

Progettazione di una piattaforma mobile "DietiDeals24" per la gestione di aste online

Acronimo del progetto: **INGSW2324**

Numero del progetto: **18**

Funzionalità assegnate: **[1,2,3,6,7]**

**Dipartimento di Ingegneria Elettrica e delle Tecnologie
dell'Informazione**

C.D.L. in Informatica

Committente: Software Engineering UniNA

Docente 1: Prof. Sergio di Martino

Docente 2: Prof. Francesco Cutugno

Data di inizio: 10 Dicembre 2023

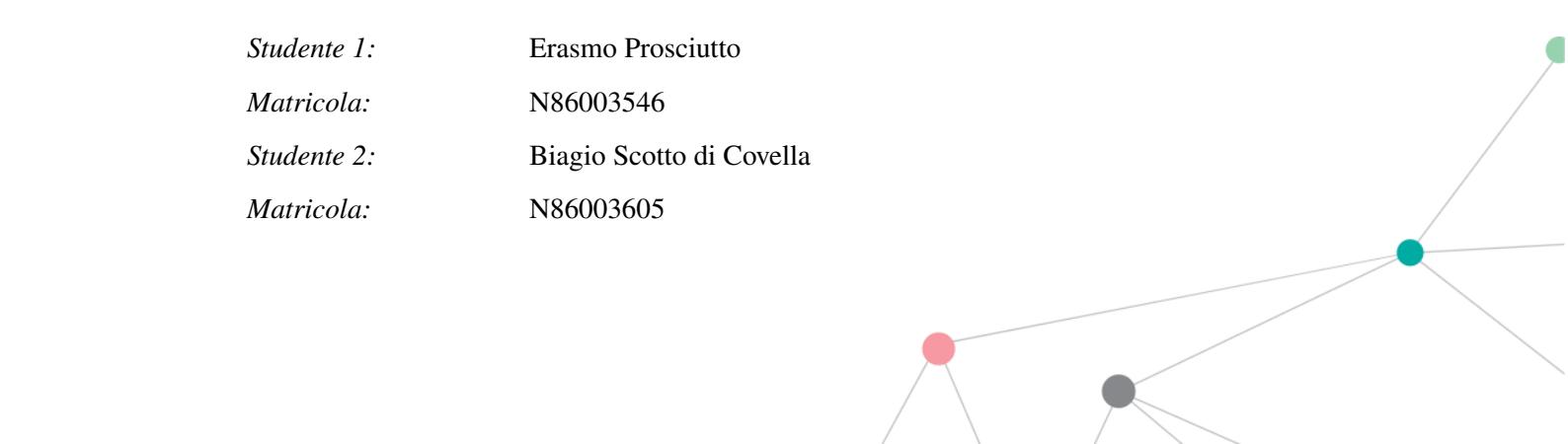
Data di fine: 11 Aprile 2024

Studente 1: Erasmo Prosciutto

Matricola: N86003546

Studente 2: Biagio Scotto di Covella

Matricola: N86003605





Revisioni

Versione	Data	Descrizione	Autore
1.0	10 Dicembre 2023	Creazione file	Scotto di Covella, Prosciutto
1.1	11 Dicembre 2023	Sistemazione sezioni PDF e grafica	Scotto di Covella
1.2	15 Dicembre 2023	Stesura sezioni del documento dei Requisiti Software	Scotto di Covella, Prosciutto
1.3	17 Dicembre 2023	Inserimento immagini, tabelle e stesura descrizioni	Scotto di Covella
1.4	18 Dicembre 2023	Revisione ortografica, stilistica e contenutistica	Prosciutto
1.5	25 Dicembre 2023	Completamento stesura Analisi dei requisiti	Scotto di Covella, Prosciutto
1.6	17 Dicembre 2023	Inserimento immagini, tabelle e stesura descrizioni	Scotto di Covella
1.7	30 Dicembre 2023	Completamento stesura Specifica dei requisiti	Scotto di Covella, Prosciutto
2.1	10 Gennaio 2024	Stesura sezioni del documento di Design del sistema	Scotto di Covella, Prosciutto
2.2	15 Gennaio 2024	Inserimento immagini, tabelle e stesura descrizioni	Scotto di Covella
2.3	20 Gennaio 2024	Revisione ortografica, stilistica e contenutistica	Prosciutto
2.4	21 Gennaio 2024	Completamento stesura del documento di Design del sistema	Scotto di Covella, Prosciutto
3.1	22 Gennaio 2024	Stesura sezioni del codice sorgente sviluppato	Prosciutto
3.2	20 Marzo 2024	Completamento stesura del documento di codice sorgente sviluppato	Scotto di Covella, Prosciutto
4.1	21 Marzo 2024	Stesura sezioni del documento di Testing	Scotto di Covella, Prosciutto
4.2	23 Marzo 2024	Inserimento tabelle e codice sviluppato	Scotto di Covella, Prosciutto



