un utente in base al suo id

Tabella 3.4: API REST per le aste all'inglese

3.1.7.5 DownwardAuction

Verbo	URI HTTP	Protetta?	Cosa fa?
POST	/downward-auctions/create	Si	Permette di creare un'asta al ribasso
POST	/downward-auctions/{id}/image	Si	Permette di caricare un'immagine per un'asta al ribasso in base al suo id
GET	/downward-auctions	Si	Permette di recuperare tutte le aste al ribasso
GET	/downward-auctions/first-six	Si	Permette di recuperare le prime 6 aste al ribasso
GET	/downward-auctions/category/{category}	Si	Permette di recuperare le aste al ribasso in base ad una categoria
GET	/downward-auctions/{id}	Si	Permette di recuperare un'asta al ribasso in base al suo id
GET	/downward-auctions/search/{keyword}	Si	Permette di recuperare delle aste al ribasso in base al nome e/o descrizione
GET	/downward-auctions/owner/{id}	Si	Permette di recuperare tutte le aste al ribasso di un utente in base al suo id

Tabella 3.5: API REST per le aste al ribasso

3.1.7.6 Offer

Verbo	URI HTTP	Protetta?	Cosa fa?
GET	/offers/english/{id}	Si	Permette di recuperare tutti gli offerenti di un'asta all'inglese in base al suo id
GET	/offers/english/offerer/{id}	Si	Permette di recuperare tutte le aste a cui un utente ha fatto un'offerta in base al suo id
POST	/offers/english	Si	Permette di fare un'offerta ad un'asta all'inglese
POST	/offers/downward	Si	Permette di acquistare un oggetto di un'asta al ribasso

Tabella 3.6: API REST per gli offerenti

3.1.7.7 Image

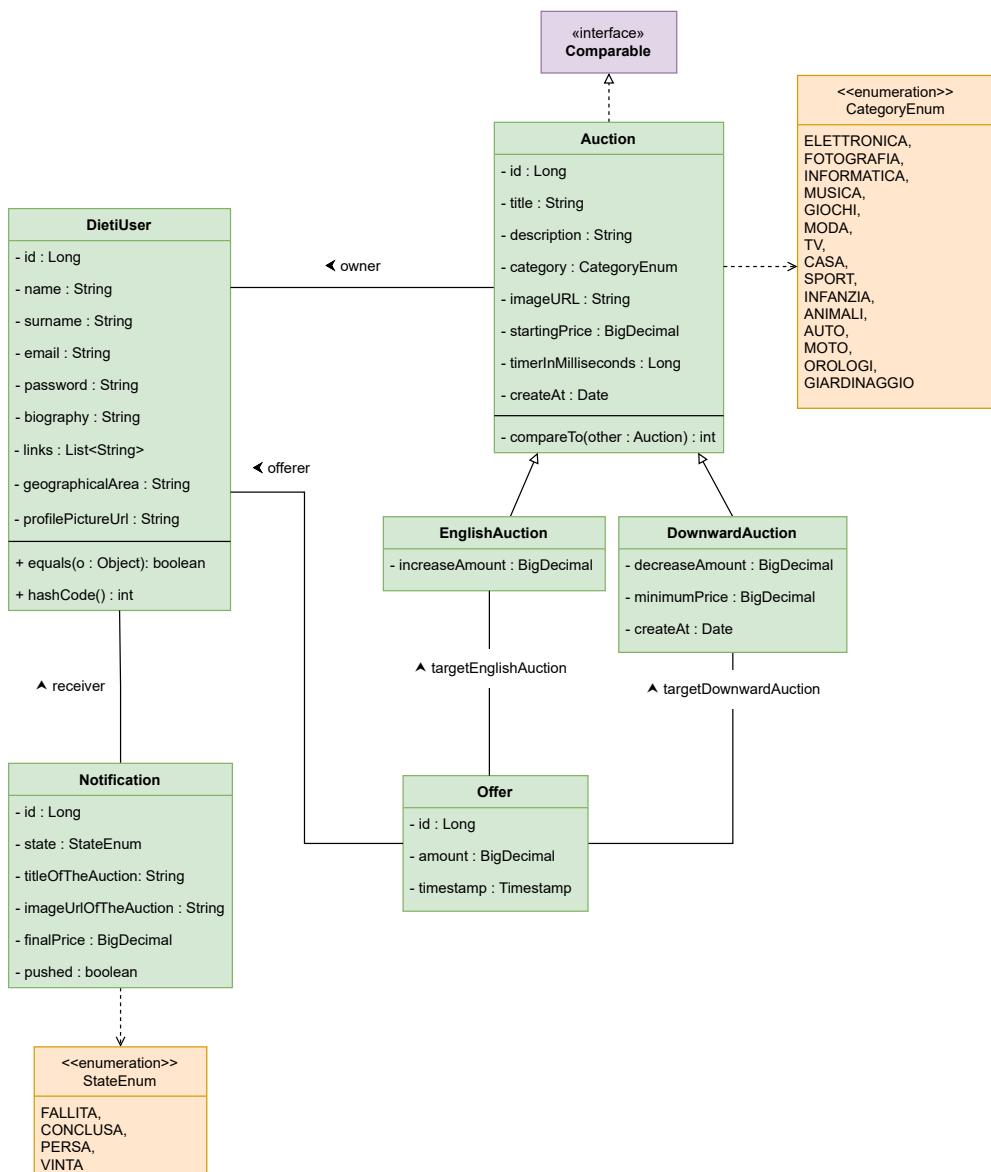
Verbo	URI HTTP	Protetta?	Cosa fa?
GET	/images	Si	Permette di recuperare un'immagine in base al suo url

