

Documentazione del software

DietiDeals24

Roberto Ingenito

N86004077

Simone Ingenito

N86004063

Lorenzo Sequino

N86004367

Anno accademico 2023/2024

Indice

1	Introduzione	4
2	Analisi dei Requisiti	5
2.1	Obiettivi della Raccolta dei Requisiti	5
2.2	Requisiti individuati	6
2.2.1	Requisiti funzionali	6
2.2.2	Requisiti non funzionali	7
2.2.3	Requisiti di dominio	8
2.3	Modello dei casi d'uso	9
2.3.1	Tabella	9
2.3.2	Descrizioni dettagliate	9
2.3.3	Diagrammi	12
2.4	Individuazione del target degli utenti.	13
2.4.1	Descrizione del Servizio	13
2.4.2	Target Demografici	13
2.4.3	Target Geografici	13
2.4.4	Target Comportamentali	13
2.4.5	Fonti	14
2.5	Tabelle di Cockburn	15
2.5.1	Creazione asta silenziosa	15
2.5.2	Offerta asta silenziosa	16
2.5.3	Aggiungere link social	17
2.5.4	Visualizzazione delle aste create ancora attive	18
2.6	Prototipazione visuale attraverso Mockup	19
2.6.1	Accesso e registrazione	19
2.6.2	Home	20
2.6.3	Storico aste	21

2.6.4	Storico aste - pagine di dettaglio	22
2.6.5	Profilo	23
2.6.6	Profilo Venditore	24
2.6.7	Piazzare un'offerta	25
2.6.8	Creare un'asta	26
2.7	Valutazione dell'usabilità a priori	27
2.7.1	Euristiche di Nielsen	27
2.7.2	Glossario	29
3	Specifiche dei requisiti	31
3.1	Modelli di dominio	31
3.2	Sequence diagram di due casi d'uso	43
3.2.1	Creazione asta all'inglese	43
3.2.2	Piazzare un'offerta su un'asta al ribasso	44
3.3	Progettazione degli Event-Based Statecharts	45
3.3.1	Creazione asta silenziosa	45
3.3.2	Offerta asta silenziosa	46
3.3.3	Aggiungi link social	47
3.3.4	Visualizzare asta create	48
4	Documento di design	49
4.1	Architettura del Software	49
4.1.1	Client	49
4.1.2	Server	49
4.1.3	Interazione tra Client e Server	50
4.2	Design pattern utilizzati	50
4.2.1	Model-View-Controller (MVVM)	50
4.3	Analisi delle scelte tecnologiche utilizzate	51
4.3.1	Client: Confronto con altre tecnologie	51
4.3.2	Server	51
4.4	Diagrammi UML di Design	53
4.4.1	View	53
4.4.2	Model	53
4.4.3	ViewModel	54

4.4.4	Routes	55
4.4.5	Services	55
4.5	Diagrammi di Sequenza di Design	56
4.5.1	Caso d'uso: Creazione Asta Silenziosa	56
4.5.2	Caso d'uso: Offerta Asta Silenziosa	57
4.5.3	Caso d'uso: Aggiunta Link Social	58
4.5.4	Caso d'uso: Storico Aste create	59
5	Strumento di Versioning	60
5.1	Scelte adottate	60
5.1.1	Vantaggi di GitHub come strumento di versioning	60
5.2	Repository Client	61
5.3	Repository Server	61
6	Report di Qualità del Codice	62
6.1	SonarQube	62
7	Testing e Valutazione sul campo dell'Usabilità	64
7.1	Unit Testing	64
7.1.1	checkEmailAndPassword	64
7.1.2	checkPriceDifference	66
7.1.3	checkNewOfferPrice	69
7.1.4	checkBaseAndRaiseThreshold	71
7.2	Valutazione dell'Usabilità sul campo	72
7.2.1	Obiettivi della valutazione sul campo	72
7.2.2	Metodologia	73
7.2.3	Valutazione delle euristiche di Nielsen	73
7.2.4	Conclusioni e raccomandazioni	76

Introduzione

DietiDeals24 è un'app mobile progettata per consentire agli utenti di creare e partecipare a diverse tipologie di aste in modo semplice e immediato. L'app supporta tre principali modalità di asta:

- **Asta all'inglese:** Gli utenti fanno offerte incrementali, e l'oggetto va al miglior offerente.
- **Asta al ribasso:** L'asta parte da un prezzo massimo che diminuisce progressivamente fino a quando un utente decide di accettare il prezzo corrente.
- **Asta silenziosa:** Gli utenti inviano offerte in modo anonimo, e il vincitore viene determinato alla fine dell'asta senza che le offerte siano rese visibili durante il processo.

Gli utenti possono scegliere di partecipare come venditori, creando aste per i loro prodotti o servizi, oppure come acquirenti, facendo offerte per aggiudicarsi gli articoli in vendita.

Analisi dei Requisiti

2.1 Obiettivi della Raccolta dei Requisiti

La raccolta dei requisiti è una fase fondamentale per garantire che il prodotto finale soddisfi gli stakeholder e gli obiettivi prefissati.

L'obiettivo principale di questa fase è identificare, analizzare e documentare le esigenze e le aspettative delle parti interessate, in particolare sono stati individuati tre tipi di requisiti: funzionali, non funzionali e di dominio.

- I requisiti funzionali definiscono le funzionalità che il sistema deve offrire, descrivendo come il software interagirà con gli utenti e con altri sistemi.
- I requisiti non funzionali, riguardano le qualità del sistema, come le prestazioni, la sicurezza, l'usabilità e la scalabilità.
- I requisiti di dominio definiscono le regole di comportamento che il sistema deve applicare.

2.2 Requisiti individuati

2.2.1 Requisiti funzionali

- l'utente può registrare un nuovo account
- l'utente può utilizzare un'account da lui registrato per accedere al sistema
- l'utente può registrarsi utilizzando credenziali di terze parti (es.: google, facebook, ...)
- l'utente può personalizzare il proprio profilo (non obbligatorio):
 - aggiungere nome e cognome
 - aggiungere una breve bio
 - link al proprio sito web
 - link ai propri social
- l'utente può creare un'asta
 - inserisce la foto
 - inserisce la descrizione
 - inserisce la categoria
 - asta all'inglese
 - il venditore inserisce una base d'asta (visibile a tutti)
 - L'asta include un intervallo di tempo fisso per presentare nuove offerte (di default 1 ora), l'utente può impostare un valore a sua scelta
 - L'asta include una soglia di rialzo (di default 10€), l'utente può impostare un valore a sua scelta
 - l'utente può creare un asta al ribasso e inserisce:
 - un prezzo iniziale
 - la durata del timer per il decremento del prezzo (di default 1 ora).
 - l'importo per ciascun decremento
 - il prezzo minimo segreto (non visibili pubblicamente).
 - l'utente può creare un asta silenziosa
 - inserisce data e ora di scadenza
 - può accettare un'offerta
- l'utente può presentare un'offerta (per qualsiasi tipo d'asta)
- l'utente può cercare un'asta per parola chiave
- l'utente può filtrare le aste per categoria (tecnologia, giocattoli, ...)
- l'utente può visualizzare i dettagli di un'asta
- l'utente può visualizzare il profilo del venditore dell'asta
- Visualizzazione storico aste create (in corso e concluse)
- Visualizzazione storico a cui hai partecipato (in corso e concluse)

2.2.2 Requisiti non funzionali

- il sistema deve gestire la concorrenza in caso di offerte uguali contemporaneamente:

Se, in un'asta all'inglese, più compratori fanno la stessa offerta contemporaneamente, solo uno avrà la meglio.

Stesso discorso per un'asta al ribasso, se più compratori decidono di acquistare l'asta a quel prezzo, solo uno avrà la meglio.

1. **Usabilità:**

L'interfaccia utente deve essere intuitiva e facile da utilizzare, con una chiara visualizzazione delle informazioni relative alle aste e ai profili utenti.

2. **Performance:**

Il sistema deve gestire in modo efficiente un grande numero di aste e utenti, garantendo tempi di risposta rapidi, specialmente durante la presentazione delle offerte e la gestione dei timer.

3. **Scalabilità:**

Il sistema deve essere scalabile per supportare un numero crescente di utenti, aste e operazioni contemporanee senza compromettere le prestazioni.

4. **Sicurezza:**

Gli utenti devono poter registrarsi e autenticarsi in modo sicuro, anche tramite provider di terze parti.

Devono essere implementate misure per proteggere le offerte e i dati personali degli utenti.

5. **Affidabilità:**

Il sistema deve essere altamente disponibile, minimizzando i tempi di inattività.

I dati relativi alle aste, alle offerte e ai profili devono essere conservati in modo sicuro e accessibili in qualsiasi momento.

2.2.3 Requisiti di dominio

- l'email è univoca per ogni account. Non possono esserci più account con una sola mail
- un account è sia venditore che compratore
- l'utente non può acquistare le aste create da lui stesso
- l'asta può essere di diversi tipi: all'inglese, al ribasso, silenziosa:
 - **Asta all'inglese:** Un formato in cui le offerte incrementano il prezzo fino alla scadenza del timer, con vittoria al miglior offerente.
 - **Asta al ribasso:** Un formato in cui il prezzo diminuisce con il tempo, e la vendita avviene al primo offerente che accetta il prezzo corrente.
 - **Asta silenziosa:** Un formato in cui le offerte sono segrete e il venditore sceglie una sola offerta da accettare.
- asta all'inglese:
 - l'offerta può essere presentata a partire dal prezzo corrente.
se è la prima offerta può offrire il prezzo base, altrimenti deve presentare un'offerta più alta rispetto al prezzo corrente.
Il rialzo dev'essere un multiplo della soglia impostata nell'offerta. Ad esempio, se la soglia impostata è 10, il cliente può rialzare di 10, 20, 30, ...
 - quando viene presentata un'offerta, il timer viene resettato e il prezzo corrente viene aggiornato
 - allo scadere del timer:
 - l'asta si conclude
 - vince l'ultima offerta fatta
 - il venditore e gli acquirenti che hanno partecipato all'asta visualizzano una notifica.
- asta al ribasso:
 - al raggiungimento del timer, il prezzo verrà decrementato dell'importo previsto e il timer riparte
 - il prodotto è venduto al primo acquirente che presenta un'offerta
 - Se il prezzo raggiunge il prezzo minimo segreto senza offerte, l'asta fallisce e il venditore riceve una notifica.
(L'offerta fallisce al raggiungimento del prezzo minimo e non al ribasso successivo al prezzo minimo)
- asta silenziosa:
 - le offerte degli utenti sono segrete, non visibili agli altri utenti ma solo al venditore
 - il venditore può accettare una sola offerta
 - quando il venditore accetta un'offerta, viene inviata una notifica all'acquirente.

2.3 Modello dei casi d'uso

2.3.1 Tabella

Attore	Casi d'uso
Utente non loggato	<ul style="list-style-type: none"> • registrazione con email • registrazione con credenziali di terze parti (Google) • Accesso al sistema
Utente non loggato (venditore/acquirente)	<ul style="list-style-type: none"> • Personalizza il profilo • Visualizza storico <ul style="list-style-type: none"> – aste create (in corso e concluse) – a cui hai partecipato (in corso e concluse) • Ricerca asta per parola chiave • Filtra asta per categoria • Visualizza dettagli di un'asta • Visualizza il profilo del venditore • Crea asta all'inglese • Crea asta al ribasso • Crea asta silenziosa • Piazza un'offerta per l'asta all'inglese • Piazza un'offerta per l'asta al ribasso • Piazza un'offerta per l'asta silenziosa • Accetta un'offerta di un'asta silenziosa

2.3.2 Descrizioni dettagliate

- **Registrazione con email.**

L'utente che desidera registrarsi, può farlo inserendo dati quali: email, nome, cognome e password. I dati inseriti sono soggetti a verifica da parte del sistema (ad esempio: la mail

deve essere in un formato valido).

Una volta effettuata la registrazione, se questa ha avuto successo, l'utente accede al sistema.

- **Registrazione con credenziali di terze parti (Google).**

L'utente può registrarsi più utilizzando il proprio account Google. Una volta effettuata la registrazione, se questa ha avuto successo, l'utente accede al sistema.

- **Accesso al sistema.**

L'utente che ha già effettuato la registrazione in precedenza può accedere al sistema utilizzando la mail con la quale si è registrato e la propria password.

- **Personalizzazione del profilo.**

Andando nella sezione “Profilo”, l'utente può visualizzare le proprie informazioni e modificarle. Può aggiungere informazioni quali: bio, link al proprio sito personale, link ai social ed area geografica.

- **Visualizzazione storico aste create (in corso e concluse).**

Andando nella sezione “Storico Aste”, l'utente può visualizzare sia le aste a cui ha preso parte (quelle in corso per cui ha piazzato un'offerta, e quelle concluse) sia le aste che ha creato (in corso e concluse).

- **Ricerca aste per parola chiave.**

Nella homepage l'utente può selezionare la barra di ricerca e inserire delle parole chiave per filtrare le aste per descrizione, in modo da trovare più velocemente ciò a cui è interessato.

Nota: Se anche il filtraggio per categoria è attivato, la ricerca viene raffinata ulteriormente (parola chiave + categoria).

- **Filtraggio aste per categoria.**

Nella homepage l'utente può selezionare dei pulsanti per filtrare le aste visualizzate per categoria.

- **Visualizza dettagli di un'asta.**

L'utente interessato ad un'asta può cliccarla per accedere alla pagina di dettaglio e visualizzare tutte le informazioni di cui ha bisogno. Da questa schermata è anche possibile fare un'offerta (se l'asta è ancora in corso) e visualizzare il profilo del venditore.

- **Visualizza il profilo del venditore.**

L'utente che sta visualizzando un'asta può cliccare sul nome del venditore per visualizzare la pagina profilo del venditore.

- **Creare asta all'inglese.**

Dalla schermata home, l'utente clicca sul pulsante “Crea Asta”, poi clicca sul pulsante ”Asta all'inglese”. Viene visualizzata una nuova schermata in cui deve inserire tutte le informazioni necessarie tra cui: immagine, descrizione, categoria, base d'asta, soglia di rialzo e il timer.

- **Creare asta al ribasso.**

Dalla schermata home, l'utente clicca sul pulsante “Crea Asta”, poi clicca sul pulsante ”Asta al ribasso”. Viene visualizzata una nuova schermata in cui deve inserire tutte le informazioni necessarie tra cui: immagine, descrizione, categoria, prezzo iniziale, prezzo minimo, importo di cui decrementare, timer prima di un decremento.

- **Creare asta silenziosa.**

Dalla schermata home, l'utente clicca sul pulsante “Crea Asta”, poi clicca sul pulsante ”Asta silenziosa”. Viene visualizzata una nuova schermata in cui deve inserire tutte le informazioni necessarie tra cui: immagine, descrizione, categoria, scadenza.

- **Piazzare un'offerta per l'asta all'inglese.**

L'utente che si trova sulla schermata di un'asta all'inglese inserisce l'importo dell'offerta che vuole piazzare, poi clicca sul pulsante ”conferma offerta”.

- **Piazzare un'offerta per l'asta al ribasso.**

L'utente che si trova sulla schermata di un'asta al ribasso può cliccare il pulsante “Acquista” ed aggiudicarsi automaticamente l'asta. Dopo quest'operazione il sistema mostra una notifica che informa l'utente dell'esito dell'asta.

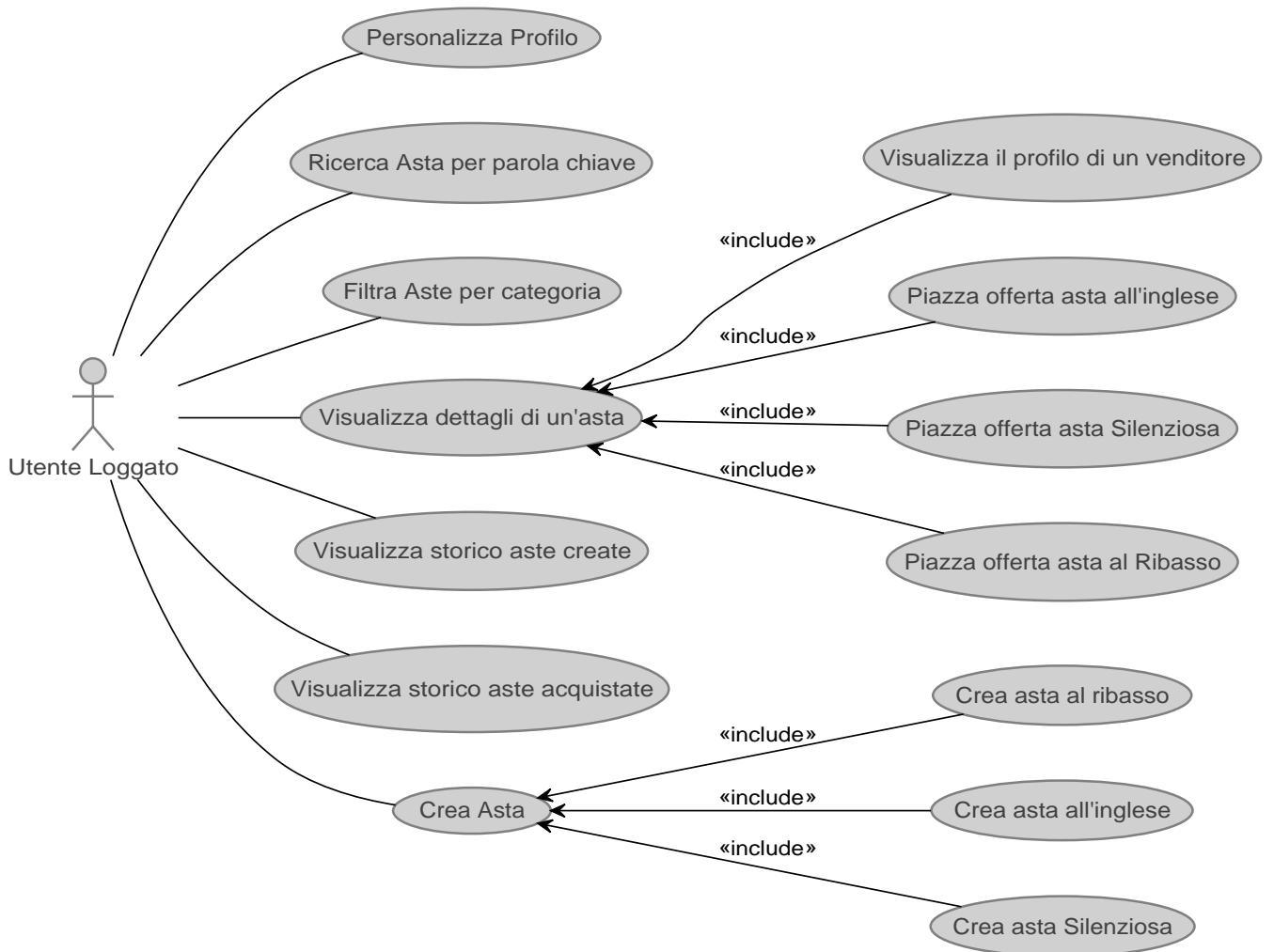
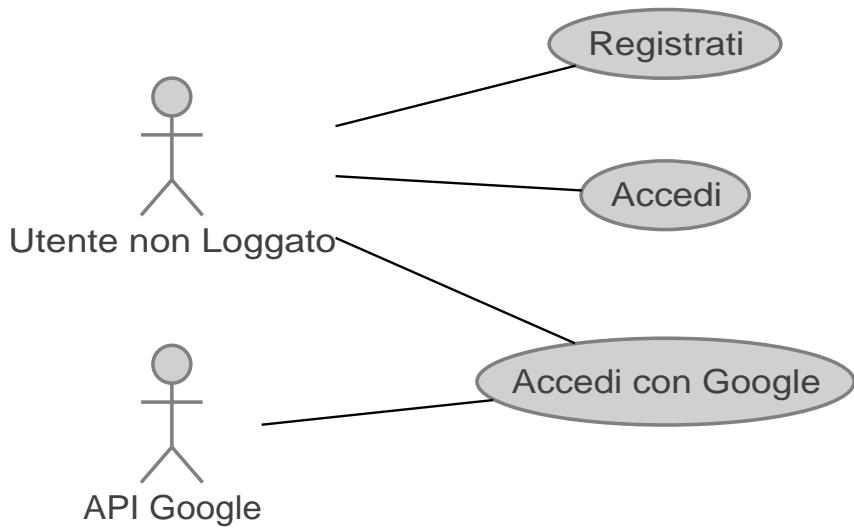
- **Piazzare un'offerta per l'asta silenziosa.**

L'utente che si trova sulla schermata di un'asta silenziosa può inserire un'offerta e cliccare sul pulsante ”conferma offerta” per inviare l'offerta.

- **Accetta un'offerta di un'asta silenziosa.**

L'utente che ha creato un'asta silenziosa, può andare nello storico delle aste create, selezionare un'asta in corso e accettare un'offerta.

2.3.3 Diagrammi



2.4 Individuazione del target degli utenti.

L'identificazione del target degli utenti è cruciale per garantire che l'applicazione soddisfi le esigenze e le aspettative dei suoi utilizzatori.

2.4.1 Descrizione del Servizio

DietiDeals24 è un'applicazione progettata per facilitare la compravendita di beni e servizi tramite aste online. Grazie a questa piattaforma, gli utenti possono partecipare a aste in tempo reale e avere l'opportunità di acquistare prodotti e servizi a prezzi vantaggiosi.

Di fatto l'applicazione si configura all'interno del settore e-commerce e questo ci ha permesso di fare analisi sul target degli utenti.

2.4.2 Target Demografici

- **Età:** Il target principale si colloca nella fascia d'età tra 25 e 45 anni, che rappresenta una parte significativa degli acquirenti online, in particolare quelli che partecipano a piattaforme di e-commerce e aste online.
- **Genere:** Tendenzialmente bilanciato, con una leggera prevalenza maschile nelle aste online, ma con un interesse crescente anche tra le donne.
- **Reddito:** Medio, con utenti che cercano offerte vantaggiose ma che dispongono di un potere d'acquisto tale da consentire la partecipazione a aste su prodotti di valore medio-alto.

2.4.3 Target Geografici

Utenti che vivono in città e usano spesso Internet per fare acquisti online. Questi servizi sono più diffusi nelle zone dove la connessione è buona e la maggior parte delle persone sa usare bene la tecnologia.

2.4.4 Target Comportamentali

Gli utenti che acquistano online tramite un servizio di aste solitamente cercano i seguenti benefici:

- **Risparmio:** acquistare prodotti a prezzi più bassi grazie alla natura competitiva delle aste.
- **Prodotti esclusivi:** Possibilità di accedere a beni rari o di valore, che potrebbero non essere facilmente disponibili nei negozi tradizionali.

- **Soddisfazione della vittoria:** Vincere un'asta può essere un'esperienza gratificante e motivante.

2.4.5 Fonti

Statista, "E-commerce user demographics in 2023", disponibile su www.statista.com.

Digital Commerce 360, "Who shops online?" disponibile su www.digitalcommerce360.com.

2.5 Tabelle di Cockburn

2.5.1 Creazione asta silenziosa

Use Case #01			
Creazione asta silenziosa			
Description	Step	User Action	System
	1	Dal mockup "Homepage" preme il pulsante "Carica Asta".	
	2		Mostra il mockup "Carica Asta" con 3 possibili scelte.
	3	Seleziona il pulsante "Crea Asta silenziosa"	
	4		Mostra il mockup "Creazione Asta silenziosa"
	5	Inserisce tutti i dati necessari e preme il pulsante "Crea Asta"	
	6		Mostra un popup con l'esito della creazione dell'asta e torna al mockup "Homepage"
Extensions	Step	User Action	System
Dati mancanti o non corretti	5.a		Mostra un popup di errore
	5.b	Visualizza il popup e riprova l'inserimento dei dati	

2.5.2 Offerta asta silenziosa

Use Case #02				Offerta asta silenziosa
Goal in Context				L'utente vuole piazzare un'offerta per un'asta silenziosa
Preconditions				L'utente deve aver effettuato l'accesso e si deve trovare nella pagina "Home"
Success End Conditions				L'utente piazza correttamente l'offerta che sarà visibile al venditore
Description	Step	User Action	System	
	1	Preme sull'asta su cui vuole fare un'offerta.		
	2		Mostra il mockup "Dettaglio asta silenziosa" dove visualizzerà i dettagli dell'asta	
	3	Inserisce l'importo che vuole puntare e poi preme il pulsante "Conferma offerta"		
	4		Il sistema mostra un messaggio di successo (mockup "Dettaglio asta silenziosa - success")	
	5	Preme il pulsante "ok"		
	6		Il sistema riporta l'utente alla pagina home	
Extensions	Step	User Action	System	
La puntata non va a buon fine, qualsiasi sia stato l'errore	3.a		Il sistema mostra una finestra indicando l'errore (mockup "Dettaglio asta silenziosa - error")	
	3.b	Preme il pulsante "ok"		
	3.c		Il sistema chiude questa finestra senza cambiare pagina.	