Versione	Data	Descrizione	Autore
4.3	24 Marzo 2024	Revisione ortografica, stilistica e contenutistica	Prosciutto
4.4	25 Marzo 2024	Completamento stesura del documento di Testing	Scotto di Covella, Prosciutto
5.0	11 Aprile 2024	Revisione generale e chiusura progetto	Scotto di Covella, Prosciutto



Indice

Revisioni	3
Elenco delle figure	7
Elenco delle tabelle	8
Parte - introduttiva	13
Parte I - Requisiti del Software	15
1. Analisi dei Requisiti	15
1.1 Requisiti di dominio	15
1.2 Requisiti Funzionali.....	15
1.3 Requisiti non funzionali.....	16
1.4 Modellazione di tutti i casi d'uso richiesti.....	17
1.5 Individuazione del target degli utenti	19
1.6 Descrizioni Testuali Strutturate per due casi d'uso significativi.....	22
1.7 Prototipazione visuale via Mock-up dell'interfaccia utente per tutti i casi d'uso	26
1.8 Valutazione dell'usabilità a priori	41
1.9 Test di usabilità: prime prove.....	43
1.10 Test di usabilità: prove dopo i miglioramenti	46
1.11 Considerazioni finali usabilità	49
2. Specifica dei Requisiti	52
2.1 Classi, oggetti e relazioni di analisi: Class Diagram.....	52
2.2 Diagrammi di sequenza di analisi per casi d'uso significativi: Sequence Diagram	56
2.3 Prototipazione funzionale via statechart dell'interfaccia grafica: Statechart.....	58
Parte II - Design del sistema	61
3. Analisi dell'architettura con esplicita definizione dei criteri di design.....	61
3.1 Criteri di Design	61
3.2 Architettura DietiDeals24	61
4. Descrizione/motivazione delle scelte tecnologiche adottate e strumenti di implementazione.....	64
5. Esempio di funzionamento del software.....	67
6. Diagramma delle classi di design	68
7. Diagrammi di sequenza di design per i casi d'uso	69
8. Database schema	72
Parte III - Codice Sorgente	74
9. Codice Sorgente sviluppato, comprensivo di eventuale Dockerfile	74
10. File di build automatica.....	74
11. Evidenza dell'uso di strumenti di versioning.....	77
12. Report di qualità del codice, generati da SonarQube o similari (nel caso solo per il back-end)....	78
13. Chiamate API	79
Parte IV - Testing	82



14. Testing e valutazione sul campo dell'usabilità	82
14.1 Individuazione e descrizione dei metodi	82
14.2 Metodo validateLogin	82
14.3 Metodo validatePin.....	86
14.4 Metodo caricaDatiJson	91
14.5 Metodo savePreferences	95
15. Valutazione dell'usabilità sul campo	101
Considerazioni finali	107
Riferimenti bibliografici.....	108