Tabella 3.7: API REST per le immagini

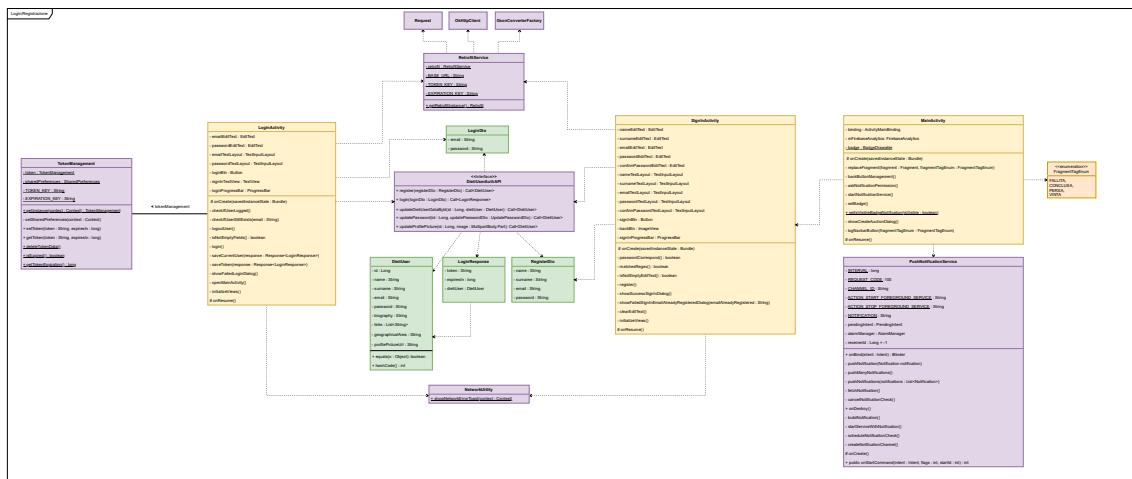
3.2 Class Diagram di Design

Qui riportiamo i Class Diagram di Design di DietiDeals24.