2.5.3 Aggiungere link social

Use Case #03				
Aggiungere link social				
Goal in Context				
L'utente vuole aggiungere un nuovo link social				
Preconditions				
L'utente deve aver effettuato l'accesso e si deve trovare nella pagina "Profilo"				
Success End Conditions				
L'utente inserisce correttamente un nuovo link social				
Description		Step	User Action	
		1	Preme il pulsante "Modifica"	
		2		
		3	Preme il pulsante "aggiungi link"	
		4		
		5	Inserisce il link e preme il pulsante "Conferma"	
		6		
		7	Preme il pulsante "salva"	
		8	Torna alla pagina profilo	
Extensions		Step	User Action	
Link non valido		5.a	Mostra il messaggio "Link non valido" appena sotto il campo di testo colorandolo di rosso	
		5.b	Ripete lo step 5	

2.5.4 Visualizzazione delle aste create ancora attive

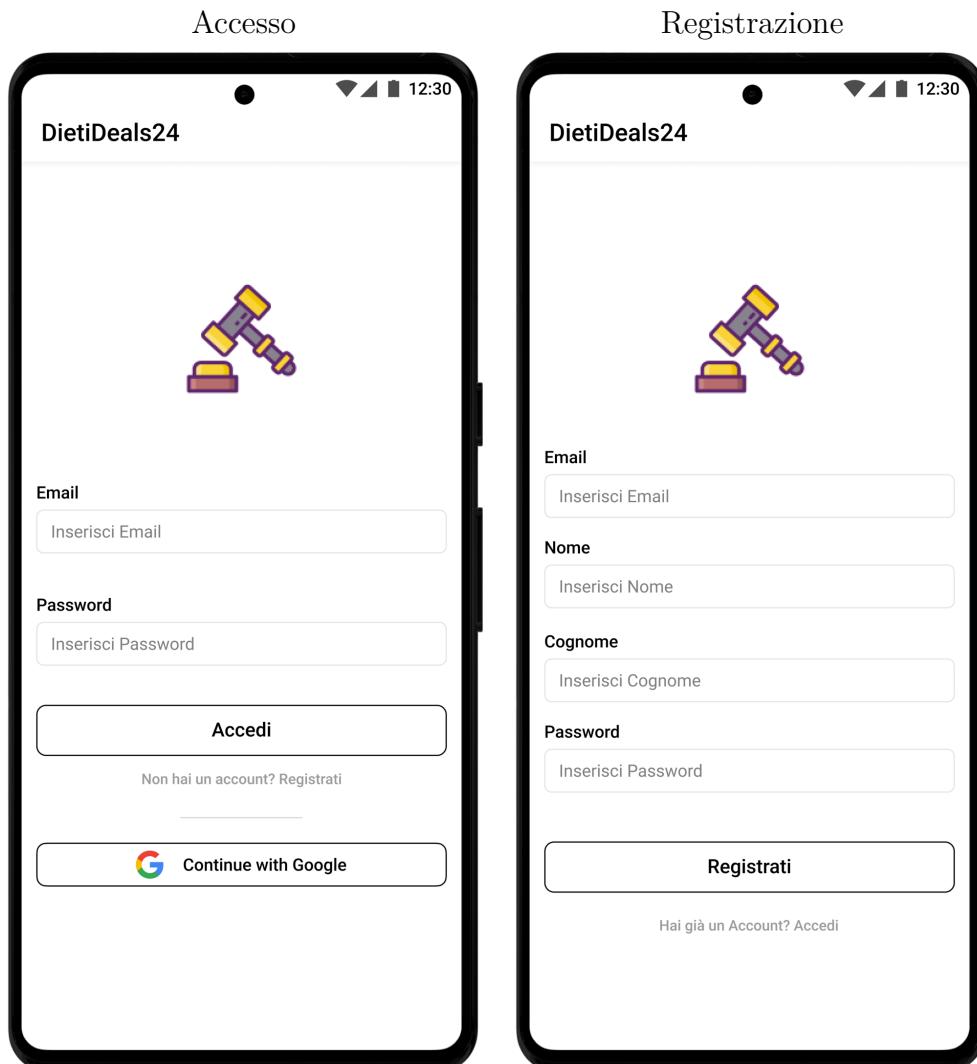
Use Case #04			
Visualizzazione delle aste create ancora attive			
Description			
Step	User Action	System	
1	Preme sul pulsante "storico aste" nella barra di navigazione in basso		
2		Mostra il mockup "Storico aste - acquistate"	
3	Preme il pulsante "Aste create"		
4		Mostra il mockup "Storico aste - create"	
5	L'utente visualizza le aste che ha creato e riconosce quelle attive grazie a un riquadro blu in alto a destra con la dicitura "in corso"		
Extensions	Step	User Action	System
L'utente non ha ancora creato aste	4.a		Mostra il messaggio "Non hai ancora creato nessun'asta"

2.6 Prototipazione visuale attraverso Mockup

I mock-up sono stati progettati con Figma. Di seguito viene riportata la lista completa dei prototipi realizzati.

2.6.1 Accesso e registrazione

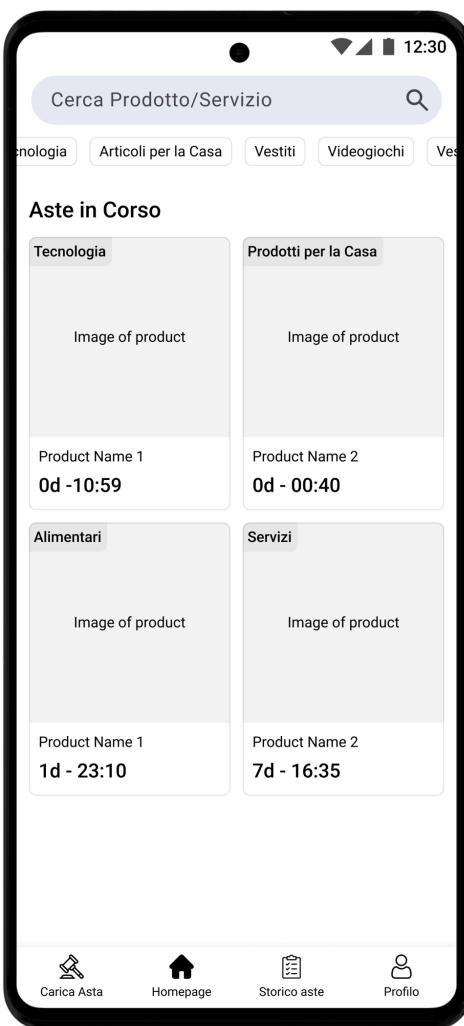
Il primo mockup mostra la schermata di accesso dove è possibile accedere con email e password oppure tramite google. Inoltre, è presente un pulsante per navigare alla pagina di registrazione. Il secondo mockup mostra la schermata di registrazione dove l'utente compilerà i vari campi e premere il pulsante "Registrati" per confermare la registrazione. Inoltre, è presente un pulsante per navigare alla pagina di accesso.



2.6.2 Home

Il mockup mostra la pagina principale dell'app, progettata per visualizzare e interagire con le aste in corso. Questi sono gli elementi chiave:

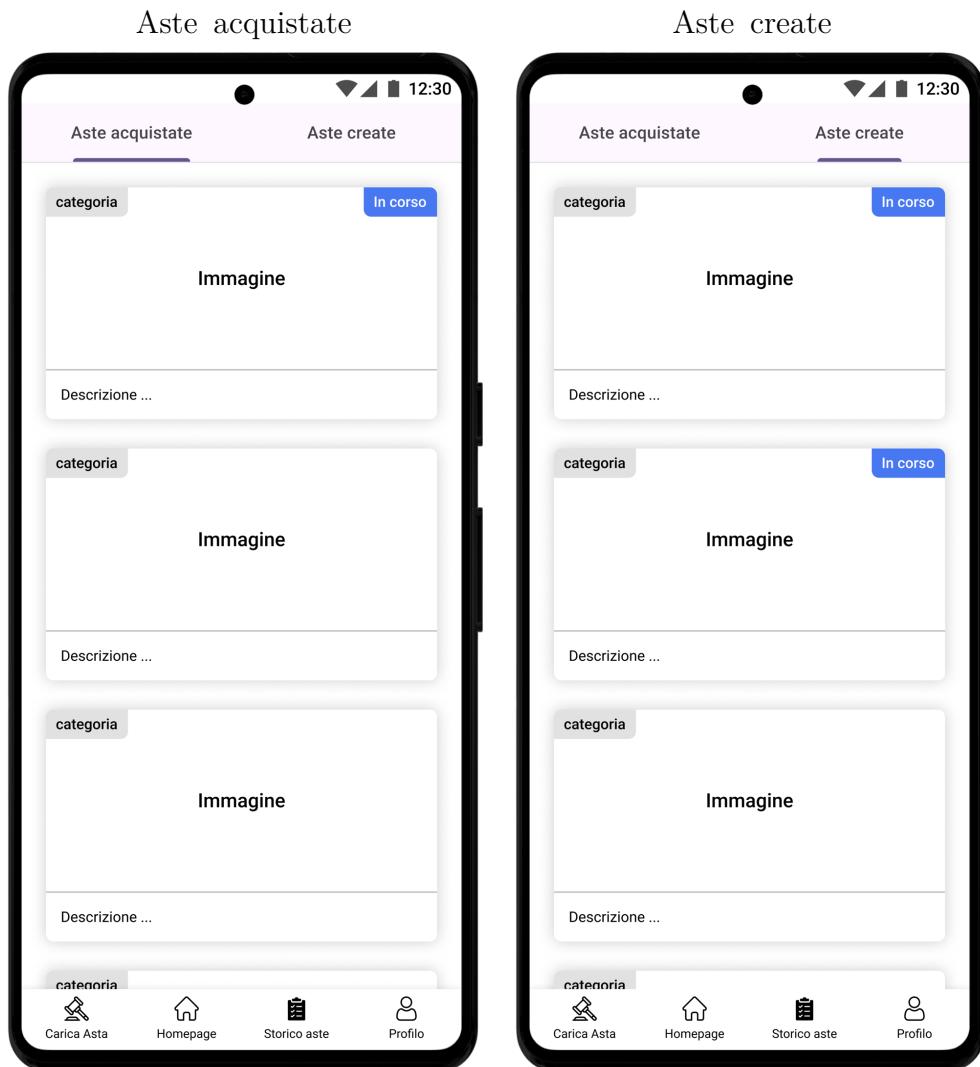
- Una barra di ricerca in alto per cercare le aste in base alla loro descrizione.
- La lista delle categorie scorrevole orizzontalmente, cliccando su una di queste, verranno mostrate solo le aste di quella categoria.
- Una sezione "Aste in Corso" che mostra le aste attive.



2.6.3 Storico aste

L'interfaccia è progettata per essere intuitiva e facilmente navigabile, permettendo agli utenti di visualizzare rapidamente le informazioni essenziali sulle loro aste, sia quelle a cui hanno partecipato come acquirenti, sia quelle che hanno creato come venditori.

È possibile scorrere tra la lista delle aste acquistate e quelle create, mediante la barra di navigazione posta in alto oppure scorrendo verso destra o sinistra.



2.6.4 Storico aste - pagine di dettaglio

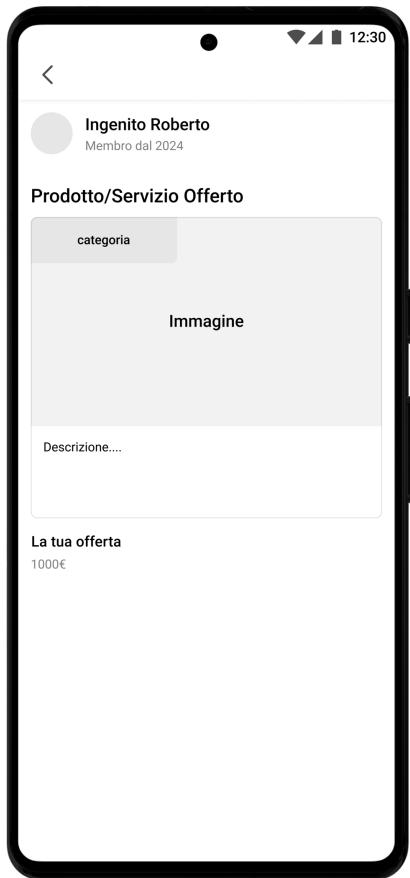
Facendo click su un'asta, nella pagina dello storico delle aste, verrà mostrato uno dei seguenti mockup in base alla condizione dell'asta (in corso / terminata).

Il primo mockup mostra la pagina di dettaglio di un'asta acquistata, in corso oppure terminata.

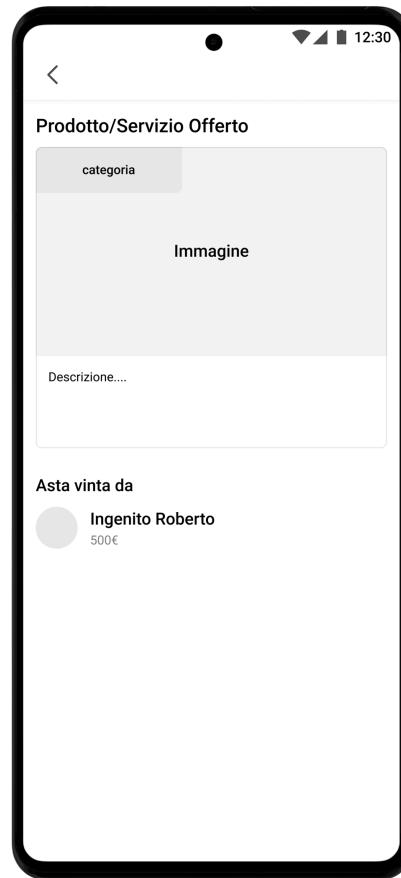
Il secondo mockup mostra la pagina di dettaglio di un'asta venduta, quindi terminata.

Il terzo mockup mostra la pagina di dettaglio di un'asta silenziosa ancora in corso, dov'è possibile accettare una tra le offerte presentate.

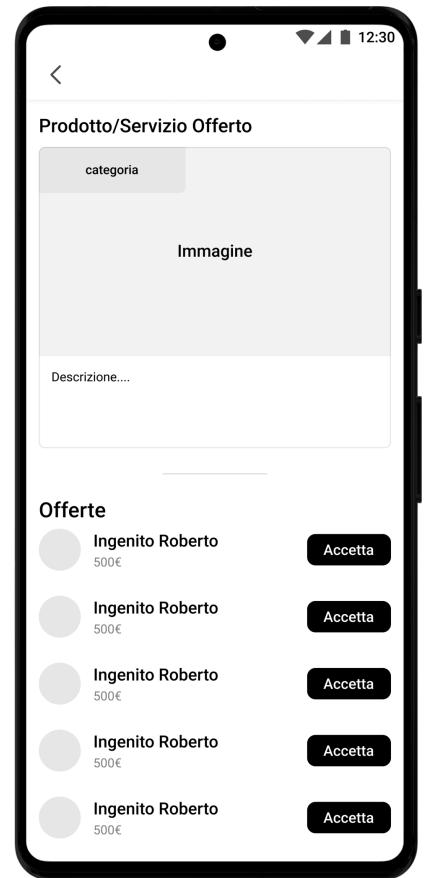
Dettagli asta acquistata



Dettagli asta venduta
(terminata)



Dettagli asta silenziosa
(in corso)

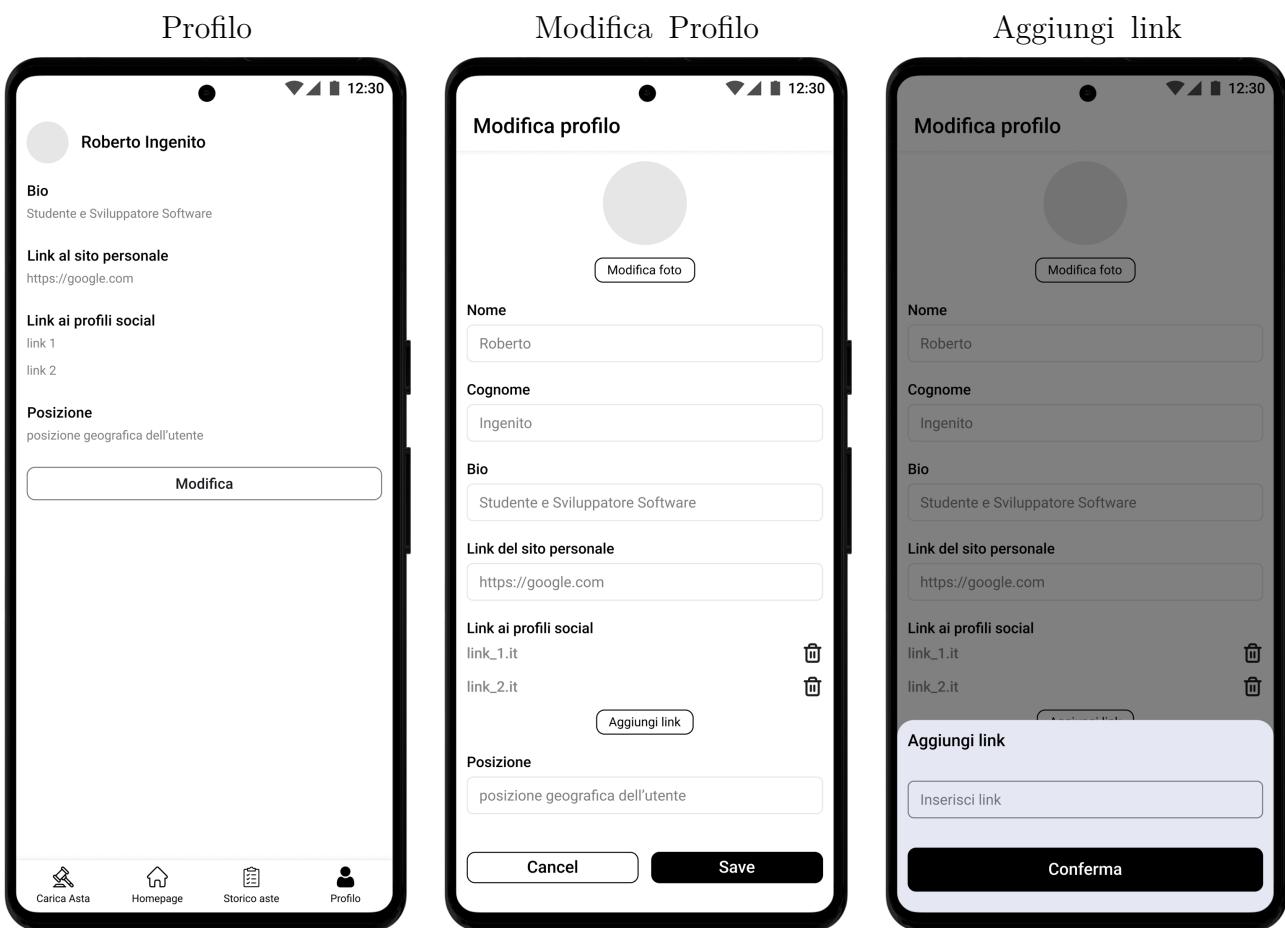


2.6.5 Profilo

Il mockup "Profilo" mostra le informazioni personali dell'utente loggato.

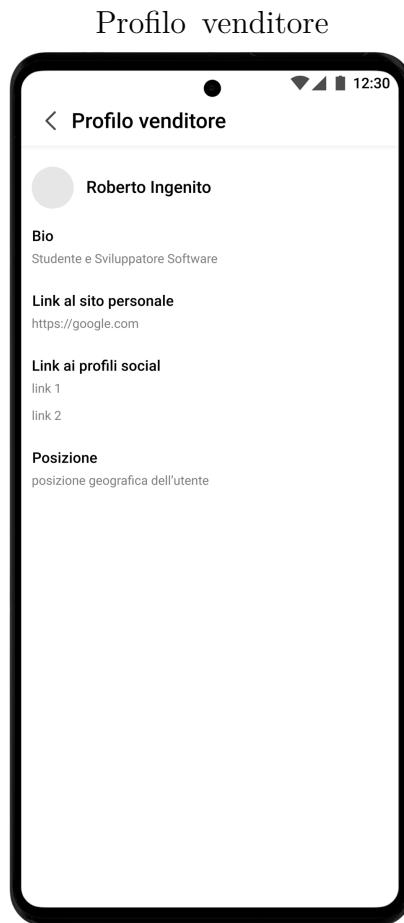
Premendo il pulsante "Modifica" verrà mostrato il mockup "Modifica profilo" dove l'utente può modificare nome, cognome, bio, link al proprio sito e link ai profili social. Premendo su "Aggiungi link" verrà mostrato il terzo mockup dove inserirà un nuovo link social.

Infine premerà il tasto "Salva" per salvare le modifiche effettuate.



2.6.6 Profilo Venditore

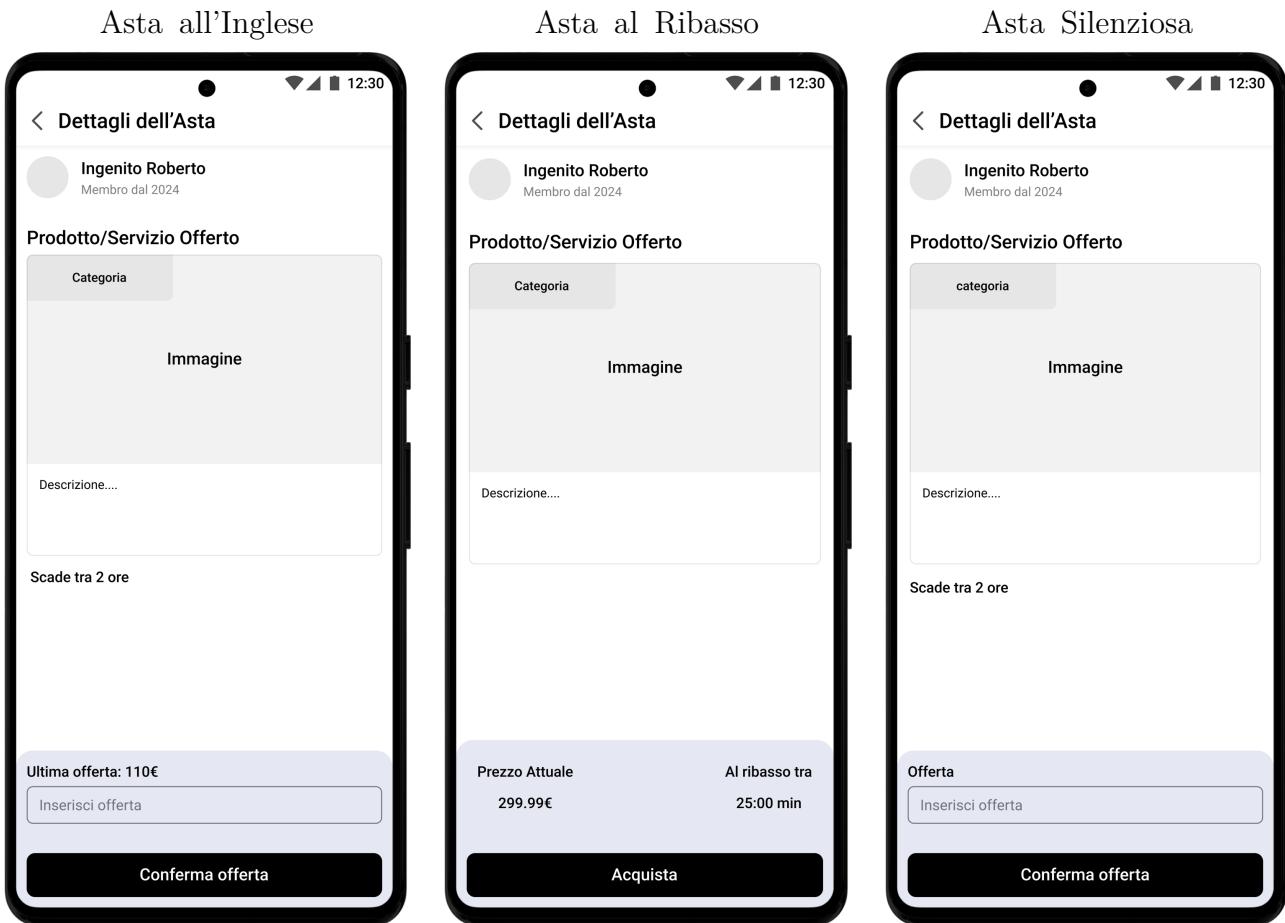
Il mockup "Profilo Venditore" mostra le informazioni personali del venditore selezionato.