Elenco delle figure

Figura 1: Use Case di Registrazione	17
Figura 2: Use Case di Accesso	17
Figura 3: Use Case di Gestione Aste	18
Figura 4: Use Case di Interazione Utente-Aste	18
Figura 5: Use Case di Gestione Account	19
Figura 6: Use Case di Scelta Dashboard	19
Figura 7: Personas n.1, Maria Rossi.....	21
Figura 8: Personas n.2, Sara Conti.....	21
Figura 9: Personas n.3, Giovanni Verdi.....	22
Figura 10: Personas n.4, Luca Bianchi	22
Figura 11: Studio UI	27
Figura 12: Schermata apertura applicazione	29
Figura 13: Schermata informativa 1	29
Figura 14: Schermata informativa 2	29
Figura 15: Schermata informativa 3	30
Figura 16: Schermata di benvenuto	30
Figura 17: Schermata di accesso.....	30
Figura 18: Schermata di registrazione 1	30
Figura 19: Schermata di registrazione 2	30
Figura 20: Schermata di registrazione 3	30
Figura 21: Schermata registrazione completata	31
Figura 22: Schermata di recupero password 1	31
Figura 23: Schermata di recupero password 2	31
Figura 24: Schermata di recupero password 3.....	32
Figura 25: Schermata termini e condizioni.....	32
Figura 26: Schermata home acquirente	32
Figura 27: Schermata categorie	33
Figura 28: Schermata aste silenziose	33
Figura 29: Schermata aste ribasso.....	33
Figura 30: Schermata dettaglio asta ribasso.....	34
Figura 31: Schermata dettaglio asta ribasso2	34
Figura 32: Schermata dettaglio asta silenziosa	34
Figura 33: Schermata dettaglio asta silenziosa2	35
Figura 34: Schermata dettaglio asta silenziosa3	35
Figura 35: Schermata dettaglio asta ribasso3	35
Figura 36: Schermata profilo venditore informazioni.....	35
Figura 37: Schermata profilo venditore negozio	35
Figura 38: Schermata profilo	35
Figura 39: Schermata modifica credenziali	36
Figura 40: Schermata modifica profilo 1.....	36
Figura 41: Schermata modifica profilo 2.....	36
Figura 42: Schermata acquisti	36
Figura 43: Schermata vendite	36
Figura 44: Schermata seguite	36
Figura 45: Schermata home venditore	37
Figura 46: Schermata aste silenziose	37
Figura 47: Schermata aste ribasso	37
Figura 48: Schermata crea asta ribasso.....	37



Figura 49: Schermata crea asta ribasso2	37
Figura 50: Schermata crea asta silenziosa	37
Figura 51: Schermata asta ribasso	38
Figura 52: Schermata asta silenziosa	38
Figura 53: Schermata asta silenziosa chiusa	38
Figura 54: Schermata asta ribasso chiusa.....	39
Figura 55: Schermata cerca	39
Figura 56: Schermata filtri	39
Figura 57: Schermata Vendi	40
Figura 58: Schermata Vendi ribasso	40
Figura 59: Schermata Vendi silenziosa.....	40
Figura 60: Schermata pop-up.....	40
Figura 61: Schermata Schede generiche	41
Figura 62: Schermata Schede stato asta.....	41
Figura 63: ClassDiagram - Entità software.....	53
Figura 64: ClassDiagram - Registrazione	53
Figura 65: ClassDiagram - Accesso.....	54
Figura 66: ClassDiagram - Gestore Aste	54
Figura 67: ClassDiagram - Gestore Utente-Aste.....	55
Figura 68: ClassDiagram - Scelta Dashboard	55
Figura 69: ClassDiagram - Gestore Account	56
Figura 70: SequenceDiagram - Crea asta al ribasso.....	57
Figura 71: SequenceDiagram - Acquista da asta al ribasso	58
Figura 72: StatechartDiagram - Crea asta al ribasso	58
Figura 73: StatechartDiagram - Acquista da asta al ribasso	59
Figura 74: Diagramma delle classi di design Front-end	68
Figura 75: Diagramma delle classi di design Back-end	69
Figura 76: Diagramma di sequenza di design per il caso d'uso Crea asta al ribasso	70
Figura 77: Diagramma di sequenza di design per il caso d'uso Acquista asta al ribasso	71
Figura 78: Database schema	72
Figura 79: Github.....	77
Figura 80: Sonarquebe - Analisi qualità codice	79
Figura 81: Control Flow Graph di validateLogin.	85
Figura 82: Control Flow Graph di validatePin.....	89
Figura 83: Control Flow Graph di caricaDatiDaJson.	94
Figura 84: Control Flow Graph di savePreferences.	99
Figura 85: Personas n.1, Maria Rossi.....	101
Figura 86: Personas n.2, Sara Conti.....	101
Figura 87: Personas n.3, Giovanni Verdi.....	101
Figura 88: Personas n.4, Luca Bianchi	101
Figura 89: Firebase - Monitoraggio schermate	104
Figura 90: Firebase - Schermate più utilizzate	105
Figura 91: Firebase - Tempo medio coinvolgimento.....	105