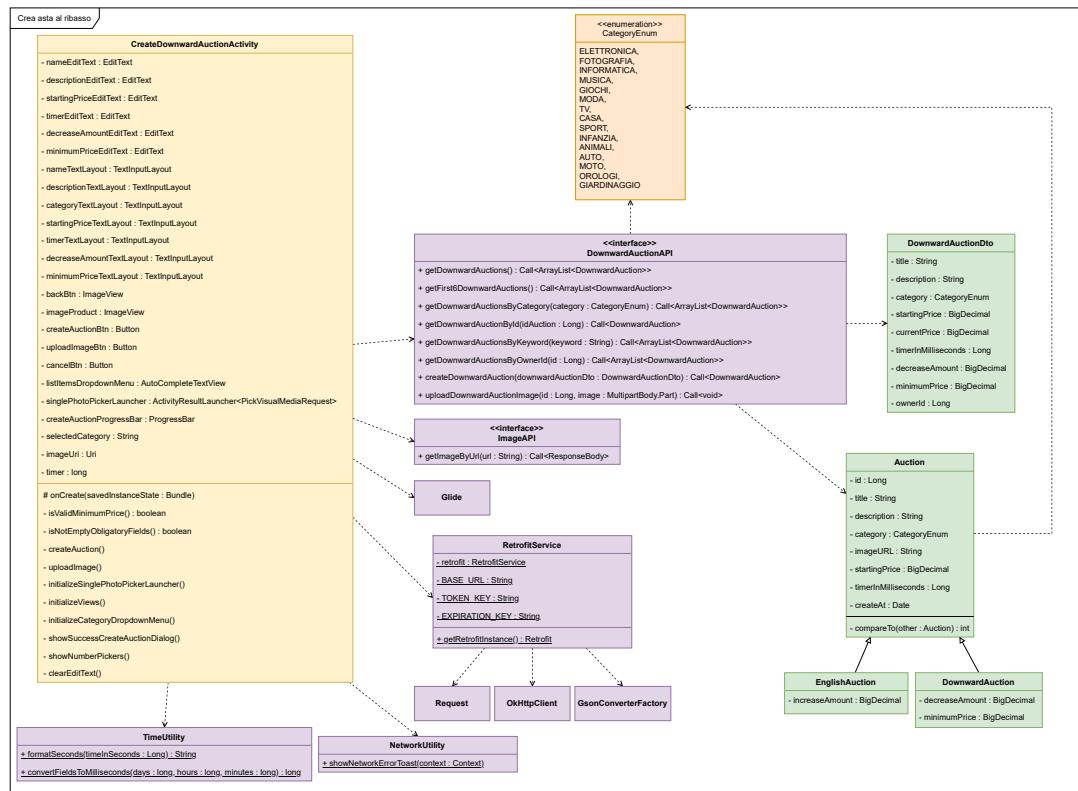
3.2.1 Entità



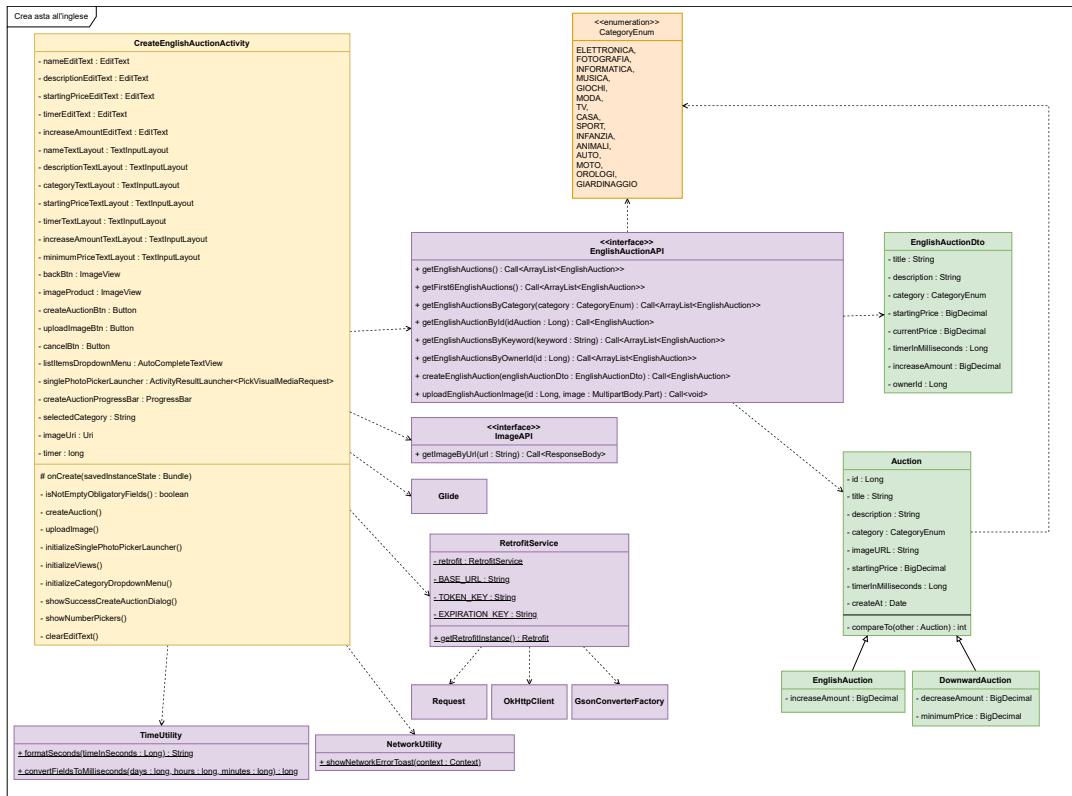
3.2.2 Login e Registrazione



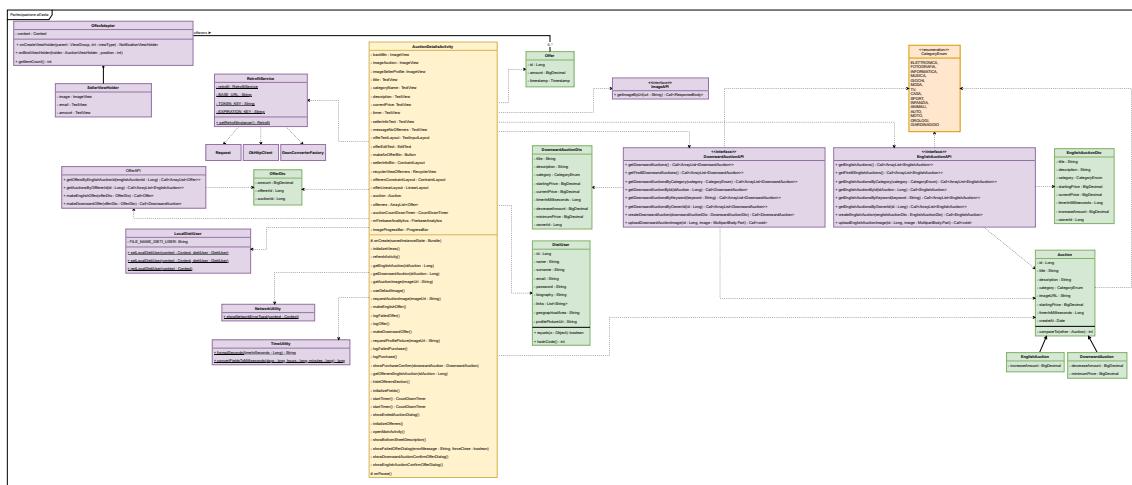
3.2.3 Creazione asta al ribasso



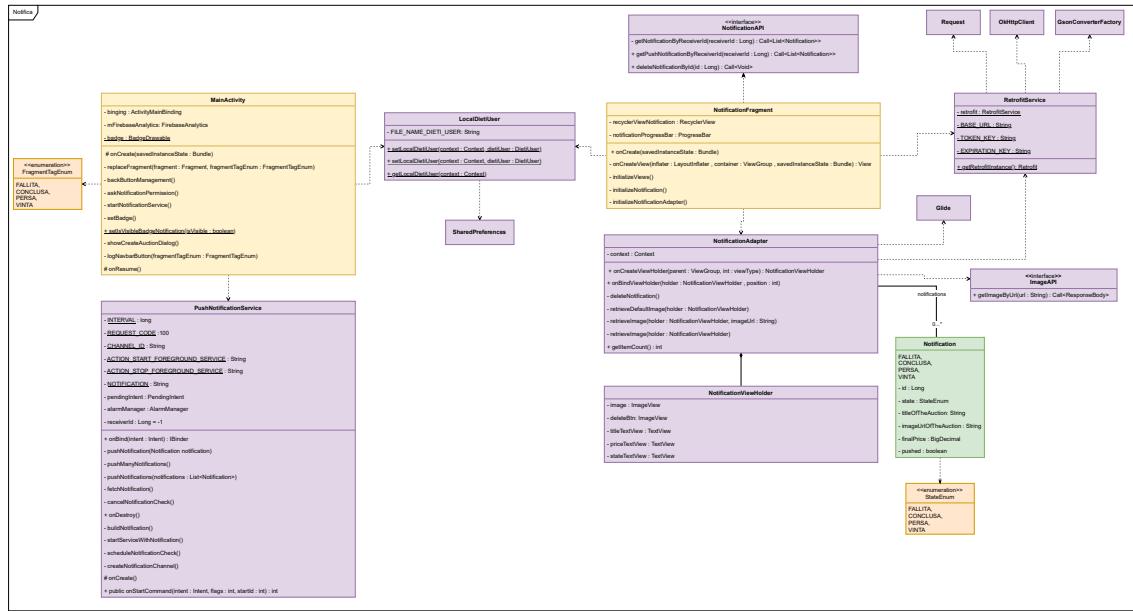
3.2.4 Creazione asta all'inglese



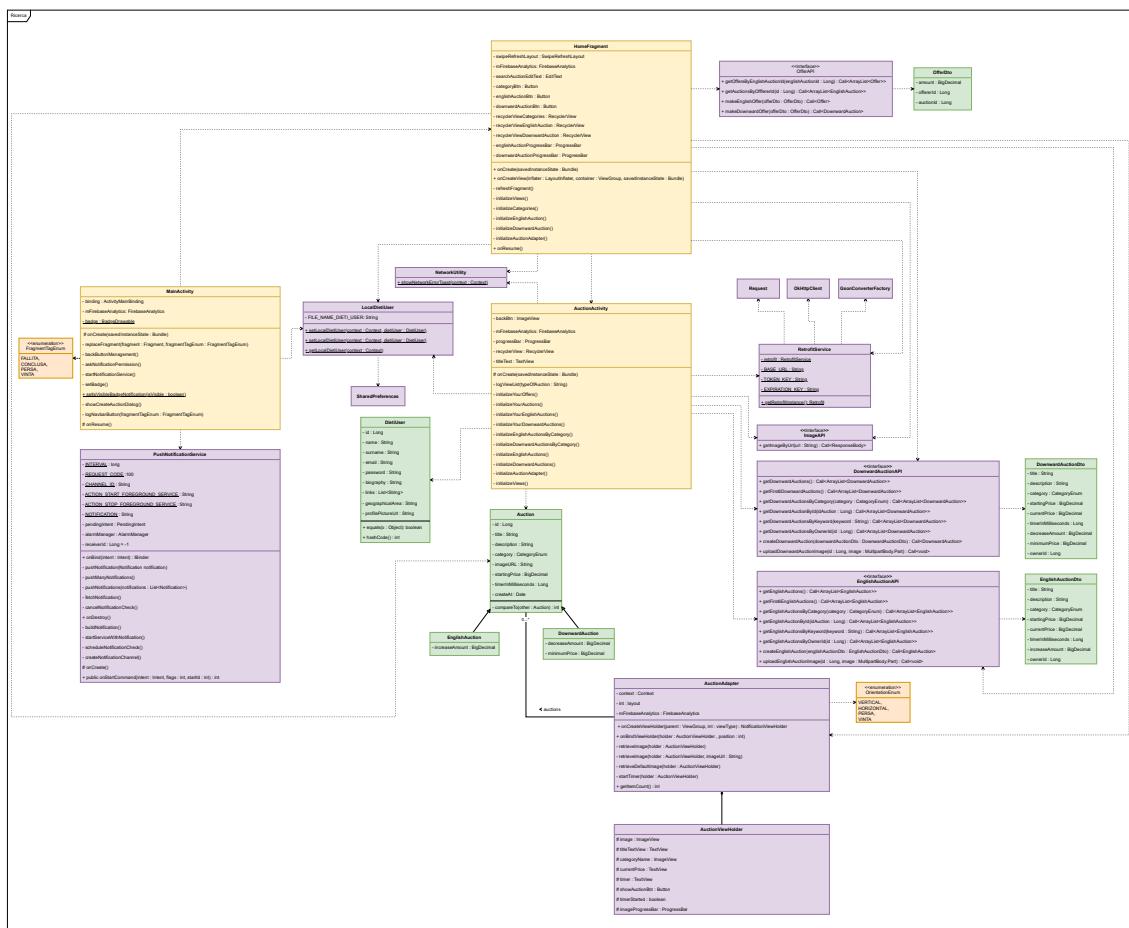
3.2.5 Piazzamento offerta



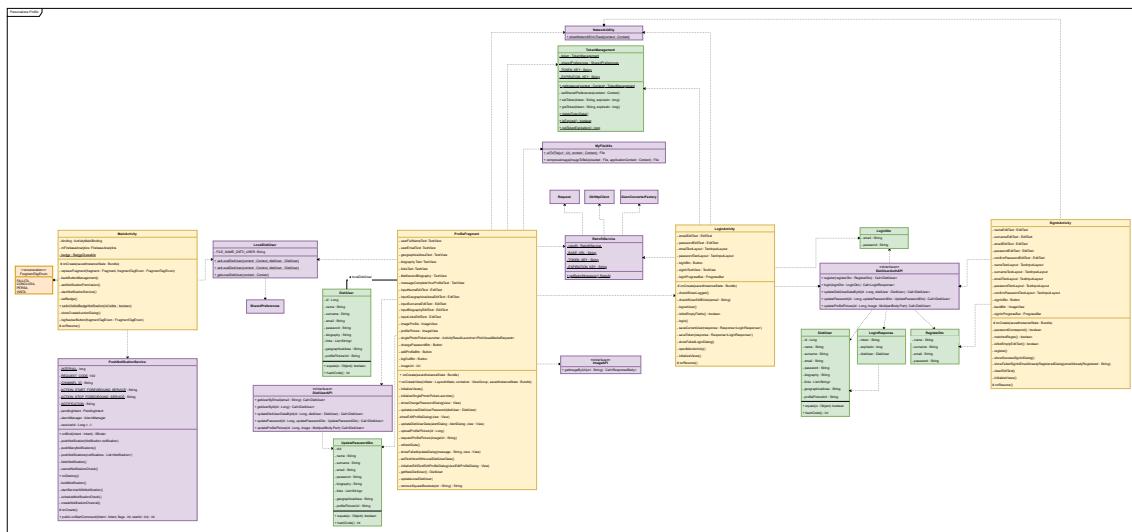
3.2.6 Notifiche



3.2.7 Ricerca



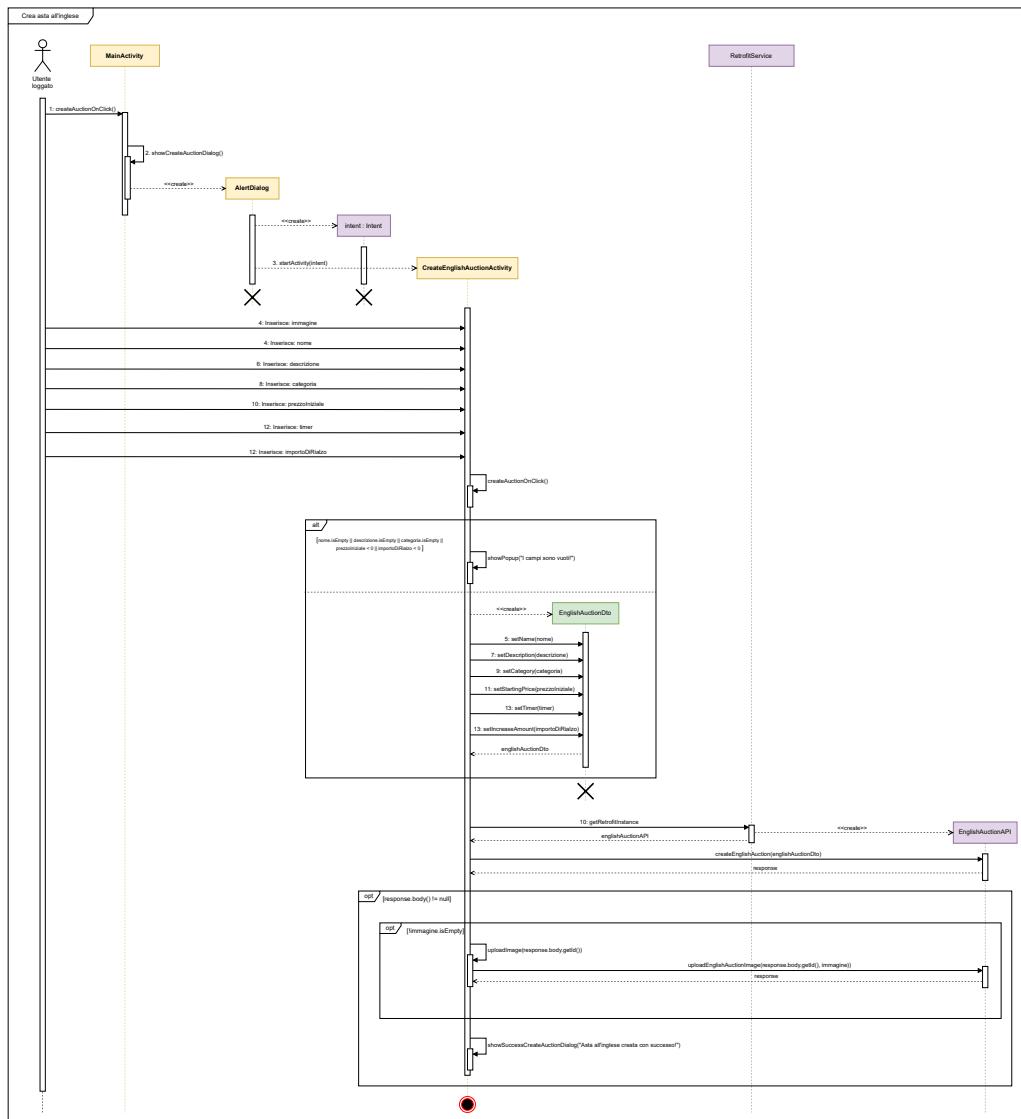
3.2.8 Personalizzazione profilo



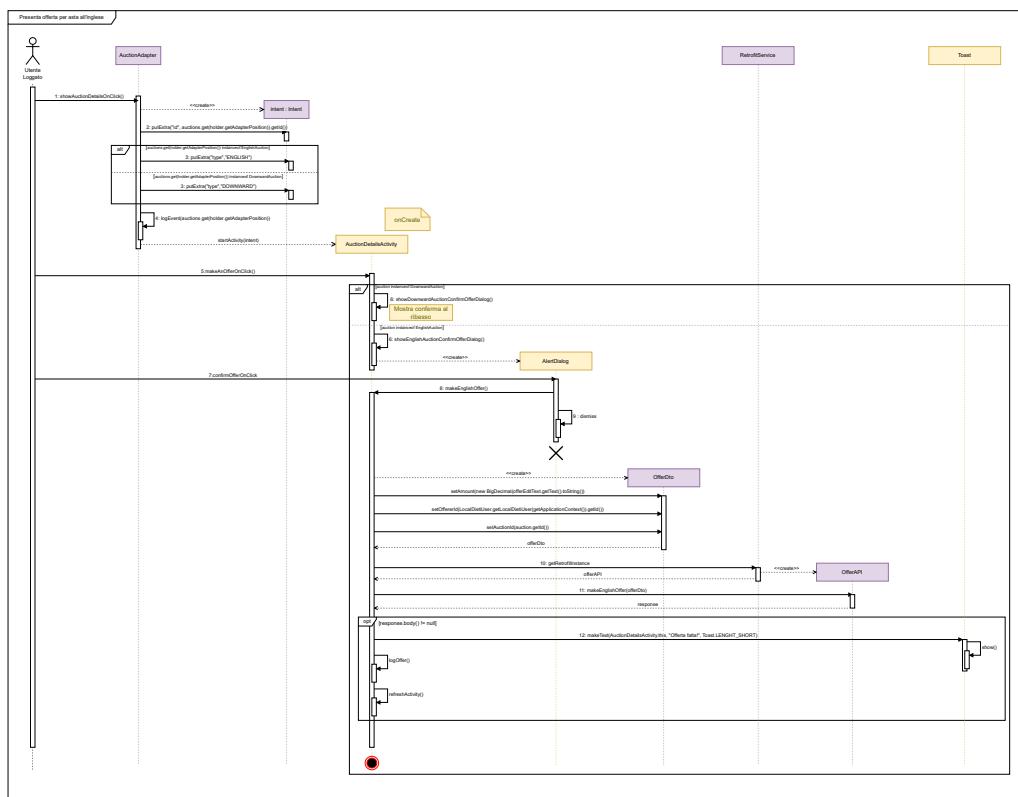
3.3 Sequence diagram di design

Qui riportiamo i Sequence Diagram di Design per le funzionalità "*Crea asta all'inglese*" e "*Presenta offerta per asta all'inglese*".

3.3.1 Crea asta all'inglese



3.3.2 Presenta offerta per asta all'inglese



Capitolo 4

Codice sorgente

4.1 Docker

Il Server di DietiDeals24 è formato da due container Docker creati mediante Docker Compose. Di seguito riportiamo il "compose.yaml" e il "Dockerfile" relativo al Backend.