2.6.7 Piazzare un'offerta

Dalla pagina Home, premendo su un'asta, l'applicazione navigherà a una di queste tre pagine a seconda del tipo di asta.

In base al tipo di asta, l'utente inserirà l'importo della sua offerta oppure l'acquisterà direttamente (nel caso dell'asta al ribasso)



2.6.8 Creare un'asta

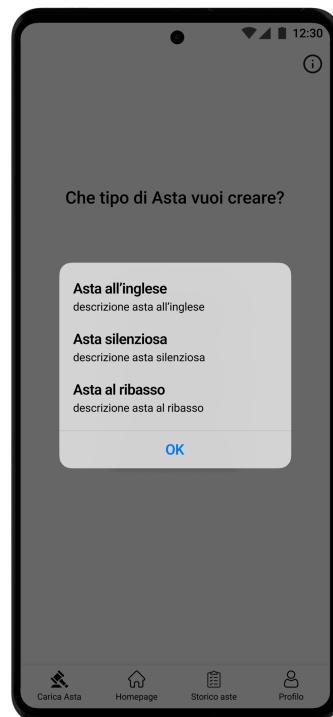
Il primo mockup mostra la pagina dov'è possibile selezionare il tipo di asta che si vuole creare. Premendo sul pulsante "i" in alto a destra, apparirà la finestra mostrata nel secondo mockup dove vengono spiegate le differenze fra i tre tipi d'asta.

I tre mockup in basso, mostrano le pagine dei diversi tipi di creazione di un'asta.

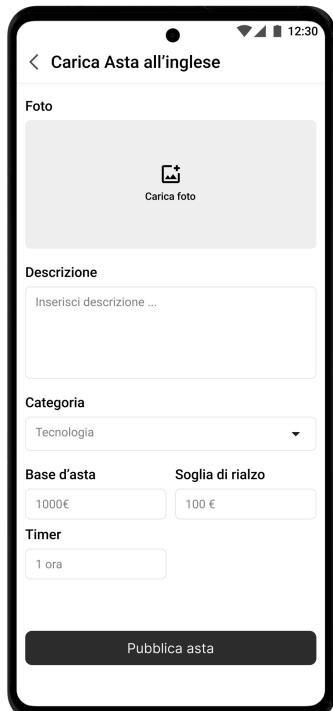
Selezione tipo di asta



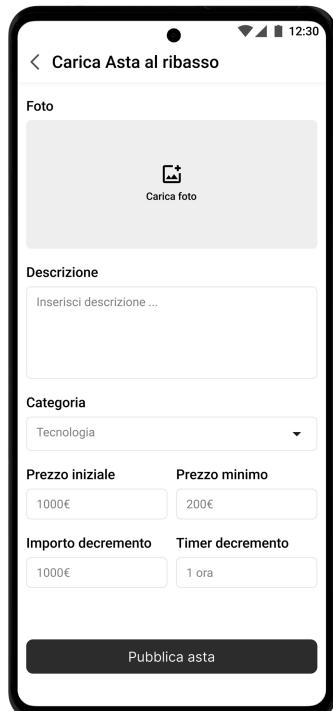
Informazioni tipi d'asta



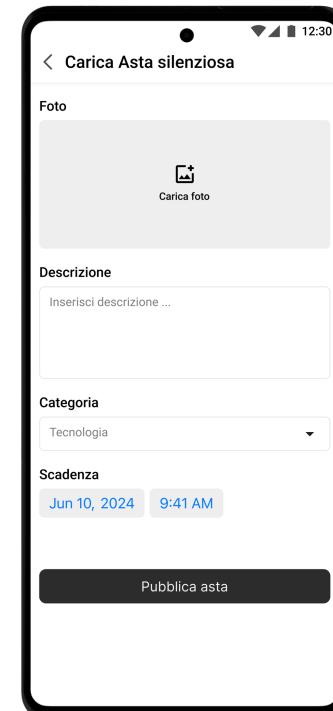
Creazione asta all'Inglese



Creazione asta al Ribasso



Creazione asta Silenziosa



2.7 Valutazione dell'usabilità a priori

La valutazione dell'usabilità a priori è un importante fase nella progettazione di interfacce utente. Questo processo permette di identificare potenziali problemi di usabilità prima che il prodotto venga rilasciato.

Uno dei metodi che abbiamo utilizzato è stato l'applicazione delle euristiche di Nielsen.

Le euristiche di Nielsen sono un insieme di dieci principi generali per la progettazione dell'interfaccia utente. Questi principi servono come linee guida, non sono regole rigide ma piuttosto suggerimenti che possono aiutare a valutare, identificare e migliorare problemi comuni di usabilità.

2.7.1 Euristiche di Nielsen

1. Visibilità dello stato del sistema

L'applicazione dovrebbe sempre tenere gli utenti informati su cosa sta succedendo, fornendo dei feedback immediati a seguito di un'azione.

All'interno dell'applicazione questa euristica viene soddisfatta implementando alert e messaggi di conferma.

2. Corrispondenza tra il sistema e il mondo reale

L'applicazione dovrebbe utilizzare un linguaggio semplice, con termini che gli utenti comprendono facilmente, evitando i termini tecnici. Ad esempio, i termini usati per descrivere le azioni come "Conferma offerta", "Modifica profilo"; oppure i messaggi di conferma come "Offerta confermata"/"Offerta rifiutata"

3. Controllo dell'utente e libertà

Gli utenti spesso commettono errori e dovrebbero essere in grado di annullare facilmente le azioni indesiderate.

Dovremmo implementare dei dialog di conferma quando l'utente preme determinati button.

4. Coerenza e standard

L'interfaccia dell'applicazione dovrebbe essere coerente in tutte le sue sezioni, sia in termini di design che di funzionalità. Ad esempio i pulsanti, i colori e le azioni devono avere lo stesso significato in tutte le schermate.

Di conseguenza, l'interfaccia è stata creata utilizzando componenti di Material Design, libreria standard di Google che fornisce componenti conformi agli standard e per cui la maggior parte degli utenti ha già una certa familiarità.

5. Prevenzione degli errori

L'app dovrebbe prevenire gli errori rendendo difficile per gli utenti fare scelte sbagliate. È meglio progettare l'interfaccia in modo da evitare che gli errori accadano, piuttosto che fornire messaggi di errore.

Anche in questo caso, l'implementazione di Dialog previene i possibili errori che gli utenti possono commettere.

6. Riconoscimento piuttosto che ricordo

Ridurre al minimo la quantità di informazioni che l'utente deve ricordare durante la navigazione. Ad esempio, fornendo suggerimenti durante l'uso dell'app per agevolare l'interazione dell'utente.

Per quanto riguarda questa euristica, abbiamo creato, nella pagina di creazione di un'asta, un "info button" che spiega brevemente le differenze tra le tipologie di asta.

7. Flessibilità ed Efficienza

L'app dovrebbe essere utilizzabile sia da utenti principianti che esperti.

L'apprendimento degli utenti è molto semplificato dall'uso di componenti UI standard, e da un interfaccia minimal e chiara.

8. Design Estetico e Minimalista

Le informazioni e le opzioni inutili dovrebbero essere eliminate. Ogni elemento dell'interfaccia dovrebbe avere uno scopo chiaro.

Di fatto, ci colleghiamo ai punti precedenti.

9. Aiuto e Documentazione

L'applicazione dovrebbe essere facile da usare senza documentazione, dovrebbe esserci comunque un aiuto disponibile per gli utenti che lo richiedono.

Come abbiamo già affermato precedentemente, l'utilizzo di componenti UI standard rende molto più facile agli utenti navigare attraverso l'interfaccia. Inoltre, le informazioni poco chiare, come possono essere la tipologia di asta che un utente può creare, è accompagnata da un "info button" facilmente individuabile, che fornisce informazioni chiare e precise.

10. Gestione degli Errori

Se si verificano errori, il sistema dovrebbe presentare messaggi d'errore chiari e non codici d'errore.

Quest'euristica viene rispettata andando a creare degli alert con messaggi d'errore chiari all'interno dell'interfaccia.

2.7.2 Glossario

UML (Unified Modeling Language) Un linguaggio di modellazione grafica standard utilizzato per visualizzare, specificare, costruire e documentare gli artefatti di un sistema software. Include vari tipi di diagrammi, come diagrammi delle classi, dei casi d'uso e delle sequenze.

Mockup Rappresentazione statica dell'interfaccia utente di un'applicazione, utilizzata per mostrare il layout visivo e simulare l'aspetto dell'interfaccia prima della realizzazione finale.

Euristiche di Nielsen Dieci principi di usabilità definiti da Jakob Nielsen per guidare la progettazione di interfacce utente, che includono visibilità dello stato del sistema, corrispondenza tra il sistema e il mondo reale, controllo dell'utente, prevenzione degli errori e altri criteri.

Casi d'uso Tecnica di analisi funzionale che descrive, in termini di attori e scenari, come il sistema interagisce con gli utenti per soddisfare determinati obiettivi. Ogni caso d'uso documenta le interazioni tra l'attore e il sistema.

Tabelle di Cockburn Modello di rappresentazione dei casi d'uso proposto da Alistair Cockburn. Ogni tabella contiene informazioni come titolo, attore principale, precondizioni, flusso principale di eventi, estensioni e risultati attesi, rendendo i casi d'uso più strutturati e comprensibili.

Attore Entità esterna al sistema che interagisce con esso. Può essere un utente umano o un altro sistema che richiede servizi al software.

Precondizione Condizione che deve essere vera prima dell'esecuzione di un caso d'uso affinché il sistema possa funzionare correttamente.

Flusso principale Sequenza di azioni standard attraverso le quali il sistema realizza il caso d'uso in condizioni ideali, senza errori o eccezioni.

Mockup ad alta fedeltà Versione dettagliata del mockup che include quasi tutte le funzionalità grafiche e interattive previste nel prodotto finale.

Specifica dei requisiti

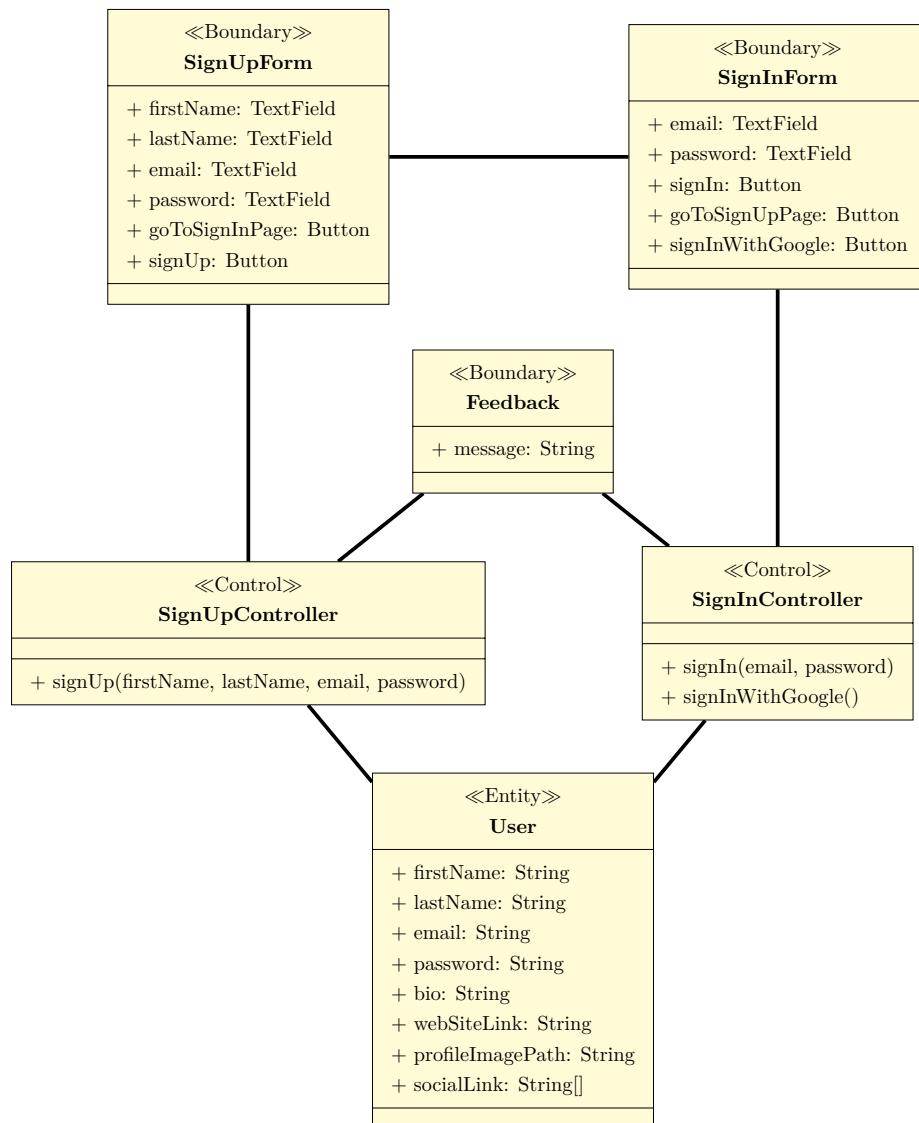
3.1 Modelli di dominio

Quando si sviluppa un progetto software, è importante avere una chiara comprensione del sistema che si sta costruendo. Un modello di dominio aiuta a definire gli elementi principali del sistema, come interagiscono tra loro e come vengono rappresentati nel software.

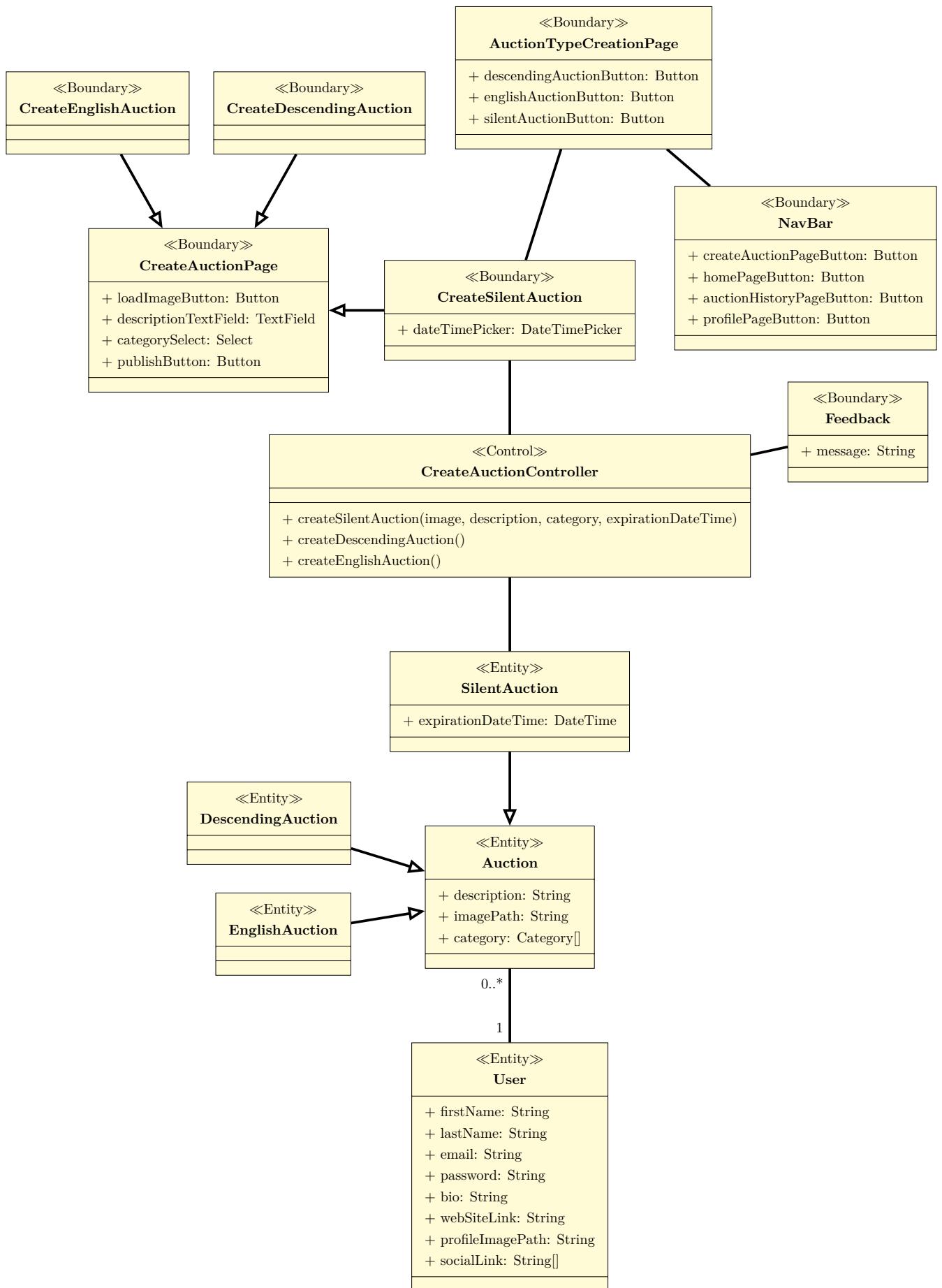
Per strutturare il nostro modello di dominio, abbiamo adottato l'approccio "Three Object Type", che suddivide gli oggetti del sistema in tre categorie principali:

- **Boundary:** Questi oggetti gestiscono l'interazione tra il sistema e gli attori esterni, come utenti o altri sistemi. Fungono da intermediari tra l'interfaccia utente e la logica interna dell'applicazione.
- **Control:** Questi oggetti implementano la logica di business e coordinano il flusso di dati tra gli oggetti Boundary e Entity. I Control mantengono separati gli aspetti di presentazione e quelli relativi ai dati, rendendo il sistema più facile da mantenere e migliorare.
- **Entity:** Questi oggetti sono utilizzati per modellare gli oggetti del mondo reale o concettuale che il sistema deve gestire e che devono essere persistenti, cioè mantenere il loro stato nel tempo.