4.1.1 Docker Compose

```
services:
  database:
    image: postgres:16.1
    ports:
      - 5432:5432
    environment:
      POSTGRES_PASSWORD: <your_password>
  server:
    build: ./Server
    ports:
      - 9090:9090
    environment:
      - SPRING_DATASOURCE_URL=jdbc:postgresql://database:5432/postgres
```

4.1.2 Dockerfile

```
FROM maven:3.8.4-openjdk-17-slim AS builder
COPY . /app
WORKDIR /app
RUN mvn clean package

FROM amazoncorretto:21
COPY --from=builder /app/target/Server-0.0.1-SNAPSHOT.jar app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

4.2 GitHub

Per sviluppare DietiDeals24 abbiamo adottato come strumento di versionamento GitHub. La repository è accessibile al seguente link: https://github.com/Claudio2ooo/INGSW2324_34

4.3 Report di qualità

SonarQube è uno strumento di analisi della qualità del codice, in varie modalità in base alla versione utilizzata. Nel corso dello sviluppo abbiamo usato il plugin **SonarLint** per l'analisi in tempo reale. Nella fase finale dello sviluppo, abbiamo caricato il nostro progetto su SonarQube per ottenere un report di qualità completo, con il quale abbiamo ulteriormente ripulito il codice, fino ad un livello che abbiamo ritenuto soddisfacente. Grazie a SonarLint, le problematiche riscontrate nel primo report di SonarQube erano comunque limitate e facilmente gestibili, e la risoluzione di esse ci ha portato alla situazione descritta dal report mostrato di seguito.



Capitolo 5

Testing

5.1 Codice JUnit per Unit Testing

Per la progettazione dei test abbiamo deciso di adottare la strategia **Black Box**. In particolare, per la definizione dei metodi di test abbiamo adottato la strategia **R-WECT**.

I metodi scelti per il testing:

- Client: `long convertFieldsToMilliseconds(long days, long hours, long minutes)`, metodo che dati in input i giorni, le ore e i minuti li converte in millisecondi, ritornando la somma di essi.
- Server: `void updateData(DietiUser toBeUpdated, DietiUser newDietiUser)`, metodo che aggiorna le informazioni di un utente.

5.1.1 Metodo convertFieldsToMilliseconds

Per il testing del metodo `long convertFieldsToMilliseconds(long days, long hours, long minutes)` sono state individuate le seguenti classi di equivalenza:

Nota: per "Valida" intendiamo che il valore compreso in una certa classe di equivalenza è un valore corretto, e quindi porterà ad un'esecuzione corretta del metodo. Mentre per "Non valida" intendiamo valori non corretti, che lanceranno un'eccezione.

Parametro	Classe di equivalenza	Stato
long days	CE1: $\{days \geq 0\}$	Valida
	CE2: $\{days < 0\}$	Non Valida
long hours	CE1: $\{0 \leq hours < 24\}$	Valida
	CE2: $\{hours < 0\}$	Non Valida
	CE2: $\{hours \geq 24\}$	Non Valida
long minutes	CE1: $\{0 \leq minutes < 60\}$	Valida
	CE2: $\{minutes < 0\}$	Non Valida
	CE2: $\{minutes \geq 60\}$	Non Valida

Di seguito è riportata la suite di test per il metodo `convertFieldsToMilliseconds`:

```

class TimeUtilityTest {

    @Test
    void testConvertFieldsToMilliseconds_covers_1_1_1() throws TimePickerException{
        long result = TimeUtility.convertFieldsToMilliseconds(30, 12, 30);
        long oracle = 263700000L;
        assertEquals(oracle, result);
    }

    @Test
    void testConvertFieldsToMilliseconds_covers_2_1_1() {
        assertThrows(TimePickerException.class, () -> TimeUtility.convertFieldsToMilliseconds(-25,
            ↳ 12, 30));
    }

    @Test
    void testConvertFieldsToMilliseconds_covers_1_2_1() {
        assertThrows(TimePickerException.class, () -> TimeUtility.convertFieldsToMilliseconds(14,
            ↳ -120, 30));
    }

    @Test
    void testConvertFieldsToMilliseconds_covers_1_3_1() {
        assertThrows(TimePickerException.class, () -> TimeUtility.convertFieldsToMilliseconds(12,
            ↳ 152, 30));
    }

    @Test
    void testConvertFieldsToMilliseconds_covers_1_1_2() {
        assertThrows(TimePickerException.class, () -> TimeUtility.convertFieldsToMilliseconds(10,
            ↳ 12, -12));
    }

    @Test
    void testConvertFieldsToMilliseconds_covers_1_1_3() {
        assertThrows(TimePickerException.class, () -> TimeUtility.convertFieldsToMilliseconds(19,
            ↳ 12, 300));
    }
}

```

5.1.2 Metodo updateData

Per il testing del metodo `void updateData(DietiUser toBeUpdated, DietiUser newDietiUser)` sono state individuate le seguenti classi di equivalenza:

Nota: per "Valida" intendiamo che il valore compreso in una certa classe di equivalenza è un valore corretto, e quindi, portando un'esecuzione corretta del metodo. Mentre per "Non valida" intendiamo valori non corretti, portando al lancio di un'eccezione.

Parametro	Classe di equivalenza	Stato
DietiUser toBeUpdated	CE1: toBeUpdated != null	Valida
	CE2: toBeUpdated = null	Non Valida
DietiUser newDietiUser	CE1: newDietiUser != null	Valida
	CE2: newDietiUser = null	Non Valida

Di seguito è riportata la suite di test per il metodo `updateData`:

```
class DietiUserServiceTest {

    DietiUserService dietiUserService = new DietiUserService();

    DietiUser toBeUpdated;
    DietiUser newDietiUser;

    @BeforeEach
    void setup() {
        toBeUpdated = new DietiUser("Mario", "Rossi", "example@mail.com", "foo", "Napoli", null);
        newDietiUser = new DietiUser("Mariano", "Rossi", "example@mail.com", "Extended biography",
            "Napoli", null);
    }

    @Test
    void updateData_covers_1_1() {
        dietiUserService.updateData(toBeUpdated, newDietiUser);
        assertEquals(new DietiUser("Mariano", "Rossi", "example@mail.com", "Extended biography",
            "Napoli", null), toBeUpdated);
    }

    @Test
    void updateData_covers_2_1() {
        assertThrows(NullPointerException.class, () -> dietiUserService.updateData(null,
            newDietiUser));
    }

    @Test
    void updateData_covers_1_2() {
        assertThrows(NullPointerException.class, () -> dietiUserService.updateData(toBeUpdated,
            null));
    }