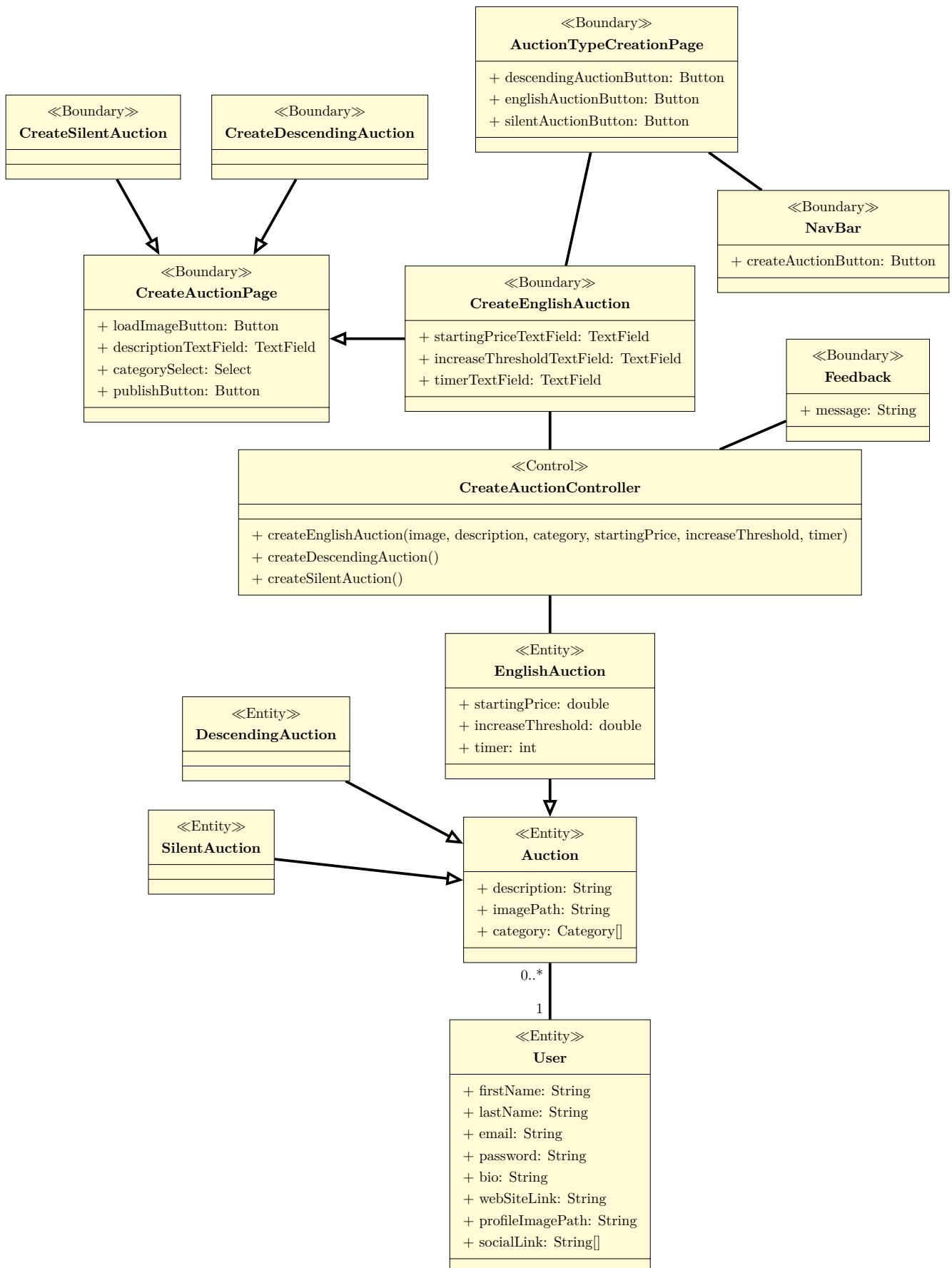
Registrazione e accesso



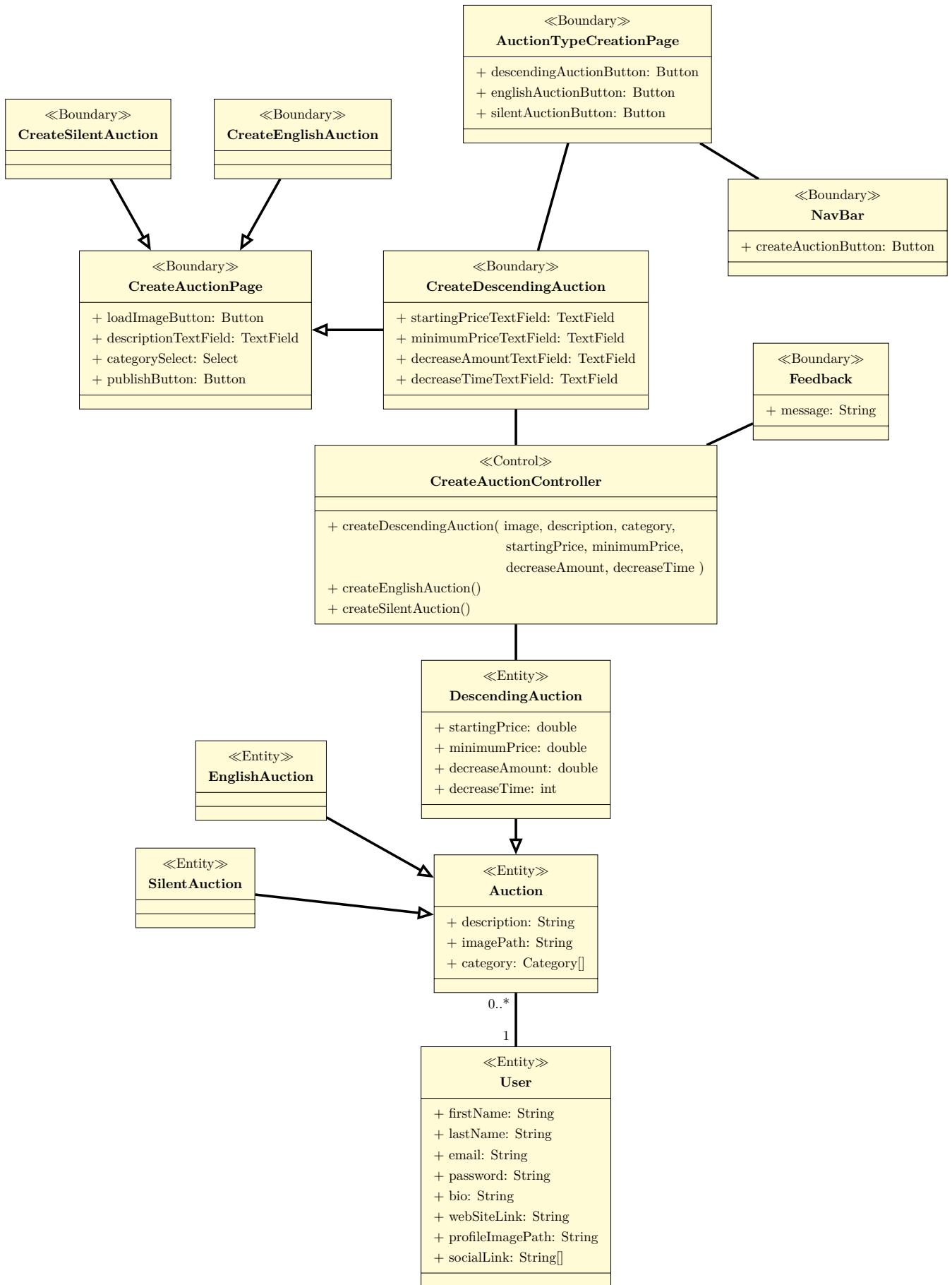
Creazione asta silenziosa



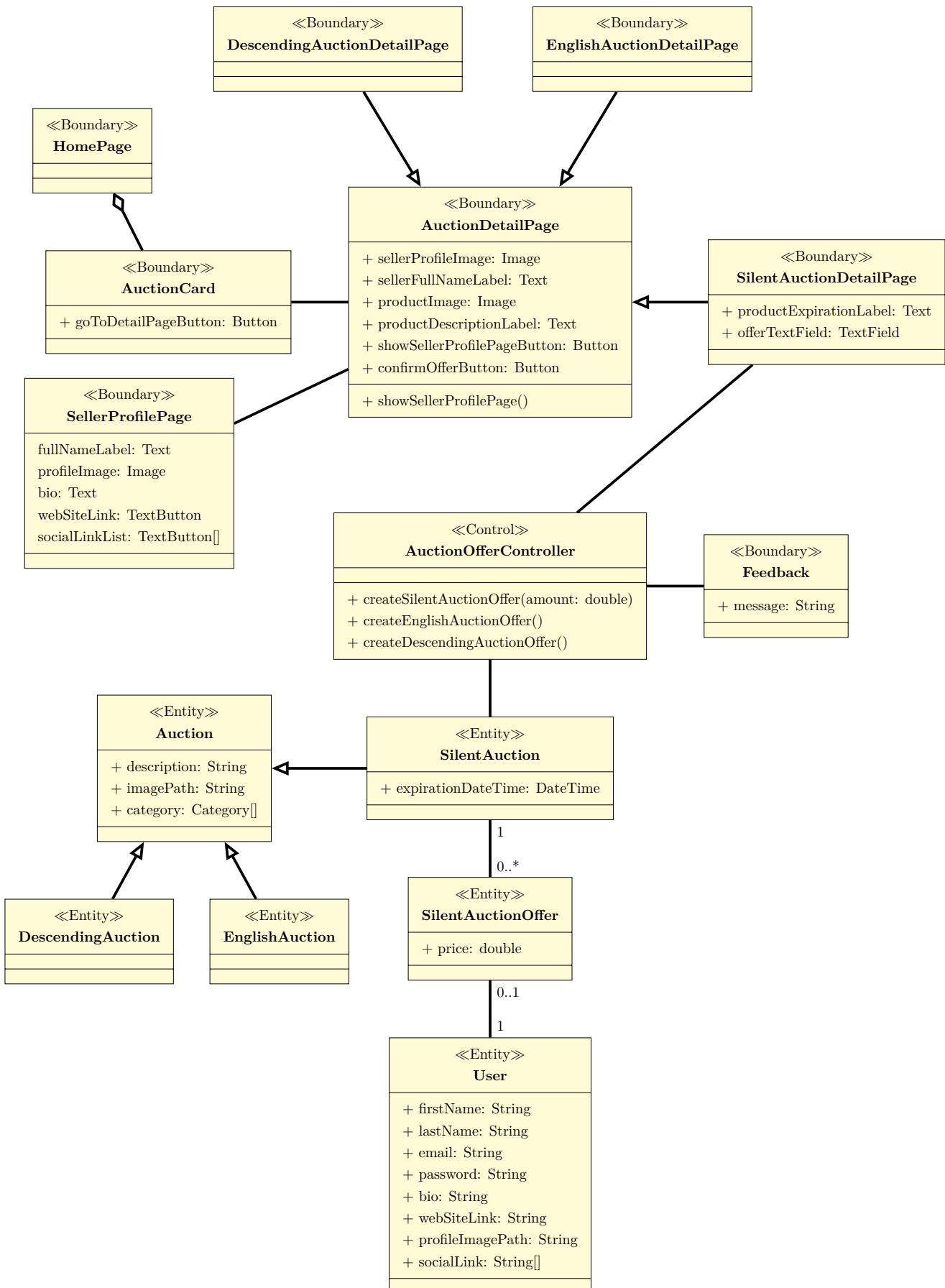
Creazione asta all'inglese



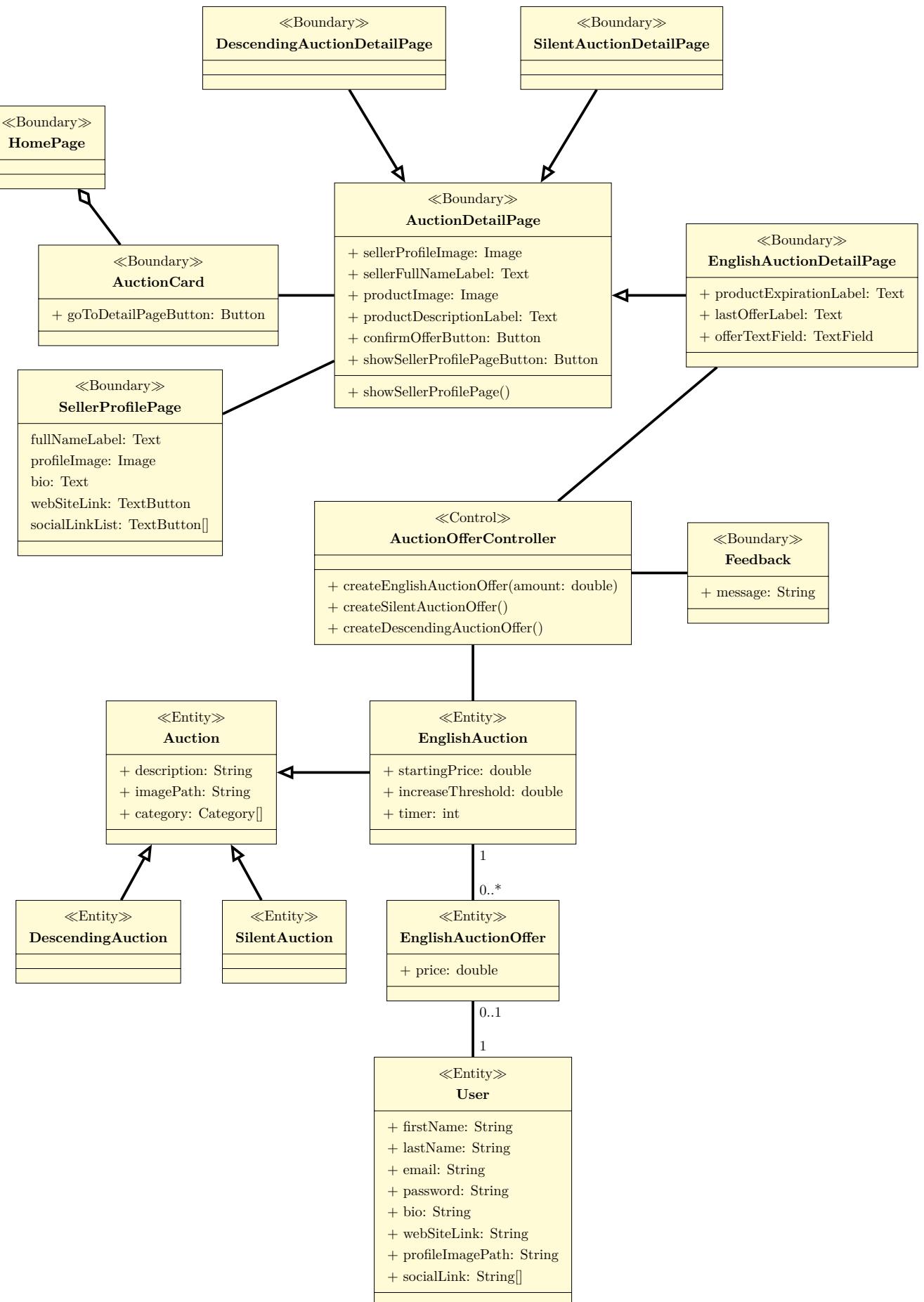
Creazione asta al ribasso



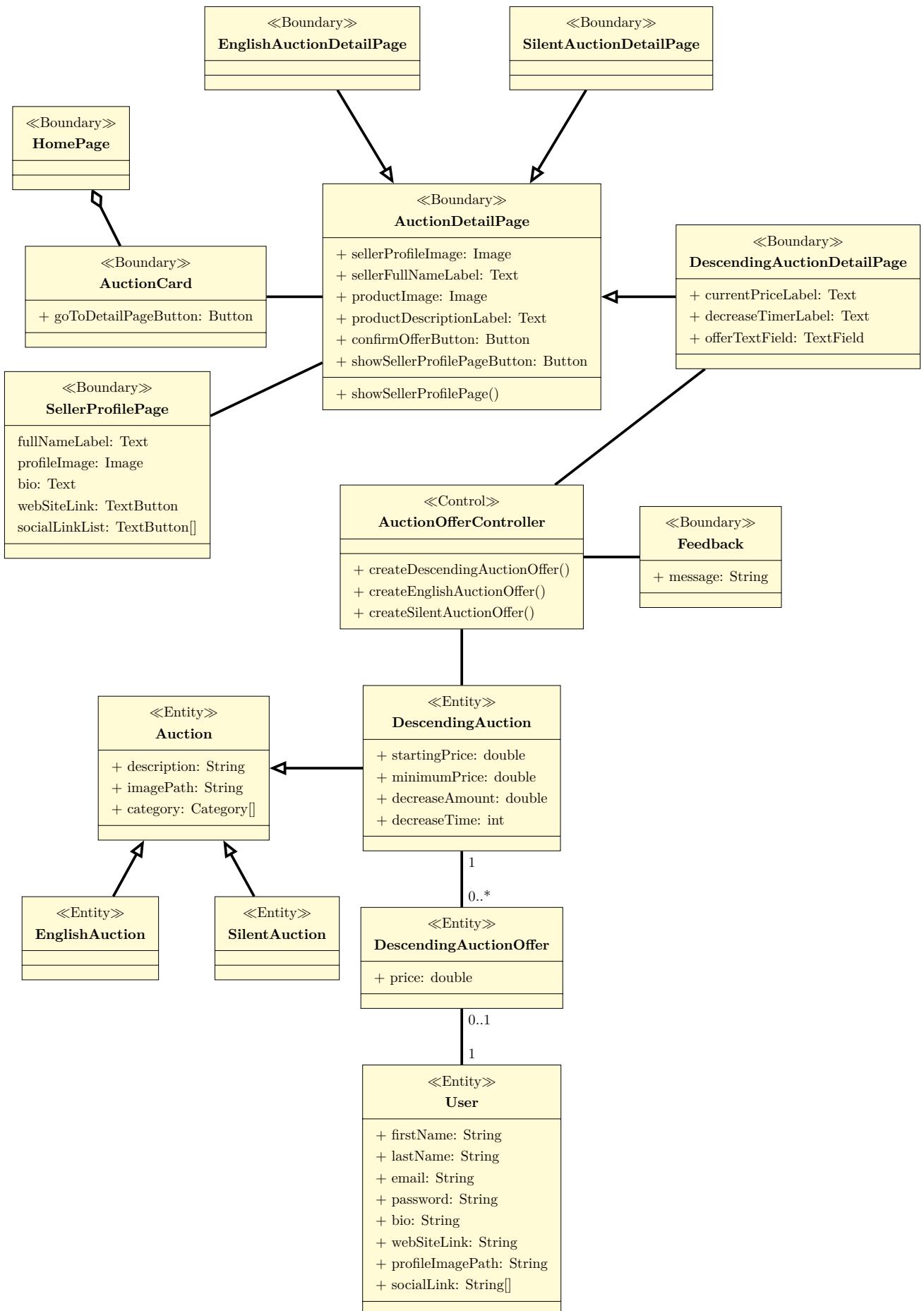
Puntare su un'asta silenziosa



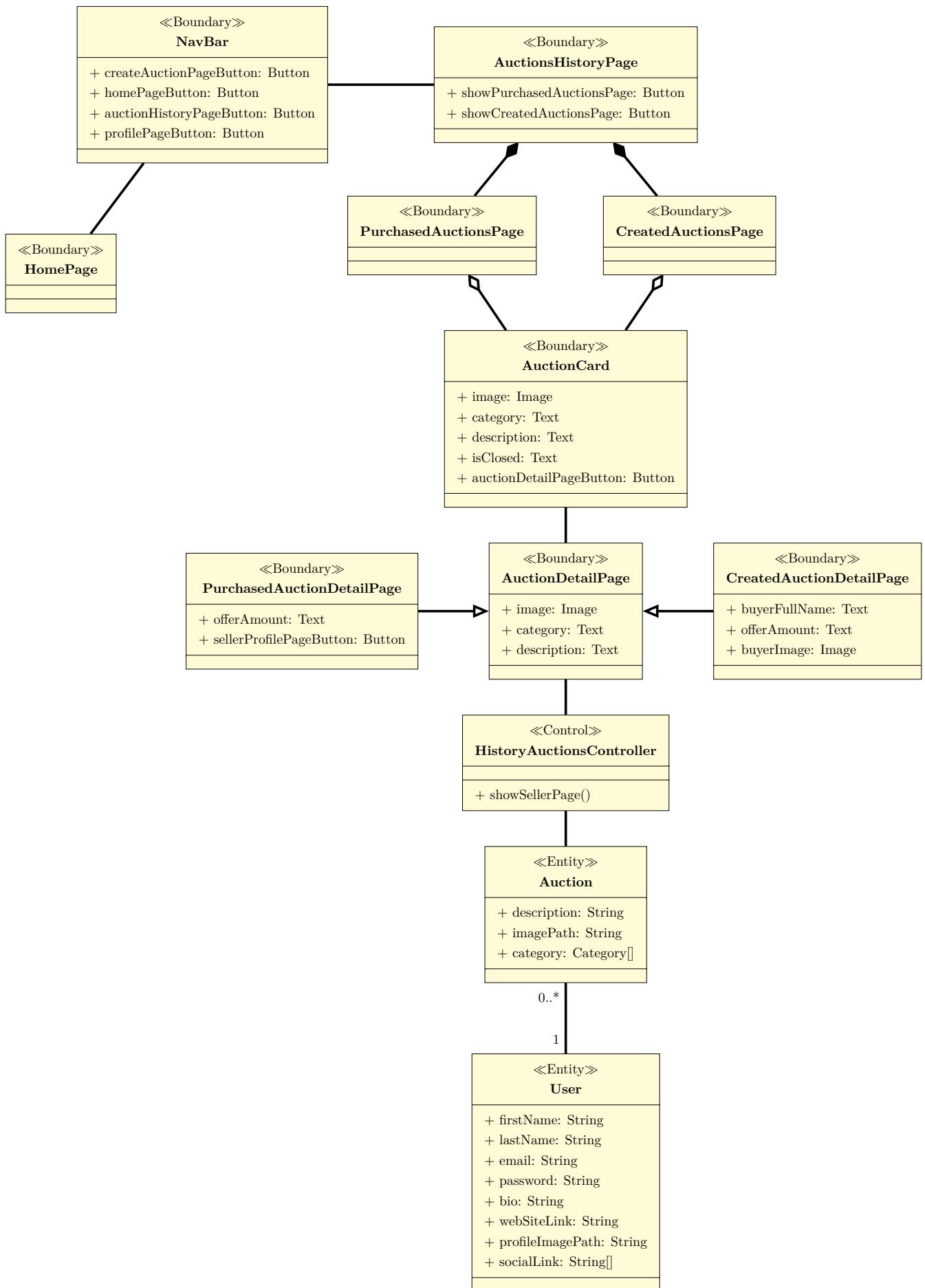
Puntare su un'asta all'inglese



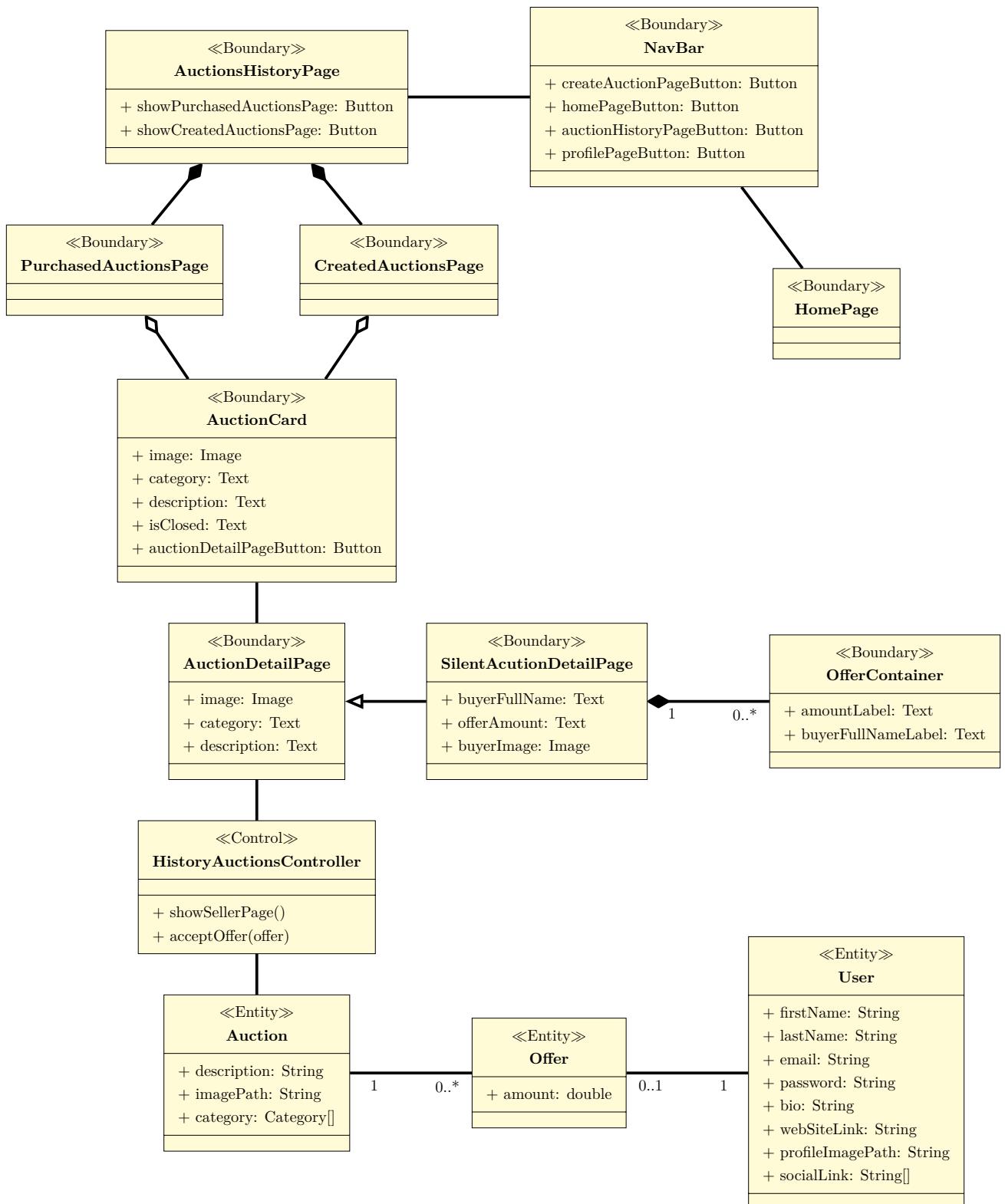
Puntare su un'asta al ribasso



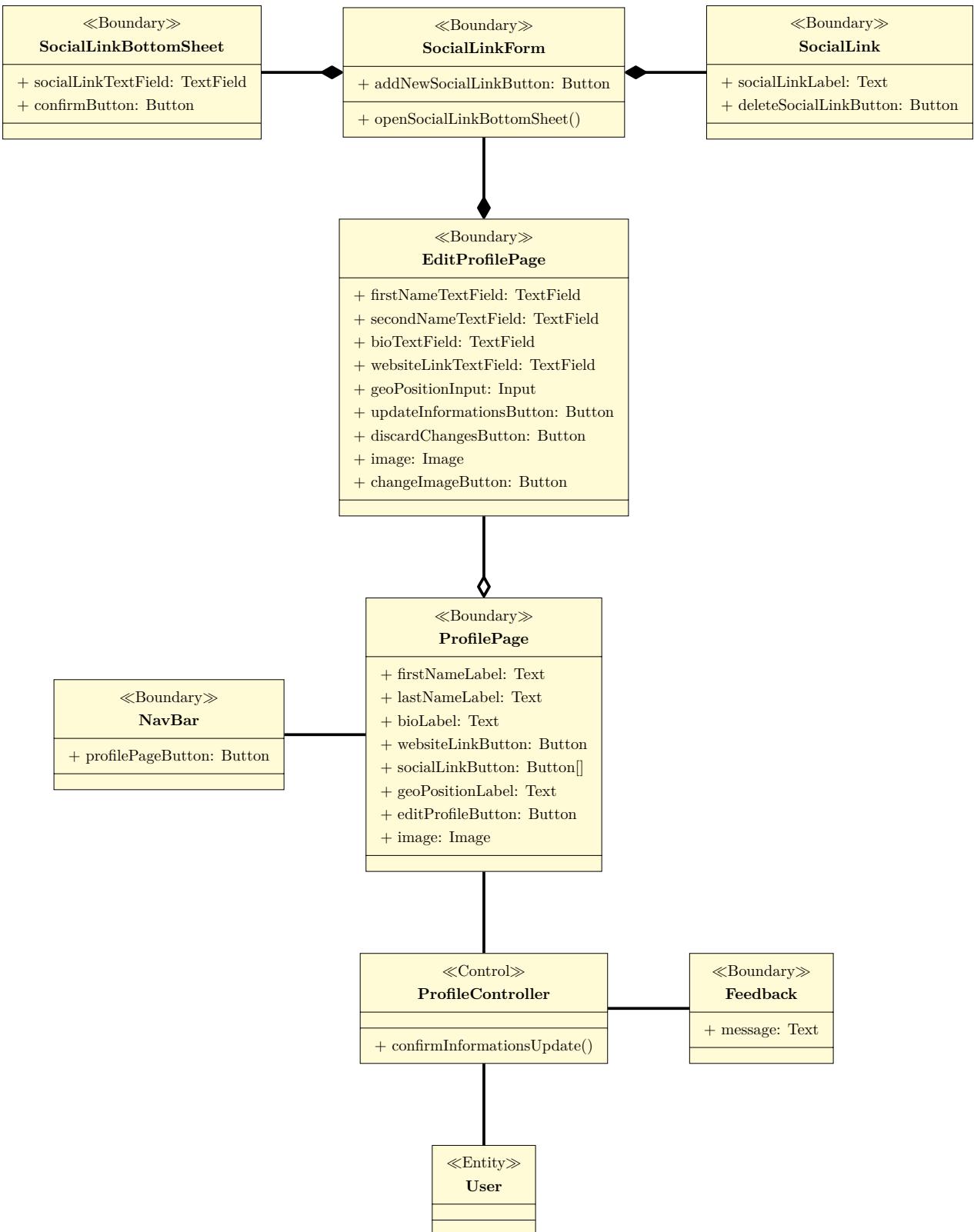
Storico aste (terminate)



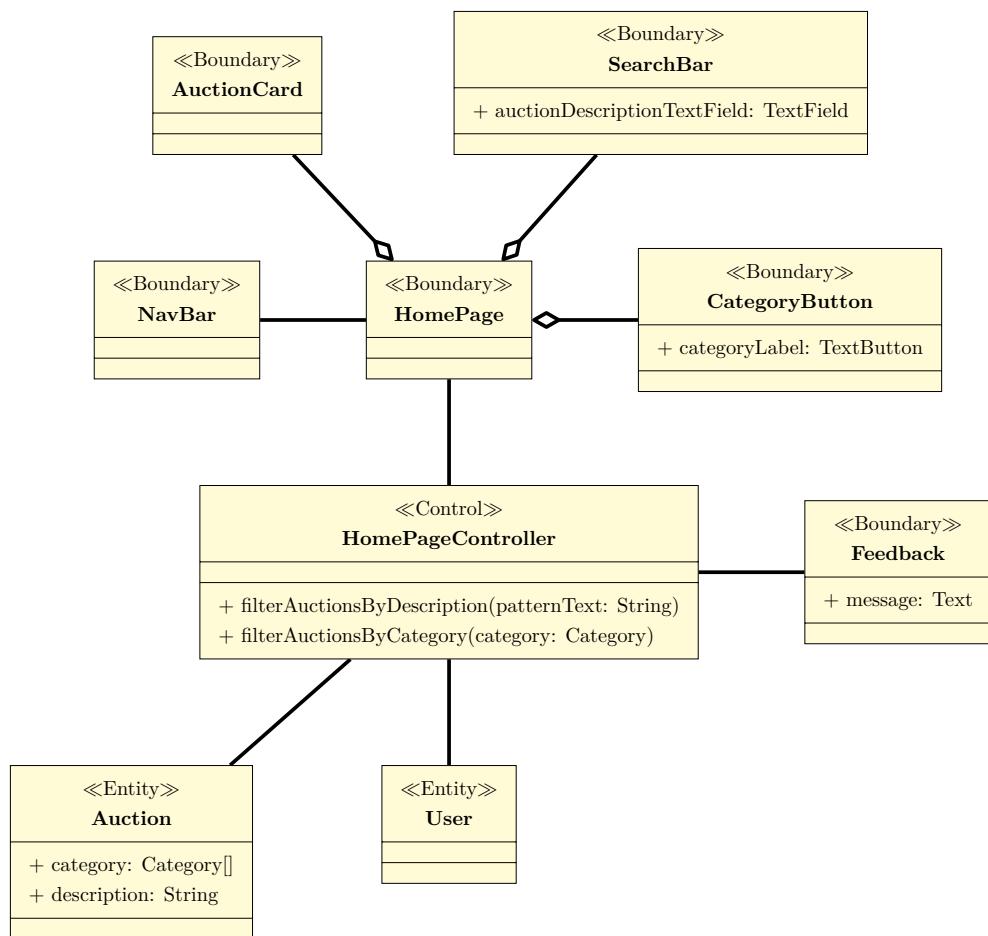
Asta silenziosa: accettazione di un'offerta



Modifica profilo



Filtraggio e ricerca di un'asta



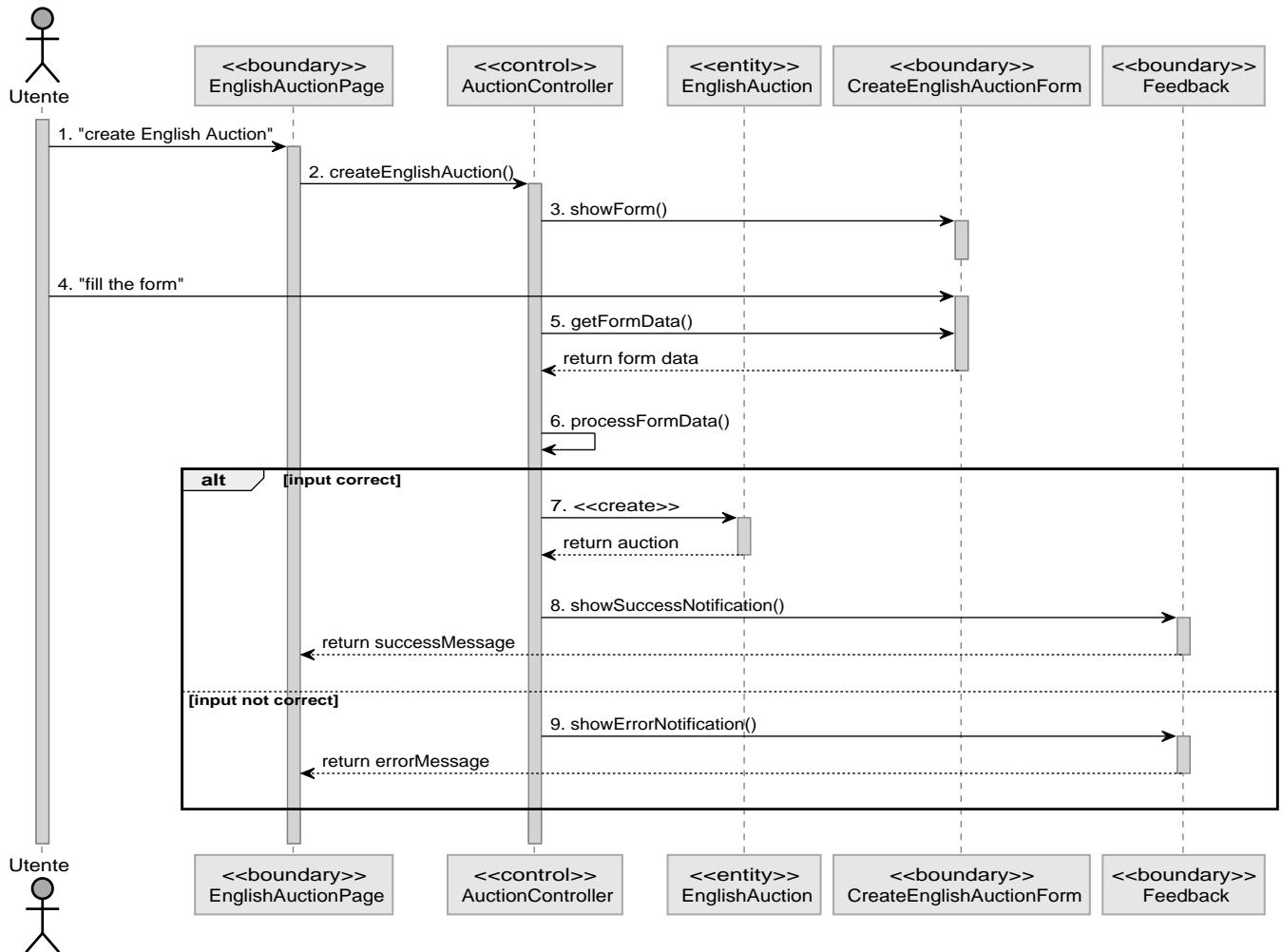
3.2 Sequence diagram di due casi d'uso

3.2.1 Creazione asta all'inglese

Questo Sequence Diagram descrive il processo di creazione di un'asta all'inglese.

L'attore principale è l'utente venditore che inizia il processo di creazione dell'asta inserendo i dati all'interno del form e cliccando il pulsante di creazione dell'asta.

A questo punto il sistema si occupa di gestire tutta la logica di creazione e di restituire un messaggio all'utente.

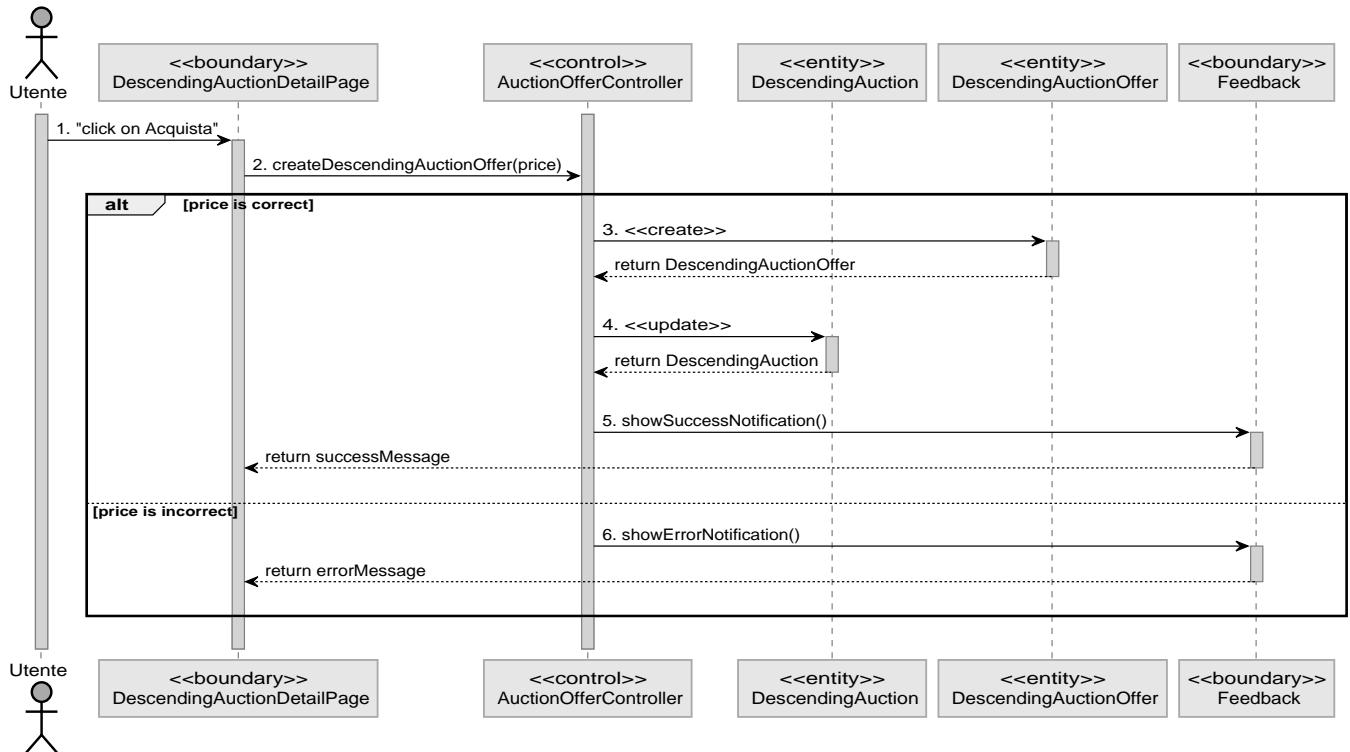


3.2.2 Piazzare un'offerta su un'asta al ribasso

Questo Sequence diagram descrive il processo di creazione di un'offerta per un'asta al ribasso.

L'attore principale è l'utente acquirente che inizia il processo di creazione dell'offerta inserendo i dati all'interno del form e cliccando il pulsante di conferma.

Il sistema, dopo aver elaborato i dati, restituisce un messaggio all'utente.



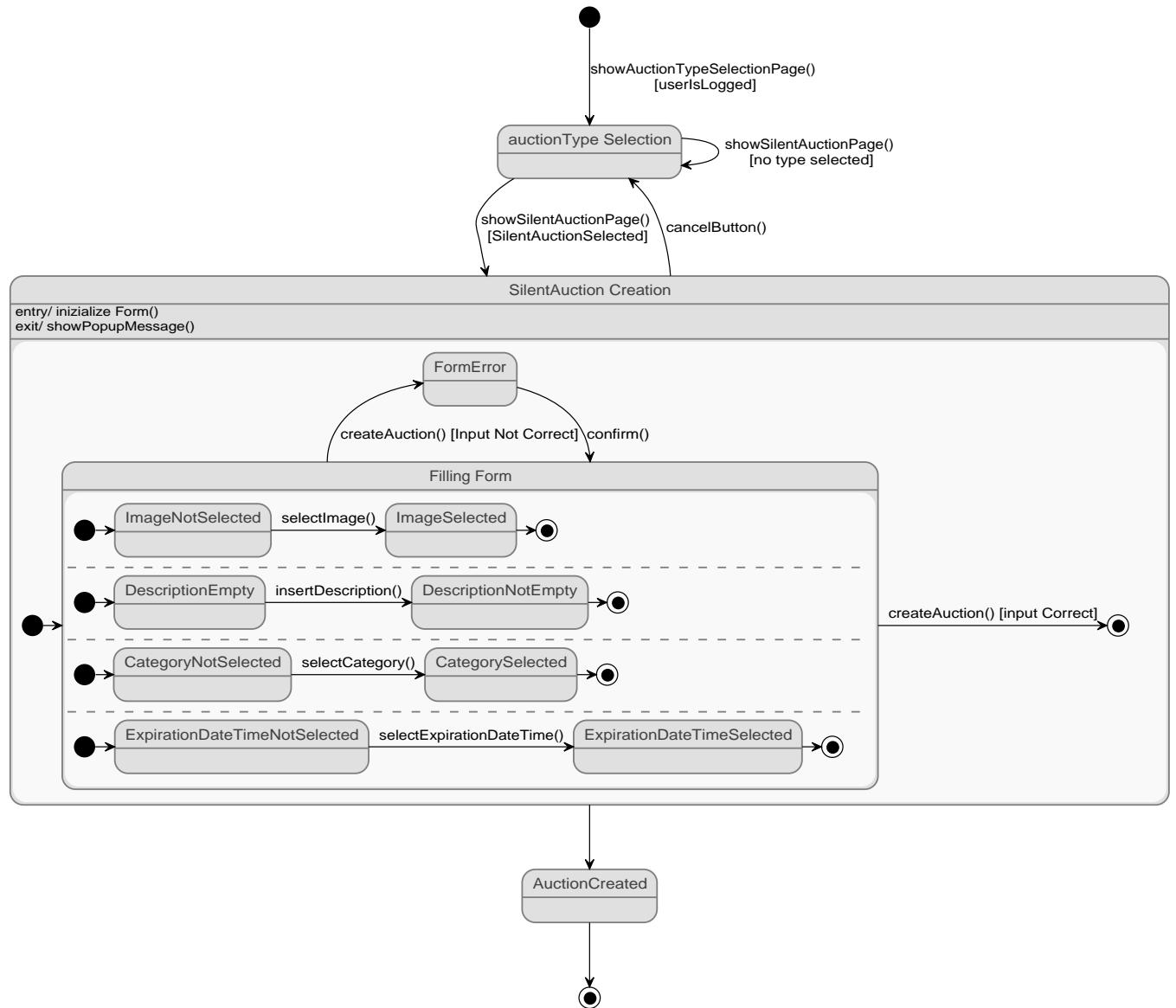
3.3 Progettazione degli Event-Based Statecharts

Gli statecharts (o statechart diagrams) sono un tipo di diagramma utilizzato per rappresentare il comportamento dinamico di un sistema attraverso i suoi stati e le transizioni tra di essi.

3.3.1 Creazione asta silenziosa

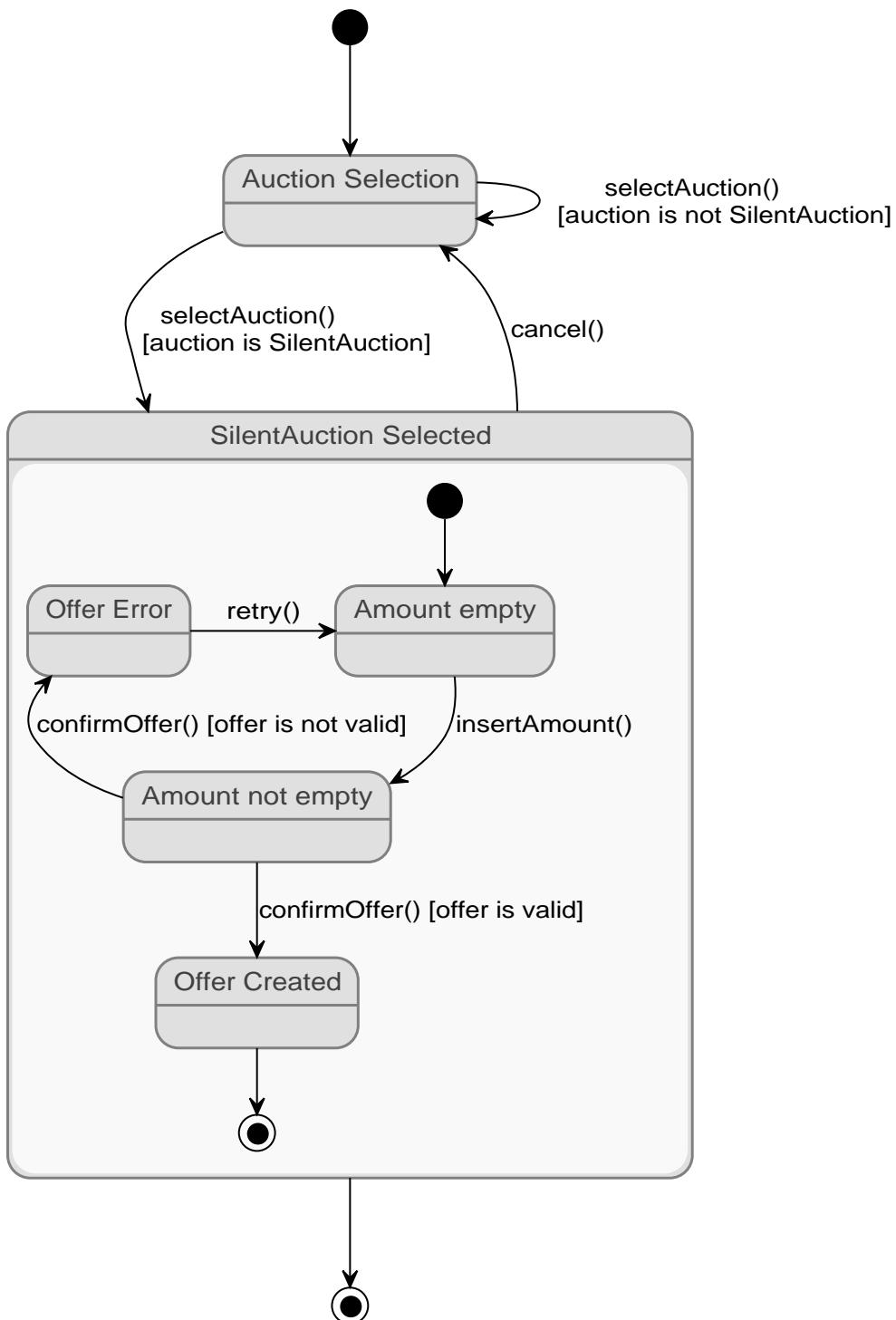
Il seguente statechart descrive il comportamento del sistema durante la creazione di un'asta silenziosa.

Per prima cosa, l'utente si trova all'interno della schermata di selezione della tipologia d'asta da creare. Dopo aver selezionato la tipologia d'asta corretta, il sistema passa alla pagina di creazione dell'asta, la quale contiene il form in cui inserire tutte le informazioni necessarie. Una volta che l'utente ha selezionato e inserito tutte le informazioni correttamente, l'asta viene creata e il sistema si aggiorna.



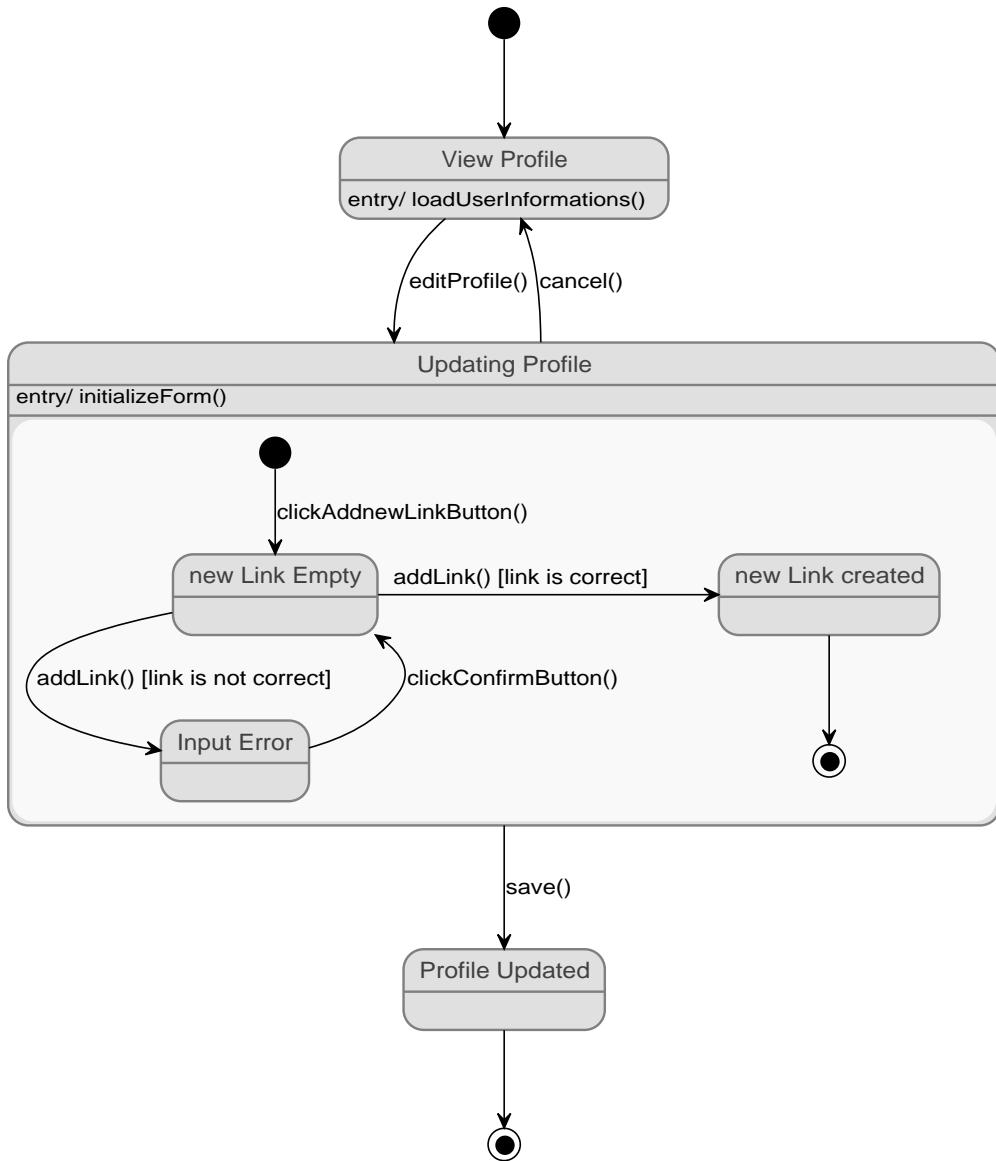
3.3.2 Offerta asta silenziosa

Questo statechart descrive il comportamento del sistema durante la creazione di un'offerta per il tipo d'asta silenziosa. Per prima cosa, l'utente seleziona l'asta cui desidera piazzare un'offerta. Dopo aver effettuato la selezione il sistema passa alla schermata di dettaglio dell'asta, in cui è possibile inserire un'offerta. Se l'offerta è corretta, il sistema si aggiorna correttamente.