    @Test
    void updateData_covers_2_2() {
        assertThrows(NullPointerException.class, () -> dietiUserService.updateData(null, null));
    }
}
```

5.2 Valutazione dell'usabilità sul campo

Per la valutazione dell'usabilità sul campo sono stati intervistati tre utenti:

- Paola non è un'esperta di dominio ed ha una esperienza basilare dell'uso di applicazioni mobile;
- Giorgia non è un'esperta di dominio ed ha esperienza avanzata dell'uso di applicazioni mobile;
- Alessandro è un esperto di dominio ed ha esperienza avanzata dell'uso di applicazioni mobile.

Ogni utente ha sostenuto test di compito per ogni funzionalità riportata negli Use Case Diagram. I test sono stati eseguiti sul prodotto nelle fasi finali dello sviluppo. Sono stati considerati successi tutti i test in cui l'utente arriva al goal sia in maniera diretta sia tramite piccole variazioni. Sono stati considerati successi parziali tutti i test in cui l'utente ha visualizzato almeno una schermata non inerente al compito richiesto. Per ogni successo parziale è riportato tra parentesi il numero di interazioni non necessarie.

	Registrazione	Login	Ricerca per keyword	Ricerca per categoria	Creazione di un'asta	Piazzamento di un'offerta
Paola	S	S	P(2)	S	P(2)	P(2)
Giorgia	S	S	P(2)	S	S	S
Alessandro	S	S	S	S	S	S
	Visualizzazione proprie aste	Visualizzazione dettagli venditore	Visualizzazione notifiche	Modifica del profilo		
Paola	P(2)	S	S	S	S	S
Giorgia	P(1)	S	S	S	S	S
Alessandro	S	S	S	S	S	S

Tabella 5.1: S = Successo, P = Successo Parziale, F = Fallimento

Per calcolare il tasso di successo abbiamo assegnato ai successi valore 1 e ai successi parziali valore 0.5. Il tasso di successo risultante è:

$$T = (24 + 6 * 0.5) / 30 = 90\%$$

5.2.1 Analytics

5.2.1.1 Google Analytics

Google Analytics è una piattaforma che raccoglie i dati dai siti web e dalle app per creare report che forniscono informazioni sulle attività e gli eventi eseguiti dagli utenti. Abbiamo implementato Google Analytics in DietiDeals24 mediante Firebase Analytics.

5.2.1.2 Report

Di seguito mostriamo due report, il primo riguardante gli eventi registrati, il secondo riguardante le activity più visualizzate.

Nome evento	+	↓ Conteggio eventi	Totale utenti	Conteggio eventi per utente
		1.384 100% del totale	18 100% del totale	76,89 Uguale alla media
1 screen_view		562	18	31,22
2 nav_bar_click		530	17	31,18
3 user_engagement		149	15	9,93
4 view_auction		48	7	6,86
5 session_start		35	17	2,06
6 english_offer		21	4	5,25
7 first_open		15	15	1,00
8 app_remove		14	14	1,08
9 view_all_auctions		5	3	1,67
10 search		3	3	1,00

Figura 5.1: Tabella degli eventi registrati

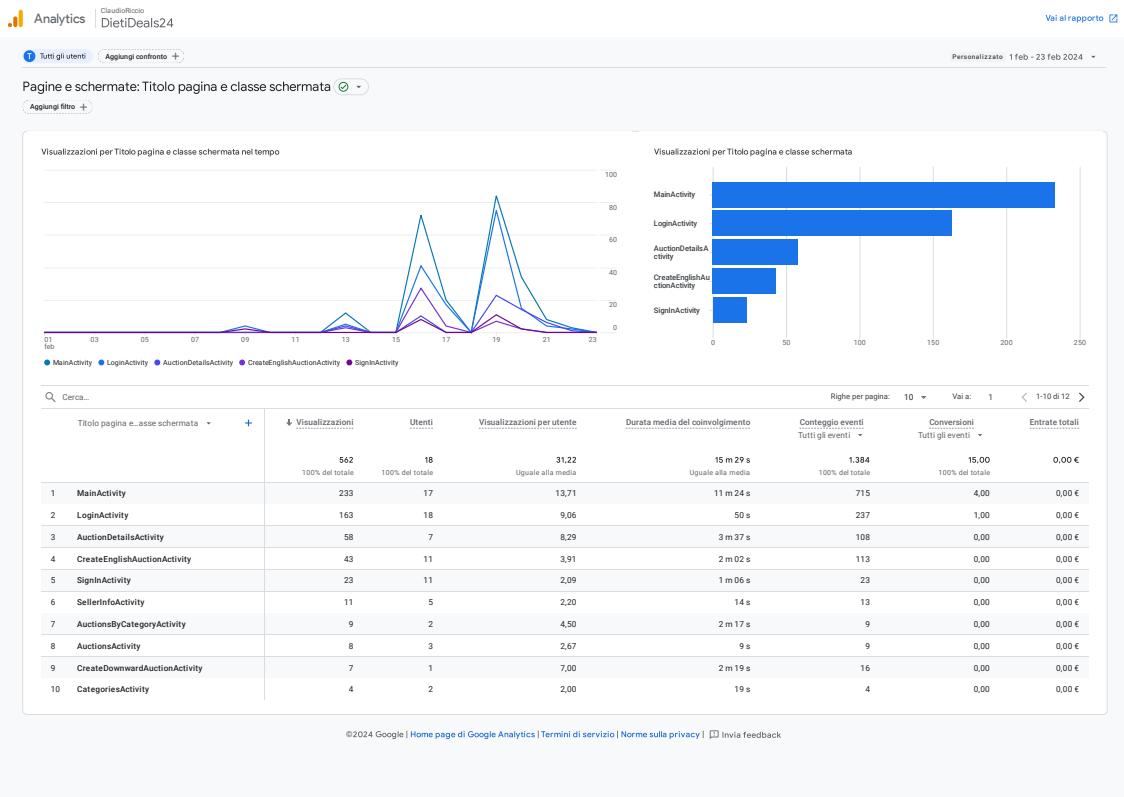


Figura 5.2: Tabella delle activity visualizzate

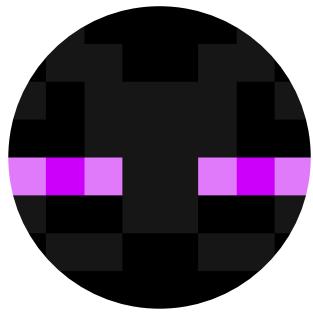
Dalle statistiche riportate, possiamo notare che gli utenti che hanno utilizzato l'app in fase di beta hanno usufruito più delle aste all'inglese. Inoltre possiamo osservare che il tempo medio per la creazione di un'asta è circa 2 minuti e 10 secondi, e quello per effettuare registrazione e login insieme ammonta a poco meno di due minuti. Questo ci dice che un utente novizio impiega circa 4 minuti per arrivare a creare un'asta, un tempo che riteniamo soddisfacente, considerando la quantità di informazioni da inserire, e che ci fa pensare che la valutazione dell'usabilità a priori abbia portato i suoi frutti.

Capitolo 6

Contributori



Salvatore Franzese
N86003142
salvator.franzese@studenti.unina.it
salvatorefranzese99@gmail.com



Claudio Riccio
N86003291
clau.riccio@studenti.unina.it
riccioclaudio2000@gmail.com