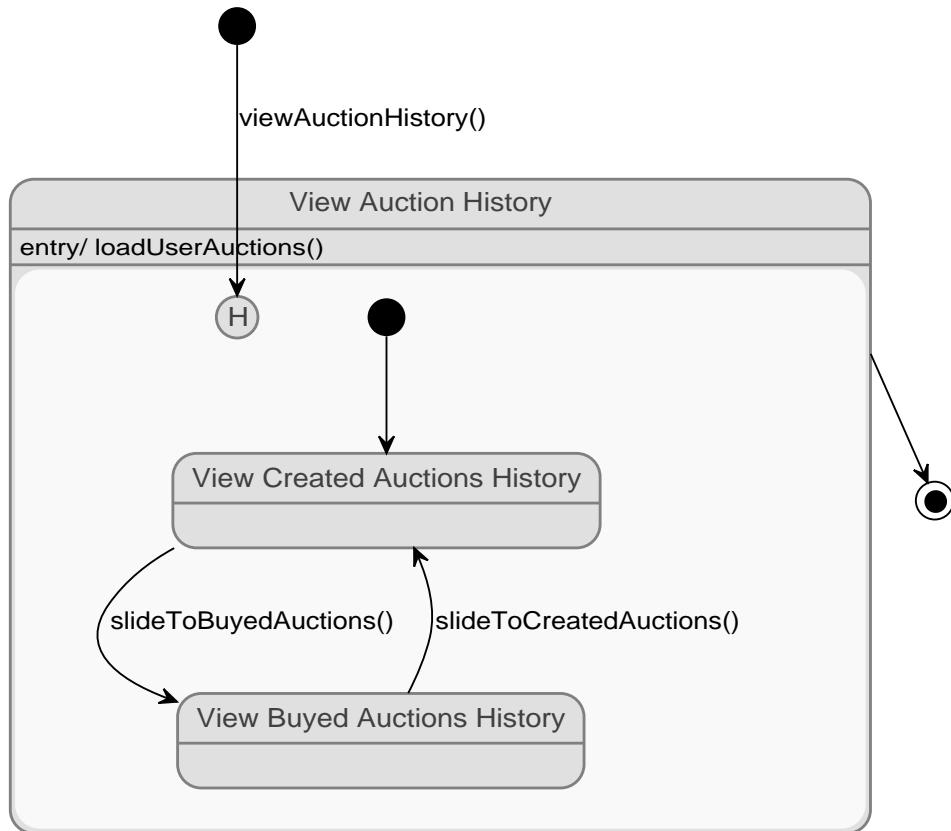
3.3.3 Aggiungi link social

Questo statechart descrive il comportamento del sistema durante l'aggiunta di un link social. Nello specifico, il sistema si trova in uno stato iniziale in cui è possibile visualizzare il profilo utente. Dopodiché, il sistema passa nello stato di aggiornamento del profilo, in cui è possibile aggiungere un nuovo link social. Dopo che il link è stato aggiunto, l'utente salva le modifiche e il sistema passa nello stato di profilo aggiornato.



3.3.4 Visualizzare aste create

Questo statechart descrive il comportamento del sistema quando l'utente vuole visualizzare lo storico delle aste create. In particolare, il sistema può trovarsi in due sottostati diversi: schermata dello storico delle aste create, e la schermata dello storico delle aste acquistate. Il sistema si ricorda dell'ultima schermata in cui si trovava precedentemente, che è di default la schermata delle aste create. Da ognuna di queste due schermate è possibile passare all'altra.



Documento di design

In questa sezione vengono illustrate le scelte architetturali e i design pattern adottati nel software. Un design pattern è una soluzione generale e collaudata per risolvere problemi ricorrenti che si incontrano durante la progettazione di software. Questi pattern possono migliorare la comprensione del codice, la manutenzione e la scalabilità di un'applicazione.

4.1 Architettura del Software

L'architettura del software è suddivisa tra il client e il server, con il client sviluppato in Flutter e il server utilizzando AWS.

4.1.1 Client

L'applicazione client è sviluppata utilizzando Flutter, un framework open-source di Google per la creazione di applicazioni mobili nativamente compilate. Flutter consente di creare un'interfaccia utente ricca e reattiva, mantenendo un'alta performance sia su Android che iOS.

Il client sarà responsabile della visualizzazione dei dati e dell'interazione con l'utente.

La gestione dello stato sarà gestito tramite il pattern MVC (Model-View-Controller) integrato con i DAO (Data Access Object). Questo approccio separa la logica di business, la gestione dello stato e l'accesso ai dati, migliorando la manutenibilità e la testabilità del codice.

4.1.2 Server

Per la gestione del server, abbiamo scelto Amazon Web Services (AWS), una piattaforma cloud che offre un'ampia gamma di servizi, tra cui elaborazione, archiviazione, e database. Il server è stato implementato utilizzando un'istanza EC2, con un database gestito tramite RDS e la logica applicativa eseguita su un backend sviluppato in Express.

4.1.3 Interazione tra Client e Server

Il client invia richieste HTTP al backend seguendo l'architettura REST. Il server riceve queste richieste, le elabora, e interagisce con il database per ottenere o aggiornare i dati.

Le risposte vengono restituite al client in formato JSON, garantendo così una facile integrazione e manipolazione dei dati all'interno dell'applicazione Flutter.

La comunicazione tra client e server avviene attraverso richieste asincrone, che permettono al client di ottenere dati aggiornati in tempo reale senza interrompere l'esperienza dell'utente.

4.2 Design pattern utilizzati

4.2.1 Model-View-Controller (MVVM)

Il pattern Model-View-ViewModel (MVVM) è stato scelto per strutturare l'applicazione in modo da separare le responsabilità e facilitare la manutenzione del codice.

- **Model:** Gestisce i dati dell'applicazione ed è responsabile della notifica alla View quando i dati cambiano. Questo permette alla View di aggiornarsi e riflettere le modifiche senza dover essere a conoscenza dei dettagli di come i dati sono gestiti.
- **View:** Si occupa della presentazione e dell'interfaccia utente. Mostra i dati agli utenti e aggiorna la visualizzazione in risposta alle modifiche del Model.
- **ViewModel:** Funziona da intermediario tra il Model e la View. Gestisce gli input degli utenti e aggiorna il Model e la View di conseguenza.

Vantaggi Principali

- **Separation of Concerns:** Il pattern MVVM promuove una chiara separazione tra la logica di business, la logica di presentazione e l'interfaccia utente, facilitando la manutenzione e l'espansione del codice.
- **Testabilità:** La logica dell'applicazione è centralizzata nel ViewModel, che può essere facilmente testato indipendentemente dalla View, rendendo più semplice lo sviluppo di test unitari.
- **Data Binding:** MVVM sfrutta il data binding bidirezionale, permettendo di aggiornare automaticamente l'interfaccia utente quando i dati nel ViewModel cambiano, migliorando l'esperienza utente e riducendo il codice boilerplate.

- **Modularità:** La separazione tra View e ViewModel consente di riutilizzare componenti e logica in diverse parti dell'applicazione senza duplicazione del codice.

4.3 Analisi delle scelte tecnologiche utilizzate

4.3.1 Client: Confronto con altre tecnologie

La decisione di utilizzare flutter è stata presa dopo un'analisi delle opzioni disponibili, considerando vari fattori come la produttività, le prestazioni e la compatibilità.

React Native

- **Prestazioni:** Flutter compila il codice direttamente in codice nativo, mentre React Native utilizza un ponte per comunicare tra JavaScript e codice nativo. Questo può portare a prestazioni superiori in Flutter.
- **Coerenza dell'UI:** Flutter utilizza un motore di rendering proprietario, che garantisce una coerenza visiva su tutte le piattaforme, mentre React Native si basa sui componenti nativi, il che può comportare variazioni tra iOS e Android.

Sviluppo nativo (Java/Kotlin per Android, Swift per iOS)

- **Cross-Platform:** Flutter consente di scrivere un solo codice per entrambe le piattaforme, riducendo i tempi e i costi di sviluppo rispetto alla scrittura di codice separato.
- **Consistenza dell'Interfaccia:** Flutter offre un controllo totale sul rendering e sulla consistenza dell'interfaccia su entrambe le piattaforme.
- **Aggiornamenti e Manutenzione:** La manutenzione di una singola codebase è generalmente più semplice e meno costosa.

4.3.2 Server

Amazon EC2

Amazon Elastic Compute Cloud (EC2) è il servizio di AWS che fornisce capacità di calcolo nel cloud. Tramite EC2, è possibile eseguire istanze di macchine virtuali configurabili in base alle esigenze del progetto, garantendo flessibilità e controllo totale sull'infrastruttura server.

Nel nostro caso, abbiamo implementato un server *NodeJS* che esegue un backend *express*.

Amazon RDS

Amazon Relational Database Service (RDS) è il servizio gestito di AWS per database relazionali, che semplifica le attività di configurazione, gestione e scalabilità dei database.

Nel nostro caso, abbiamo impiegato RDS con un database PostgreSQL.

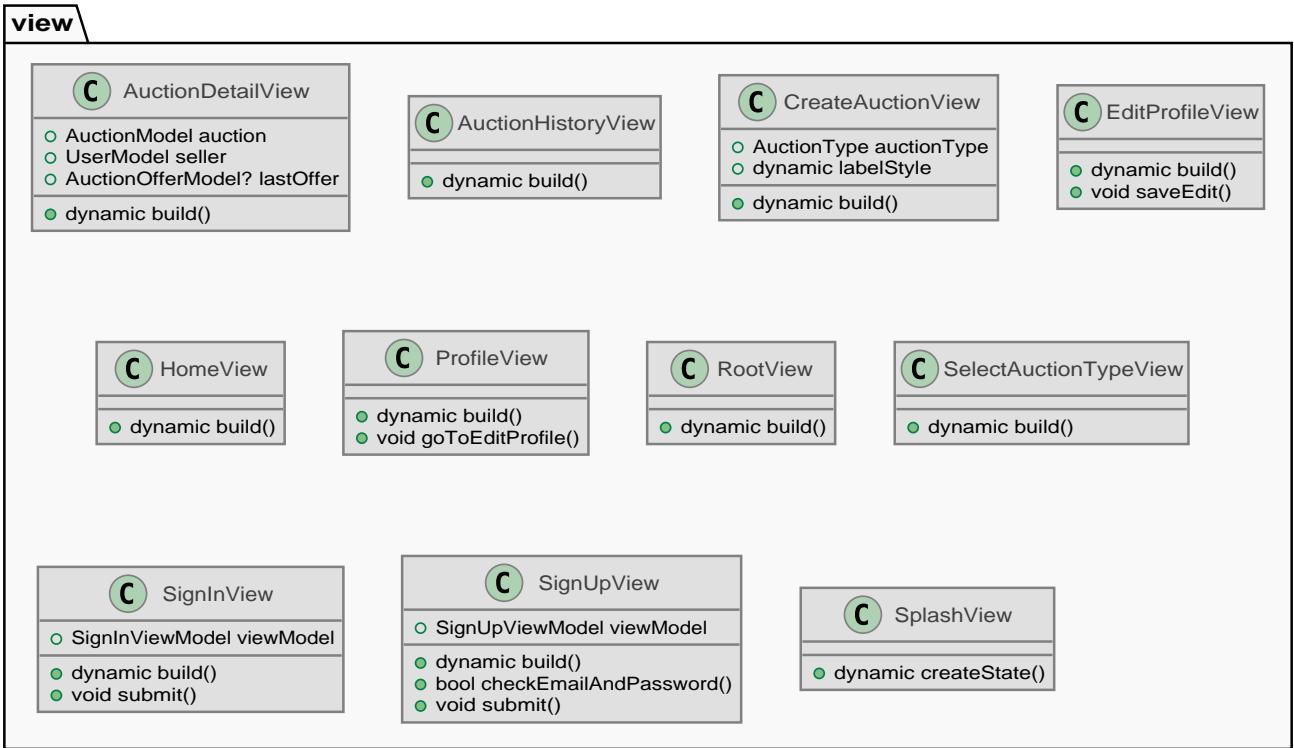
Express.js

Express.js è un framework per applicazioni web Node.js, utilizzato per sviluppare la logica applicativa del nostro backend. Ci ha permesso di implementare un'API RESTful in modo rapido ed efficiente, gestendo le richieste tra il client e il server e interfacciandosi con il database ospitato su RDS.

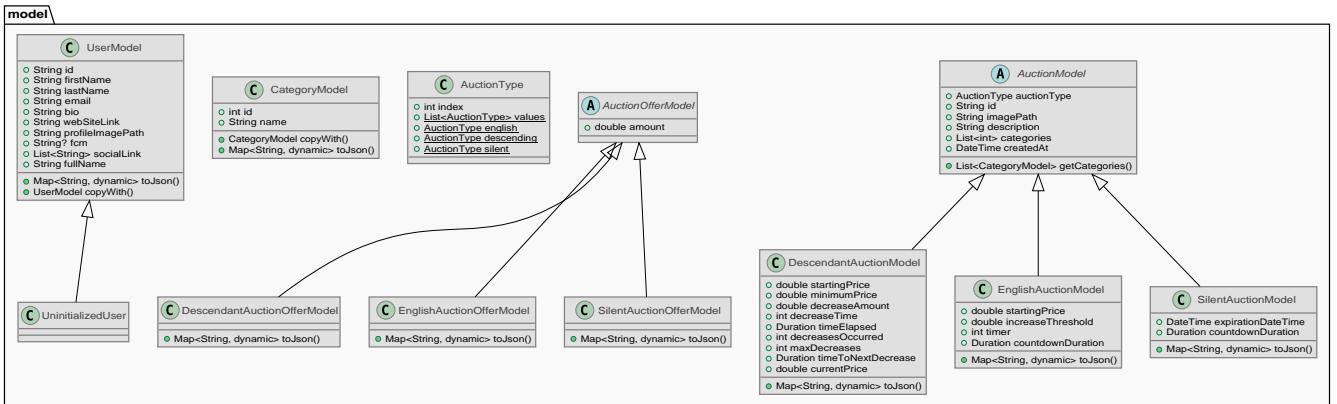
4.4 Diagrammi UML di Design

Di seguito illustriamo i diagrammi UML di Design del sistema.

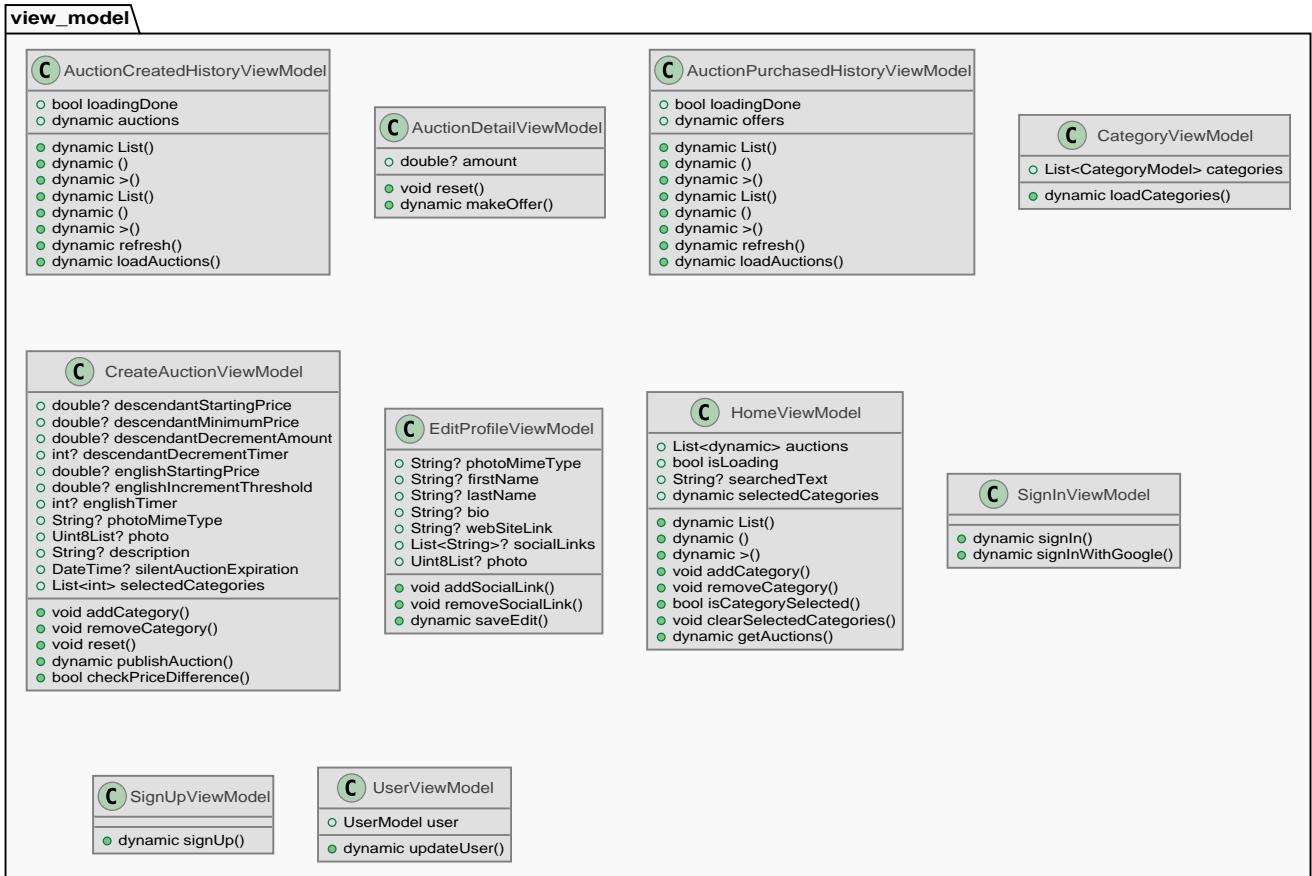
4.4.1 View



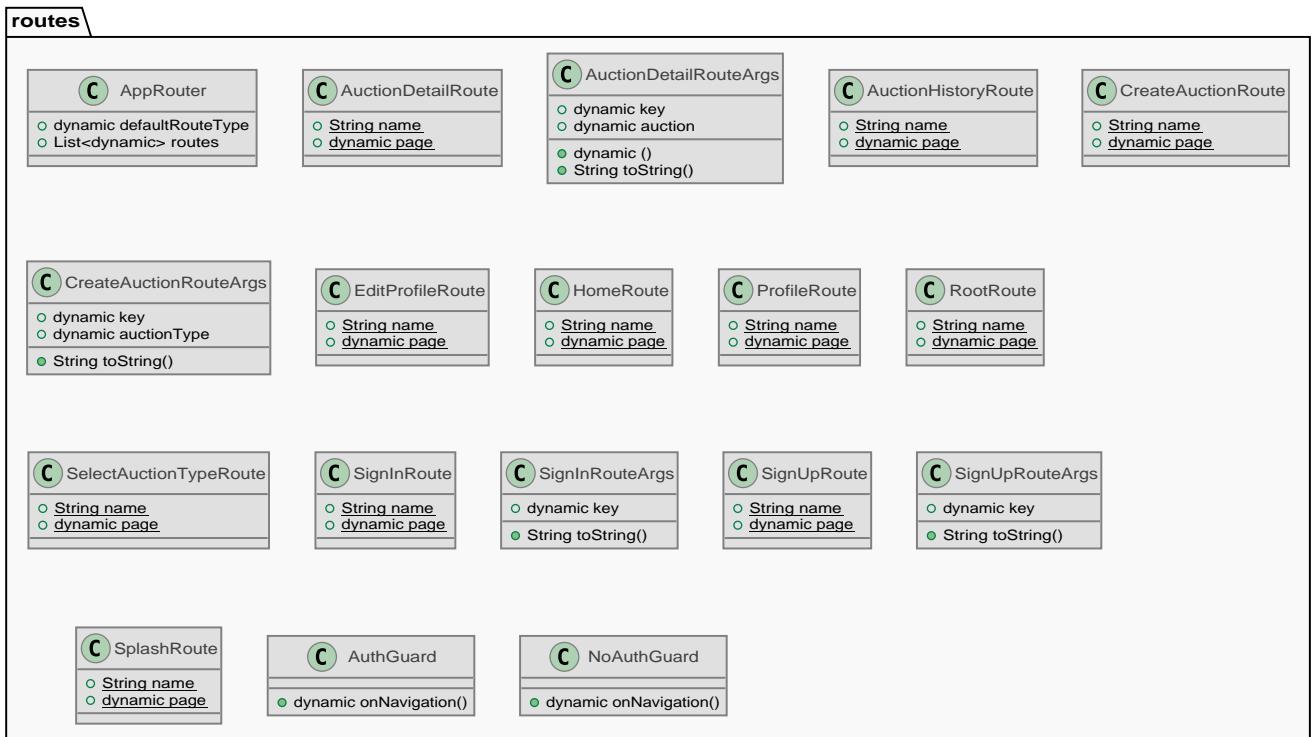
4.4.2 Model



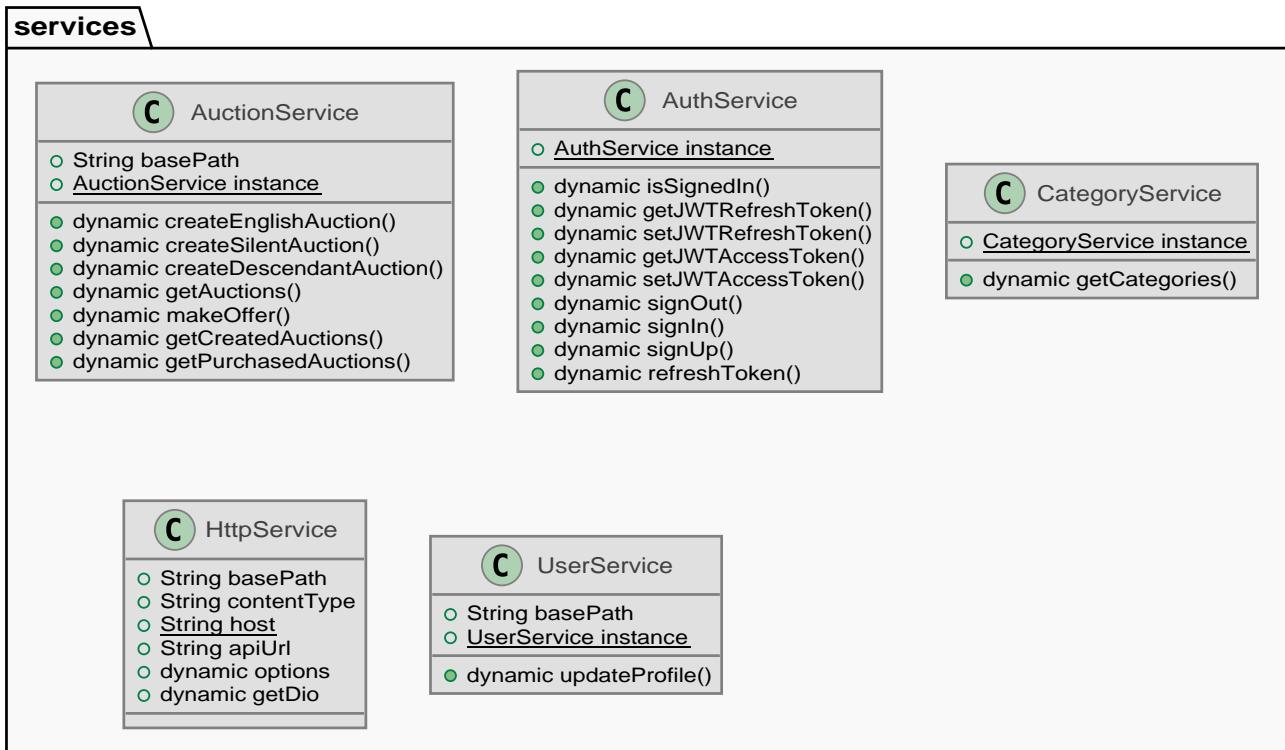
4.4.3 ViewModel



4.4.4 Routes



4.4.5 Services



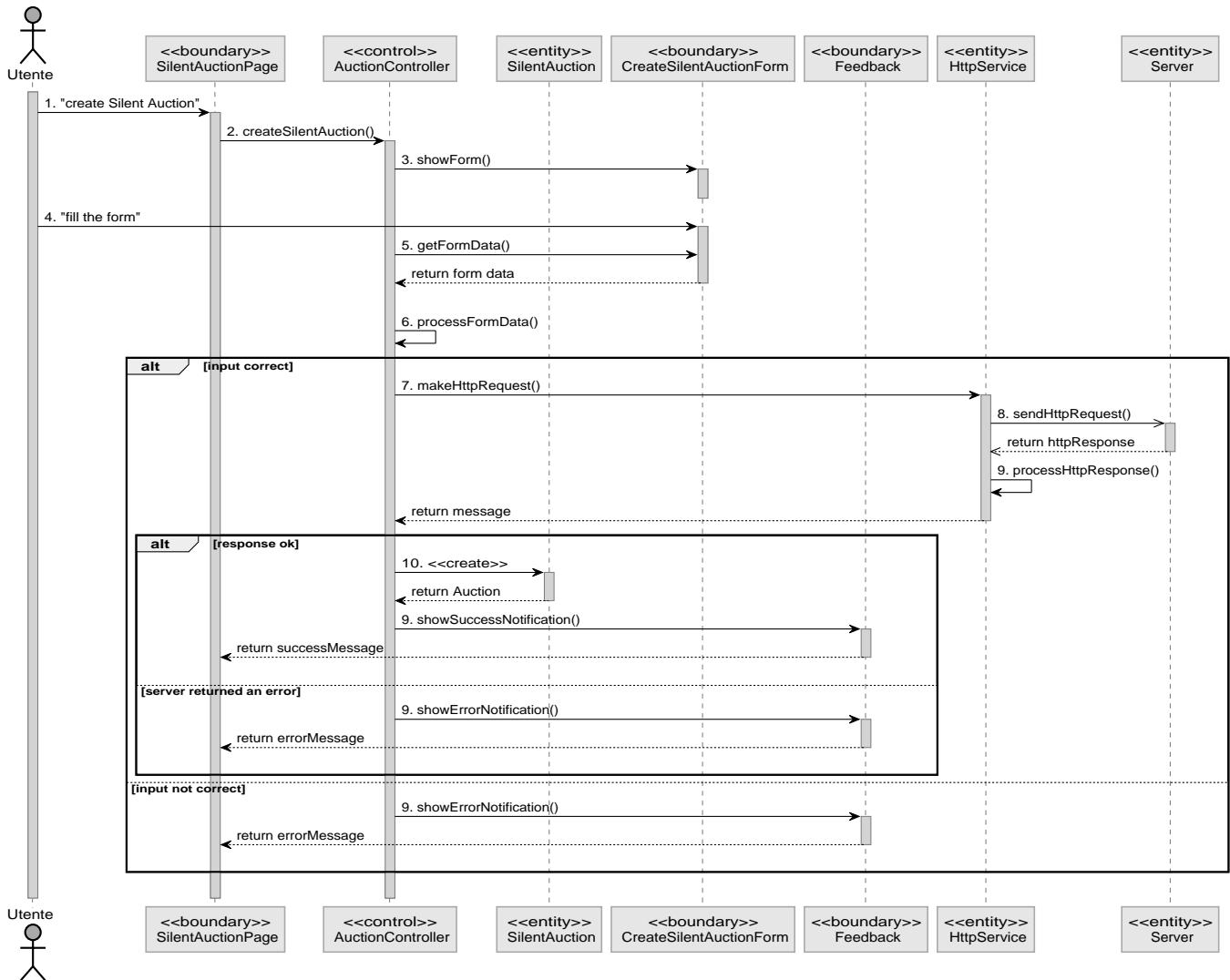
4.5 Diagrammi di Sequenza di Design

4.5.1 Caso d'uso: Creazione Asta Silenziosa

Il seguente Sequence Diagram descrive il processo di creazione di un'Asta Silenziosa.

L'attore principale è l'utente venditore, che compila il form. Il sistema elabora i dati e invia una richiesta al server.

Se la richiesta di creazione è andata a buon fine, il sistema si aggiorna correttamente, mantenendo così la coerenza con il server esterno, e restituisce un messaggio all'utente mediante l'interfaccia grafica. Altrimenti, elabora il messaggio di errore ricevuto dal server e restituisce il messaggio d'errore corrispondente all'utente.

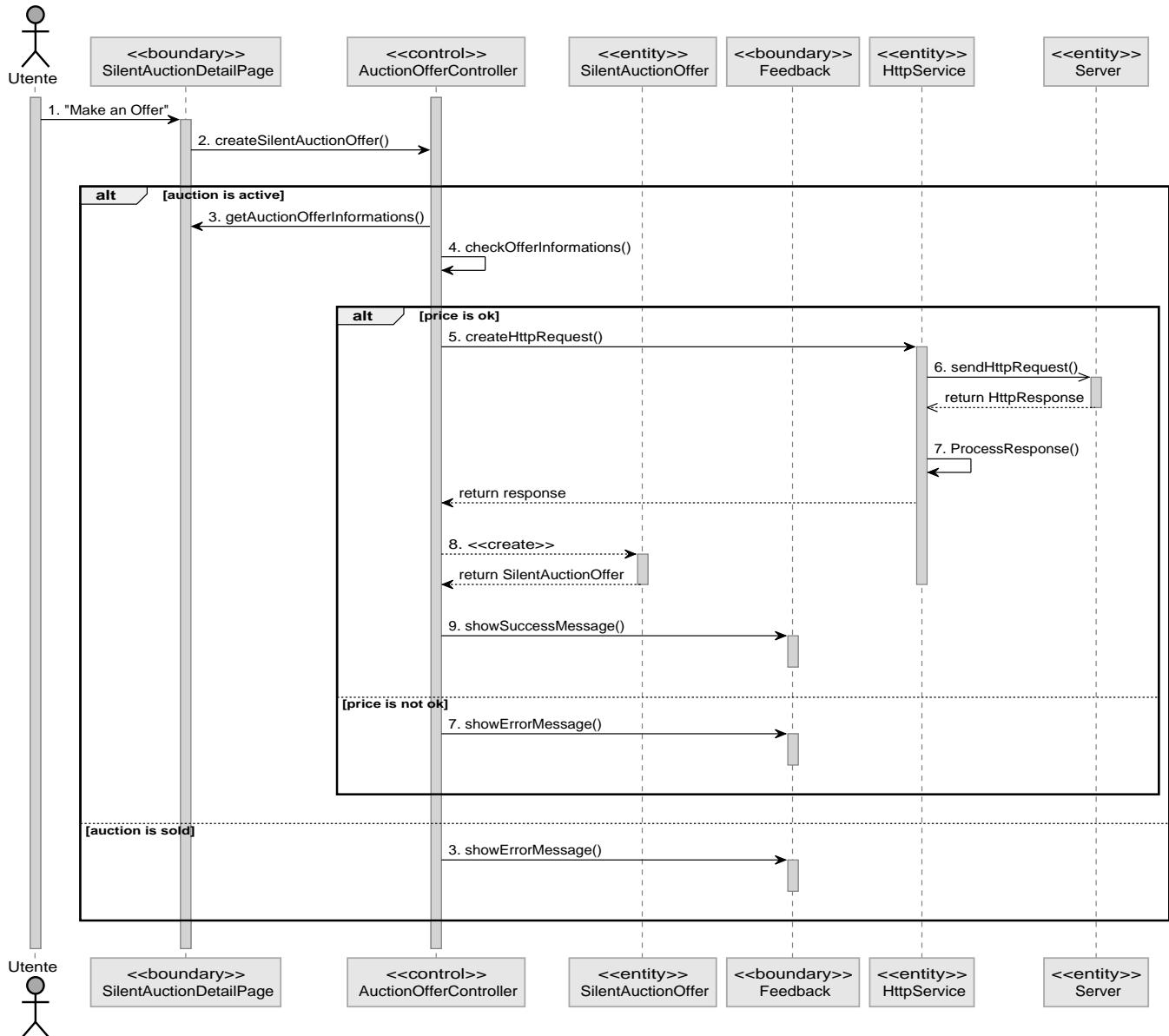


4.5.2 Caso d'uso: Offerta Asta Silenziosa

Il seguente Sequence Diagram descrive il processo di creazione di un'offerta per l'Asta Silenziosa.

L'attore principale è l'utente acquirente, che inserisce i dati necessari alla creazione dell'offerta, compilando il form, e invia i dati al sistema.

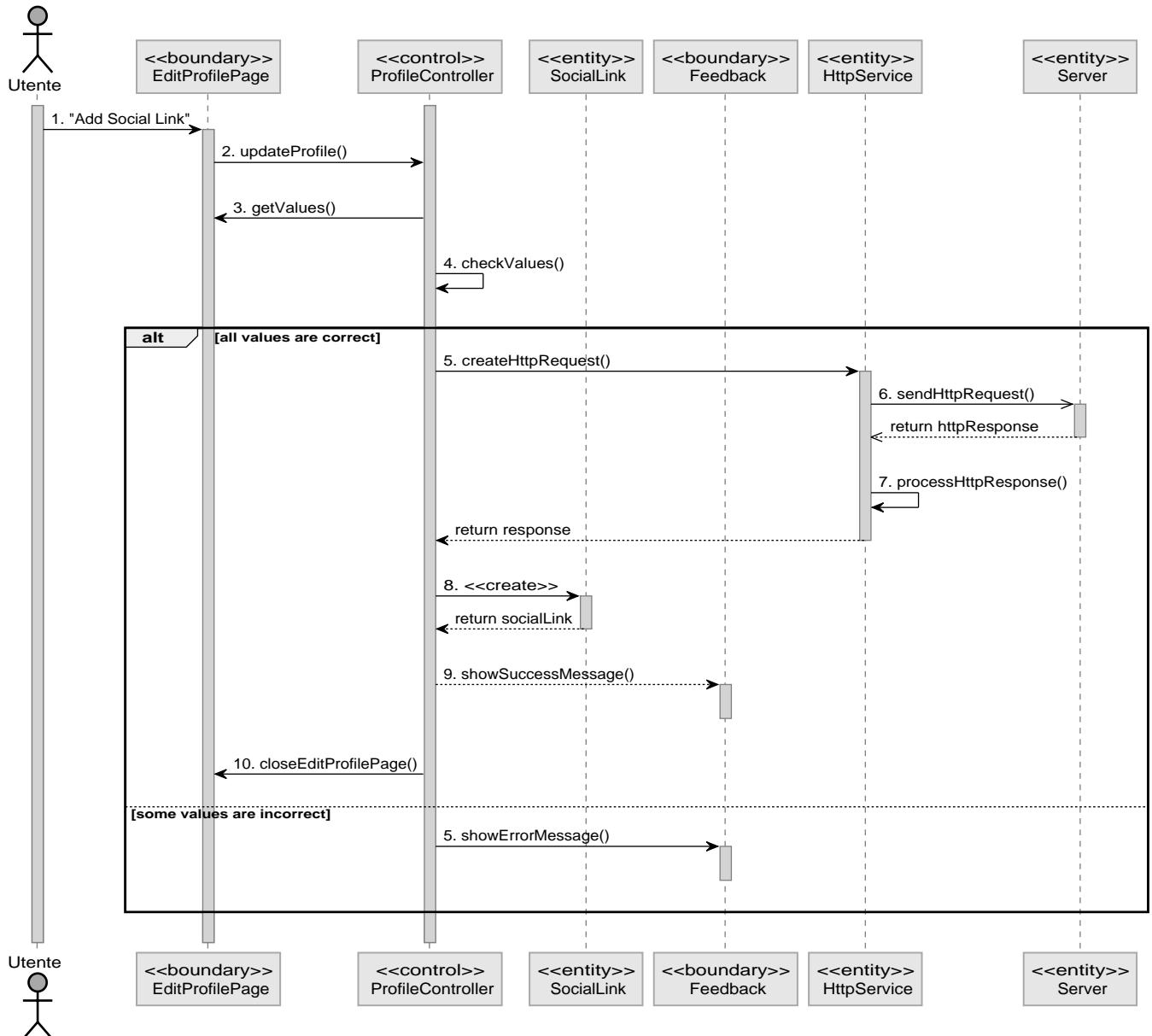
Il sistema verifica che l'asta sia attiva, e invia una richiesta al server per la creazione dell'offerta, restituendo l'esito all'utente, mediante l'interfaccia grafica.



4.5.3 Caso d'uso: Aggiunta Link Social

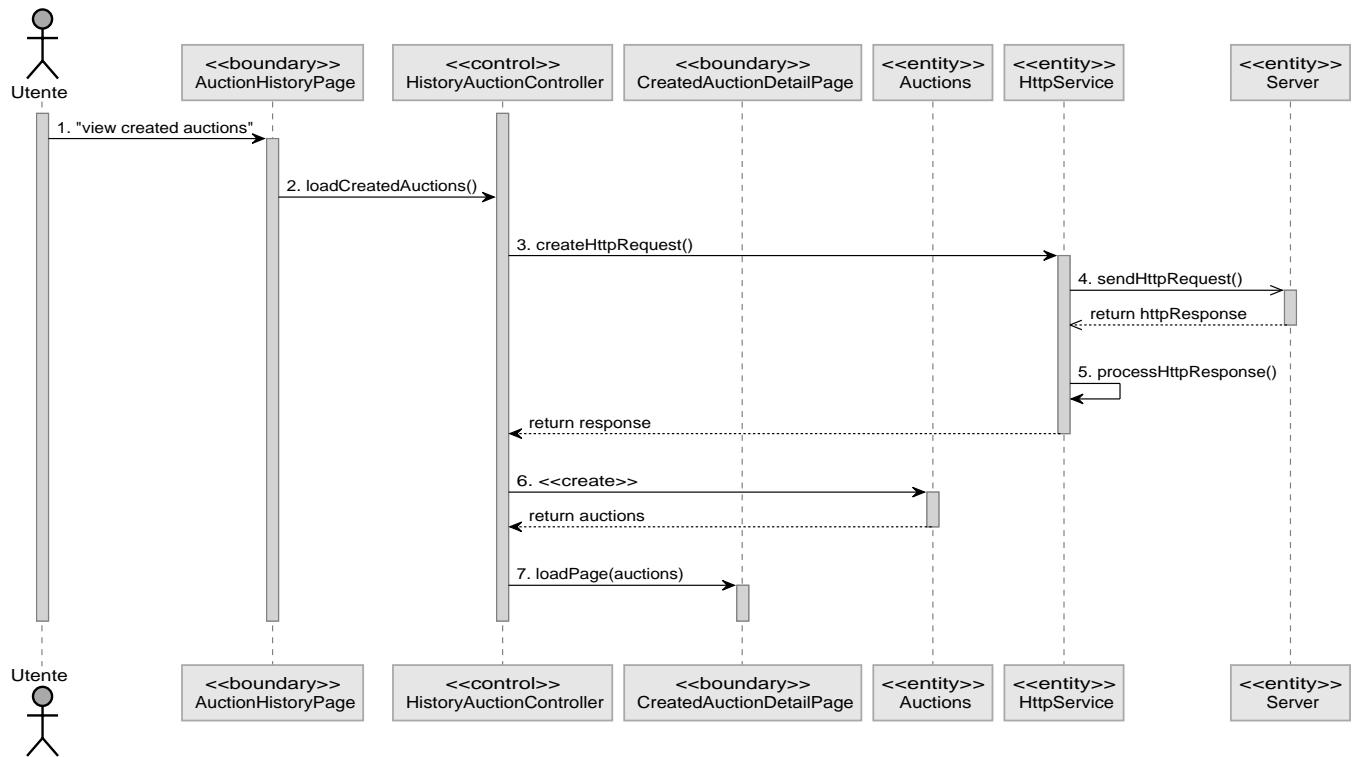
Il seguente Sequence Diagram descrive il processo di aggiunta di un nuovo link social all'interno della pagina profilo dell'utente.

Il sistema prende, in generale, tutti i valori presenti nel form, e verifica che siano corretti. A questo punto invia una richiesta al server e restituisce un risultato. Nel caso l'esito sia positivo, aggiorna il model correttamente e notifica l'utente.



4.5.4 Caso d'uso: Storico Aste create

Il seguente Sequence Diagram descrive il processo per cui l'utente visualizza lo storico delle aste create. Analogamente ai sequence mostrati in precedenza, l'utente naviga all'interno della schermata e il sistema invia una richiesta al server che restituirà le aste create all'utente.



Strumento di Versioning

5.1 Scelte adottate

GitHub è una piattaforma di hosting per il controllo di versione e la collaborazione, costruita attorno al sistema di versioning distribuito Git. Questo strumento è ampiamente utilizzato nella comunità degli sviluppatori per gestire progetti software.

Per il versioning di **DietiDeals24** abbiamo scelto proprio Github, andando a separare la repository del Client, da quella del Server, per una maggiore organizzazione del progetto.

5.1.1 Vantaggi di GitHub come strumento di versioning

L'utilizzo di GitHub come strumento di versioning presenta numerosi vantaggi, tra cui:

1. Collaborazione facilitata

GitHub consente a più sviluppatori di lavorare contemporaneamente sullo stesso progetto senza conflitti. Attraverso le funzionalità di pull request, gli utenti possono proporre modifiche al codice, che possono essere revisionate e integrate nel progetto principale. Questo processo rende la collaborazione più fluida e organizzata.

2. Tracciamento delle modifiche

Ogni modifica apportata al codice è registrata nella cronologia del repository. Gli sviluppatori possono visualizzare chi ha apportato una modifica, quando e perché, il che aiuta a mantenere una documentazione chiara delle evoluzioni del progetto.

3. Ritorno a versioni precedenti

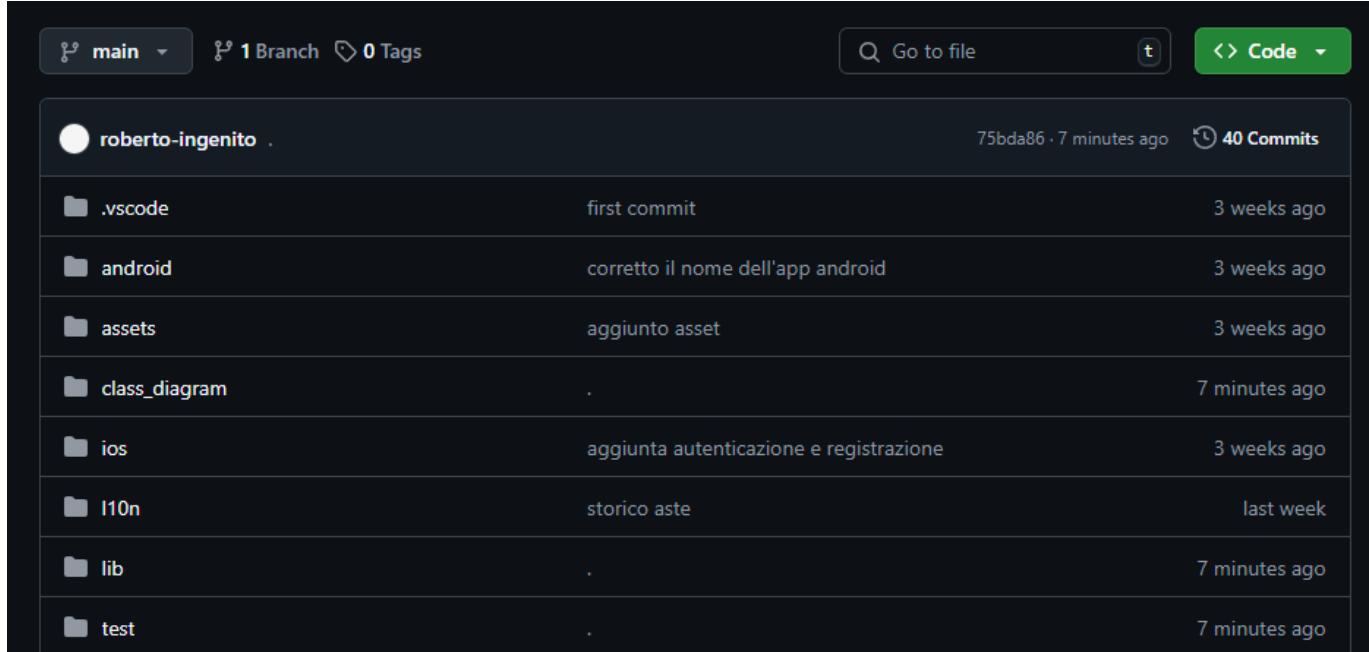
GitHub consente di annullare modifiche o di tornare a versioni precedenti del codice. Questo è particolarmente utile quando una nuova funzionalità introduce un bug o quando è necessario ripristinare un comportamento precedente.

4. Backup e sicurezza

I repository su GitHub sono ospitati nel cloud, fornendo un backup sicuro del codice.

5.2 Repository Client

<https://github.com/progetto-INGSW-23-24/flutter-app>

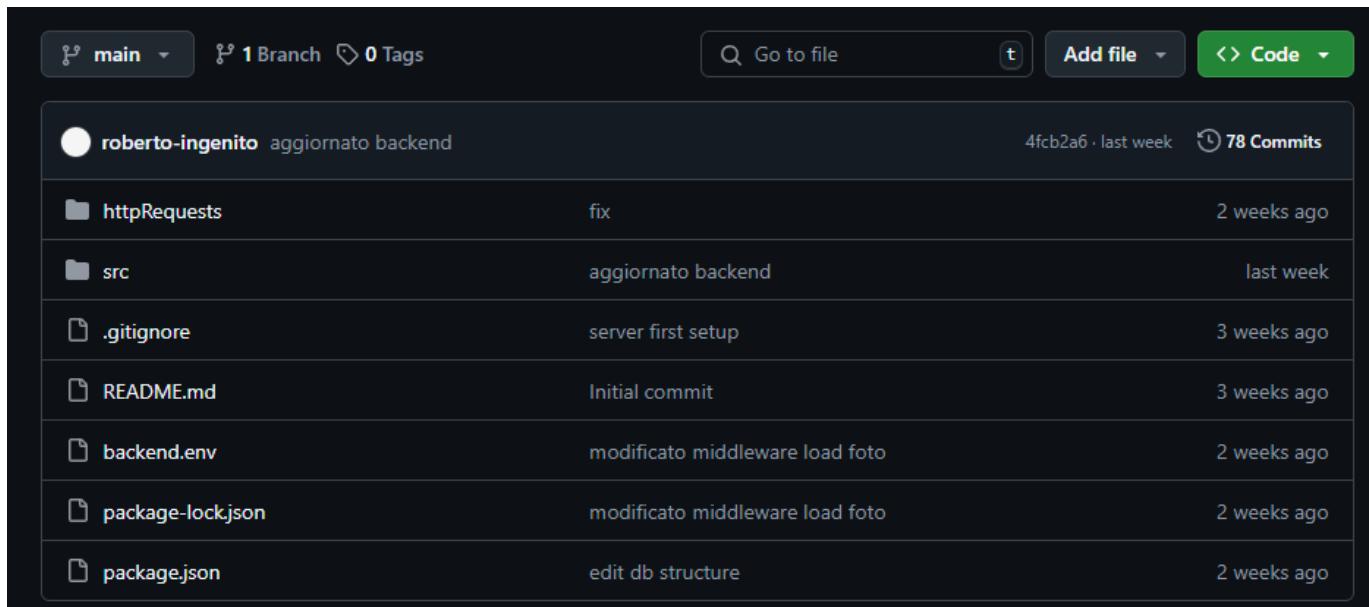


The screenshot shows a GitHub repository page for 'flutter-app'. At the top, it displays 'main' branch, 1 Branch, 0 Tags, a search bar 'Go to file', and a 'Code' button. The commit history is listed below, starting with 'roberto-ingenito' and a commit made 7 minutes ago. The commits are organized by folder: .vscode, android, assets, class_diagram, ios, l10n, lib, and test. Each commit includes a description and a timestamp.

Commit	Description	Time Ago
.vscode	first commit	3 weeks ago
android	corretto il nome dell'app android	3 weeks ago
assets	aggiunto asset	3 weeks ago
class_diagram	.	7 minutes ago
ios	aggiunta autenticazione e registrazione	3 weeks ago
l10n	storico aste	last week
lib	.	7 minutes ago
test	.	7 minutes ago

5.3 Repository Server

<https://github.com/progetto-INGSW-23-24/backend>



The screenshot shows a GitHub repository page for 'backend'. At the top, it displays 'main' branch, 1 Branch, 0 Tags, a search bar 'Go to file', an 'Add file' button, and a 'Code' button. The commit history is listed below, starting with 'roberto-ingenito' and a commit made last week. The commits are organized by file: httpRequests, src, .gitignore, README.md, backend.env, package-lock.json, and package.json. Each commit includes a description and a timestamp.

Commit	Description	Time Ago
roberto-ingenito aggiornato backend	4fcf2a6 · last week	78 Commits
httpRequests	fix	2 weeks ago
src	aggiornato backend	last week
.gitignore	server first setup	3 weeks ago
README.md	Initial commit	3 weeks ago
backend.env	modificato middleware load foto	2 weeks ago
package-lock.json	modificato middleware load foto	2 weeks ago
package.json	edit db structure	2 weeks ago

Report di Qualità del Codice

La qualità del codice rappresenta un elemento fondamentale nello sviluppo software. Un codice di alta qualità facilita la comprensione e la manutenzione da parte degli sviluppatori, riduce il numero di bug e vulnerabilità, aumentando la stabilità dell'applicazione.

Inoltre, un codice ben strutturato e pulito è essenziale per garantire la scalabilità del progetto. Permette di aggiungere nuove funzionalità senza compromettere le performance o l'affidabilità del sistema.

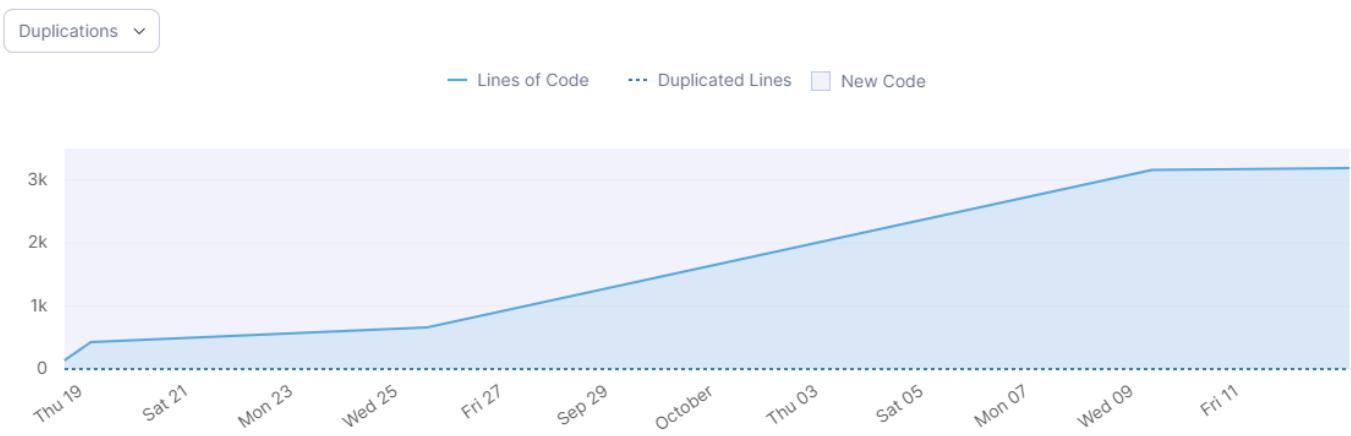
6.1 SonarQube

Per monitorare e migliorare la qualità del codice, è stato impiegato SonarQube. Questo permette di identificare potenziali bug, vulnerabilità e problemi tecnici, fornendo suggerimenti per ottimizzare il codice.

- **Duplicazioni:** Le duplicazioni di codice possono portare a una maggiore complessità e difficoltà nella manutenzione. SonarQube rileva segmenti di codice duplicati e suggerisce di refattorizzarli per migliorare l'efficienza e ridurre il rischio di inconsistenze.
- **Bugs:** Rappresentano errori nel codice che possono causare comportamenti imprevisti o malfunzionamenti dell'applicazione.
- **Vulnerabilità:** Questi indici segnalano potenziali rischi di sicurezza presenti nel codice.
- **Code Smells:** Indicano aree del codice che, pur non essendo errori diretti, possono ridurre la leggibilità, la manutenibilità o la performance dell'applicazione.

Duplicazioni

Nel nostro progetto, il grafico delle duplicazioni di codice si è mantenuto costante nel tempo, nonostante l'incremento delle linee di codice complessive. Questo andamento positivo indica che, anche con la crescita del codice, siamo riusciti a gestire efficacemente la duplicazione, mantenendo un livello di duplicazione sotto controllo.



Errori

Il grafico degli errori del nostro progetto presenta un andamento particolarmente positivo: inizia con un numero elevato di *code smells* e mostra una costante diminuzione fino a raggiungere lo zero.

La diminuzione degli errori nel codice ha incrementato la stabilità dell'applicazione e riducendo i malfunzionamenti. Inoltre, ha facilitato la manutenzione e l'evoluzione del software, permettendo di aggiungere nuove funzionalità con maggiore sicurezza e minor rischio di introdurre nuovi bug.



Testing e Valutazione sul campo dell'Usabilità

7.1 Unit Testing

All'interno del progetto lo Unit Testing è stato implementato seguendo la metodologia **Black Box**. Per l'implementazione di suddetti test è stato utilizzato il framework di testing nativo di Flutter che segue il modello xUnit come richiesto dal committente.

7.1.1 checkEmailAndPassword

Questo metodo ha la funzione di verificare che l'email e la password inserite in fase di registrazione rispettino i criteri di dominio per poter essere considerate valide.

Analizziamo quelle che sono le classi di equivalenza con le quali andremo a realizzare il testing:

EMAIL

- A1: L'email rispetta il pattern dell'espressione regolare (classe valida)
- A2: L'email non rispetta il pattern dell'espressione regolare (classe non valida)

PASSWORD

- B1: La password contiene da 0 a 7 caratteri (classe non valida)
- B2: La password contiene da 8 a 16 caratteri (classe valida)
- B3: La password contiene più di 16 caratteri (classe non valida)

Utilizziamo come criterio di copertura R-WECT (Weak Robust Equivalence Class Testing) il quale ci garantisce un buon grado di copertura evitando però la scrittura di molti test case. In definitiva, un buon compromesso rispetto ai criteri SECT e WECT.

Testiamo ogni classe un'unica volta escludendo le combinazioni di classi non valide, otteniamo 3 test cases: A1 x B1, A2 x B2, A1 x B3.

```
1 group('Test checkEmailAndPassword con R-WECT', () {
```

```
2 // Caso 1: Email valida, password troppo breve
3 test('Email valida e password troppo breve (A1, B1)', () => {
4   expect(checkEmailAndPassword('test@example.com', 'short')).toBeFalsy();
5 });
6
7 // Caso 2: Email non valida, password valida
8 test('Email non valida e password valida (A2, B2)', () => {
9   expect(checkEmailAndPassword('invalid-email', 'validPass1')).toBeFalsy();
10 });
11
12 // Caso 3: Email valida, password troppo lunga
13 test('Email valida e password troppo lunga (A1, B3)', () => {
14   expect(checkEmailAndPassword('test@example.com', 'thisPasswordIsWayTooLong')).toBeFalsy();
15 });
16 });
```

7.1.2 checkPriceDifference

Questo metodo verifica che nella creazione dell'Asta al Ribasso, il prezzo minimo sia minore del prezzo iniziale e che siano entrambi maggiori di zero.

Analizziamo le classi di equivalenza:

minPrice

- A1: Il prezzo minimo è maggiore di zero e minore di startingPrice (classe valida)
- A2: Il prezzo minimo è minore di zero (classe non valida)
- A3: Il prezzo minimo è maggiore di zero e maggiore di startingPrice (classe non valida)

startingPrice

- B1: Il prezzo iniziale è maggiore di zero (classe valida)
- B2: Il prezzo iniziale è minore di zero (classe non valida)

Potremmo utilizzare anche per questo metodo, il criterio di copertura R-WECT. Chiediamoci però, cosa succederebbe se il prezzo minimo fosse un valore molto vicino allo zero?

Consideriamo quindi una tecnica in questo caso migliore per effettuare il testing, il criterio **Boundary Values**, ossia il criterio dei valori limite. In questo caso, il numero di test case è pari a 9.

Per ogni parametro andiamo a testare: Il valore minimo, appena sopra il minimo, appena sotto al massimo, valore massimo e un test case con tutti i valori medi.

```

1 class PriceValidator {
2     static bool validatePrices(double minPrice, double startingPrice) {
3         if (minPrice <= 0 || startingPrice <= 0) return false;
4         if (minPrice >= startingPrice) return false;
5         return true;
6     }
7 }
8
9 group('Test dei valori limite per checkPriceDifference', () {
10     // Test con valori medi
11     test('Test 1: Valori medi', () {
12         expect(PriceValidator.validatePrices(50, 100), true);
13     });
14
15     // Test per minPrice (mantenendo startingPrice costante a 100)

```

```
16 test('Test 2: minPrice al minimo', () {
17     expect(PriceValidator.validatePrices(0.01, 100), true);
18 });
19
20 test('Test 3: minPrice appena sopra il minimo', () {
21     expect(PriceValidator.validatePrices(0.02, 100), true);
22 });
23
24 test('Test 4: minPrice appena sotto il massimo', () {
25     expect(PriceValidator.validatePrices(98, 100), true);
26 });
27
28 test('Test 5: minPrice al massimo', () {
29     expect(PriceValidator.validatePrices(99, 100), true);
30 });
31
32 // Test per startingPrice (mantenendo minPrice costante a 50)
33 test('Test 6: startingPrice al minimo', () {
34     expect(PriceValidator.validatePrices(50, 50.01), true);
35 });
36
37 test('Test 7: startingPrice appena sopra il minimo', () {
38     expect(PriceValidator.validatePrices(50, 50.02), true);
39 });
40
41 test('Test 8: startingPrice appena sotto il massimo', () {
42     expect(PriceValidator.validatePrices(50, 999.99), true);
43 });
44
45 test('Test 9: startingPrice al massimo', () {
46     expect(PriceValidator.validatePrices(50, 1000), true);
47 });
48
49 // Test aggiuntivi per casi non validi (per completezza)
50 test('Test extra: minPrice negativo', () {
51     expect(PriceValidator.validatePrices(-1, 100), false);
52 });
53
54 test('Test extra: startingPrice negativo', () {
55     expect(PriceValidator.validatePrices(50, -1), false);
```

```
56  });
57
58 test('Test extra: minPrice maggiore di startingPrice', () {
59   expect(PriceValidator.validatePrices(100, 50), false);
60 });
61});
```

7.1.3 checkNewOfferPrice

Questo metodo verifica che il nuovo importo per un'offerta in Asta all'Inglese sia maggiore di zero e maggiore dell'ultima offerta.

Analizziamo le classi di equivalenza:

newAmount

- A1: La nuova offerta è maggiore di zero e maggiore della precedente (classe valida)
- A2: La nuova offerta è minore di zero (classe non valida)
- A3: La nuova offerta è maggiore di zero ma minore della precedente (classe non valida)

previousAmount

- B1: La vecchia offerta è maggiore di zero (classe valida)
- B2: La vecchia offerta è minore di zero (classe non valida)

Utilizziamo il criterio di copertura R-WECT sopra citato. Andiamo ad effettuare i seguenti test cases: A1 x B1, A2 x B1, A3 x B1, A1 x B2. Il numero di test cases necessari sarà 4.

```

1 class OfferValidator {
2     static bool validateOffer(double newAmount, double previousAmount) {
3         if (newAmount <= 0 || previousAmount <= 0) return false;
4         if (newAmount <= previousAmount) return false;
5         return true;
6     }
7 }
8
9 group('R-WECT Tests for Bid Validation', () {
10    // Test Case 1: A1 x B1 (tutte le classi valide)
11    test('Test Case 1 (A1 x B1): Nuova offerta valida e vecchia offerta valida', () {
12        // newAmount > previousAmount > 0
13        expect(OfferValidator.validateOffer(100, 50), true);
14    });
15
16    // Test Case 2: A2 x B1 (newAmount negativo)
17    test('Test Case 2 (A2 x B1): Nuova offerta negativa', () {
18        // newAmount < 0, previousAmount > 0
19        expect(OfferValidator.validateOffer(-10, 50), false);
20    });
21

```

```
22 // Test Case 3: A3 x B1 (newAmount minore del precedente)
23 test('Test Case 3 (A3 x B1): Nuova offerta minore della precedente', () {
24     // 0 < newAmount < previousAmount
25     expect(OfferValidator.validateOffer(40, 50), false);
26 });
27
28 // Test Case 4: A1 x B2 (previousAmount negativo)
29 test('Test Case 4 (A1 x B2): Vecchia offerta negativa', () {
30     // newAmount > 0, previousAmount < 0
31     expect(OfferValidator.validateOffer(100, -10), false);
32 });
33 })
```

7.1.4 checkBaseAndRaiseThreshold

Questo metodo verifica che nella creazione dell'asta all'inglese la base d'asta e la soglia di rialzo siano entrambi maggiori di zero.

Analizziamo le classi di equivalenza:

basePrice

- A1: La base d'asta è minore di zero (classe non valida)
- A2: La base d'asta è maggiore di zero (classe valida)

increaseThreshold

- B1: La soglia di rialzo è minore di zero (classe non valida)
- B2: La soglia di rialzo è maggiore di zero (classe valida)

Utilizziamo il criterio di copertura **Boundary Values**, ossia dei valori limiti. Il numero di test cases è sempre 9.

```

1 class AuctionValidator {
2
3     static bool validateAuctionParameters(double basePrice, double increaseThreshold) {
4
5         if (basePrice <= 0 || increaseThreshold <= 0) return false;
6
7         return true;
8     }
9 }
10
11
12
13
14
15
16
17
18
19
20
21
22
23

```

```

8 group('Boundary Value Analysis Tests for Auction Parameters', () {
9     // Test con valori nominali (il +1)
10    test('Test 1: Valori nominali', () {
11        expect(AuctionValidator.validateAuctionParameters(100, 10), true);
12    });
13
14    // Test per basePrice (primi 4 test, mantenendo increaseThreshold costante)
15    test('Test 2: basePrice al minimo', () {
16        expect(AuctionValidator.validateAuctionParameters(0.01, 10), true);
17    });
18
19    test('Test 3: basePrice appena sopra il minimo', () {
20        expect(AuctionValidator.validateAuctionParameters(0.02, 10), true);
21    });
22
23    test('Test 4: basePrice appena sotto il massimo', () {

```

```

24     expect(AuctionValidator.validateAuctionParameters(999.99, 10), true);
25 );
26
27 test('Test 5: basePrice al massimo', () {
28     expect(AuctionValidator.validateAuctionParameters(1000, 10), true);
29 );
30
31 // Test per increaseThreshold (ultimi 4 test, mantenendo basePrice costante)
32 test('Test 6: increaseThreshold al minimo', () {
33     expect(AuctionValidator.validateAuctionParameters(100, 0.01), true);
34 );
35
36 test('Test 7: increaseThreshold appena sopra il minimo', () {
37     expect(AuctionValidator.validateAuctionParameters(100, 0.02), true);
38 );
39
40 test('Test 8: increaseThreshold appena sotto il massimo', () {
41     expect(AuctionValidator.validateAuctionParameters(100, 99.99), true);
42 );
43
44 test('Test 9: increaseThreshold al massimo', () {
45     expect(AuctionValidator.validateAuctionParameters(100, 100), true);
46 });
47 });

```

7.2 Valutazione dell'Usabilità sul campo

La valutazione dell'usabilità sul campo si concentra sull'analisi del sistema **DietiDeals24** nel suo ambiente reale d'uso. Abbiamo utilizzato le stesse euristiche di Nielsen già applicate nella valutazione a priori, per verificare come il sistema supporti effettivamente l'esperienza utente, oltre a raccogliere feedback attraverso test pratici e analisi dei dati raccolti durante l'uso reale del prodotto.

7.2.1 Obiettivi della valutazione sul campo

L'obiettivo principale è verificare se i principi di usabilità previsti a priori siano effettivamente rispettati e se il sistema soddisfi le aspettative di usabilità degli utenti. Inoltre, si vuole identificare se emergono problematiche inattese nell'uso reale che non erano state rilevate nella fase

precedente.

7.2.2 Metodologia

Test con utenti reali

Abbiamo selezionato un gruppo rappresentativo di utenti, inclusi venditori e acquirenti, che hanno utilizzato il sistema nel contesto reale. Le attività testate includono:

- Registrazione di un account (sia come venditore che come compratore)
- Creazione e gestione di aste
- Ricerca di aste tramite filtri e parole chiave
- Partecipazione a un'asta attiva
- Modifica del profilo

Durante i test, abbiamo monitorato:

- **Tempo impiegato** per completare ogni attività
- **Numero di errori** commessi durante le operazioni
- **Feedback soggettivi** tramite questionari post-attività, per valutare il livello di soddisfazione e difficoltà percepite.

7.2.3 Valutazione delle euristiche di Nielsen

Di seguito riportiamo un'analisi euristica basata sui dati raccolti, confrontata con la valutazione a priori.

1. Visibilità dello stato del sistema

Valutazione a priori: Era stato previsto l'uso di alert e messaggi di conferma per garantire all'utente la visibilità dello stato del sistema.

Valutazione sul campo: I test sul campo hanno confermato che i messaggi di conferma sono efficaci nel fornire feedback immediati. Tuttavia, alcuni utenti hanno riportato che i messaggi a volte sono troppo brevi o poco evidenti, soprattutto quando si verificano errori durante la partecipazione a un'asta. Nei log, abbiamo rilevato alcuni casi in cui gli utenti hanno ripetuto l'azione di invio dell'offerta perché non erano sicuri che fosse stata registrata correttamente.

Intervento suggerito: Aumentare la durata o la visibilità dei messaggi di conferma, soprattutto in caso di errore.

2. Corrispondenza tra il sistema e il mondo reale

Valutazione a priori: L'interfaccia è stata progettata per utilizzare un linguaggio semplice e comprensibile, evitando termini tecnici.

Valutazione sul campo: Gli utenti hanno apprezzato l'uso di un linguaggio chiaro, ma nei questionari alcuni hanno suggerito che la terminologia relativa alle diverse tipologie di aste potrebbe essere ulteriormente semplificata, soprattutto per gli utenti principianti.

Intervento suggerito: Migliorare le spiegazioni delle tipologie di aste attraverso l'uso di tooltip o una breve guida introduttiva.

3. Controllo dell'utente e libertà

Valutazione a priori: Era stata prevista l'implementazione di dialog di conferma per annullare o confermare azioni critiche.

Valutazione sul campo: I dialog di conferma sono risultati utili, ma alcuni utenti hanno segnalato che in certe operazioni, come la modifica del profilo o la cancellazione di un'asta, sarebbe preferibile avere opzioni di annullamento più visibili.

Intervento suggerito: Aggiungere un'opzione "annulla" più evidente nei flussi di azione complessi.

4. Coerenza e standard

Valutazione a priori: L'interfaccia era stata progettata utilizzando i componenti di Material Design, garantendo una coerenza nelle interazioni.

Valutazione sul campo: Non sono emersi problemi significativi di coerenza, e gli utenti hanno elogiato la familiarità dell'interfaccia, soprattutto quelli che avevano già esperienza con altre applicazioni basate su Material Design.

5. Prevenzione degli errori

Valutazione a priori: Era stato implementato un sistema di dialog per prevenire errori comuni.

Valutazione sul campo: Durante i test, abbiamo notato una riduzione significativa degli errori grazie alla presenza di dialog di conferma. Tuttavia, i log hanno mostrato che alcuni utenti hanno inserito offerte non valide in alcune aste, indicando che i controlli di input potrebbero essere migliorati.

Intervento suggerito: Rafforzare i controlli di input nelle offerte e fornire messaggi d'errore più dettagliati.

6. Riconoscimento piuttosto che ricordo

Valutazione a priori: Era stato previsto l'uso di info button per aiutare gli utenti a riconoscere le funzionalità chiave, come la creazione di aste.

Valutazione sul campo: Gli info button sono stati utilizzati correttamente dagli utenti, ma alcuni hanno segnalato che potrebbero essere resi più visibili o interattivi per fornire maggiori informazioni in tempo reale.

Intervento suggerito: Migliorare la visibilità degli info button e offrire spiegazioni più dettagliate in caso di dubbi.

7. Flessibilità ed efficienza

Valutazione a priori: Il sistema era stato progettato per essere utilizzabile sia da principianti che da utenti esperti.

Valutazione sul campo: Gli utenti principianti hanno trovato l'interfaccia facile da usare, mentre gli utenti esperti hanno apprezzato la possibilità di utilizzare scorciatoie e funzionalità avanzate come i filtri. Tuttavia, alcuni suggeriscono l'aggiunta di funzionalità personalizzabili, come la possibilità di salvare le ricerche.

Intervento suggerito: Aggiungere opzioni avanzate per gli utenti esperti, come ricerche salvate e scorciatoie personalizzabili.

8. Design estetico e minimalista

Valutazione a priori: L'interfaccia era stata progettata per essere pulita e priva di elementi inutili.

Valutazione sul campo: L'aspetto minimalista è stato molto apprezzato e non sono stati segnalati problemi di sovraccarico informativo.

9. Aiuto e documentazione

Valutazione a priori: Era stato previsto un sistema di aiuto e suggerimenti accessibile tramite info button.

Valutazione sul campo: Gli utenti hanno trovato utile la documentazione disponibile, ma alcuni hanno suggerito di rendere più accessibili le guide per principianti, soprattutto in fase di registrazione e creazione delle prime aste.

Intervento suggerito: Introdurre una guida interattiva per i nuovi utenti.

10. Gestione degli errori

Valutazione a priori: Era stato previsto un sistema di alert con messaggi chiari.

Valutazione sul campo: I log hanno mostrato alcuni errori ripetuti, indicando che gli utenti non sempre capiscono la causa di un errore. Ad esempio, i messaggi relativi alle offerte non valide potrebbero essere più esplicativi.

Intervento suggerito: Fornire spiegazioni più dettagliate nei messaggi di errore per evitare confusione.

7.2.4 Conclusioni e raccomandazioni

La valutazione sul campo ha confermato che molte delle previsioni fatte nella valutazione a priori sono state rispettate, garantendo una buona usabilità del sistema. Tuttavia, l'uso reale del prodotto ha evidenziato alcune aree di miglioramento, soprattutto in termini di visibilità dei messaggi di conferma, gestione degli errori e personalizzazione per utenti esperti.