Elenco delle tabelle

Tabella 2: Use Case #Gestione Aste - Crea Asta al Ribasso	24
Tabella 3: Use Case #Interazione Utente-Aste - Acquista da asta al ribasso	26
Tabella 5: Vantaggi nell'utilizzare i MockUp come strumento di valutazione dell'usabilità.	26
Tabella 7: Tabella di valutazione.....	42
Tabella 8: Tabella Compiti Generali	44
Tabella 9: Tabella Compiti Acquirente	44
Tabella 10: Tabella Compiti Venditore.....	44
Tabella 11: Calcolo Usabilità Compiti Utente.....	45
Tabella 12: Calcolo Usabilità Compiti Acquirente	45
Tabella 13: Calcolo Usabilità Compiti Venditore	46
Tabella 14: Calcolo Usabilità - Risultati totali	46
Tabella 15: Tabella Compiti Generali	47
Tabella 16: Tabella Compiti Acquirente	48
Tabella 17: Tabella Compiti Venditore.....	48
Tabella 18: Calcolo Usabilità UI Migliorata Compiti Utente.....	48
Tabella 19: Calcolo Usabilità UI Migliorata Compiti Acquirente	49
Tabella 20: Calcolo Usabilità UI Migliorata Compiti Venditore.....	49
Tabella 21: Calcolo Usabilità UI Migliorata - Risultati totali	49
Tabella 23: Metodi candidati per Unit Testing	82
Tabella 24: Classi di Equivalenza del metodo validateLogin	83
Tabella 25: Combinazioni delle Classi di Equivalenza del metodo validateLogin	83
Tabella 26: Dati di test per il metodo validateLogin presenti sul database	84
Tabella 27: Node Coverage per il metodo validateLogin	86
Tabella 28: Branch Coverage per il metodo validateLogin	86
Tabella 29: Classi di Equivalenza del metodo validatePin	87
Tabella 30: Combinazioni delle Classi di Equivalenza del metodo validatePin	87
Tabella 31: Node Coverage per il metodo validatePin.....	90
Tabella 32: Branch Coverage per il metodo validatePin	90
Tabella 33: Modified Condition Coverage per il metodo validatePin.....	91
Tabella 34: Classi di Equivalenza del metodo caricaDatiDaJson	92
Tabella 35: Test Cases del metodo caricaDatiDaJson	92
Tabella 36: Branch Coverage per il metodo caricaDatiDaJson	95
Tabella 37: Classi di Equivalenza del metodo savePreferences	96
Tabella 38: Test Cases del metodo savePreferences.....	96
Tabella 39: Branch Coverage per il metodo savePreferences	100
Tabella 40: Tabella Compiti Generali	102
Tabella 41: Tabella Compiti Acquirente	102
Tabella 42: Tabella Compiti Venditore.....	102
Tabella 43: Calcolo Usabilità UI Prodotto finale Compiti Utente	103
Tabella 44: Calcolo Usabilità Prodotto finale Compiti Acquirente	103
Tabella 45: Calcolo Usabilità UI Prodotto finale Compiti Venditore	103
Tabella 46: Calcolo Usabilità prodotto finito - Risultati totali.....	104



Glossario

Dominio Informatico

Sequence Diagram Rappresentazione grafica delle interazioni tra gli oggetti in una sequenza temporale.

Class Diagram Diagramma che rappresenta le classi di un sistema e le loro interazioni.

Stakeholders Individui o organizzazioni che hanno un interesse nel progetto o sistema.

Usabilità Misura di quanto sia facile ed efficace l'uso di un sistema o prodotto da parte degli utenti.

Server Sistema informatico che fornisce servizi ad altri sistemi chiamati clienti.

White Box Test basato sulla conoscenza interna del sistema.

Requisiti non funzionali Vincoli generali e proprietà qualitative del sistema come la sicurezza, l'usabilità o la performance.

Back-end Parte del software che gestisce la logica, i database e le operazioni del server.

Android Sistema operativo mobile sviluppato da Google.

Requisiti funzionali Definiscono le funzionalità o i servizi specifici che il sistema dovrebbe offrire.

Design Pattern Soluzione ripetibile a un problema comune nel design del software.

Performance Capacità di un sistema di svolgere le sue funzioni entro limiti temporali specificati.

StateChart Diagramma che rappresenta gli stati di un oggetto e le transizioni tra questi stati.

Disponibilità Probabilità che un sistema sia operativo e funzionante quando richiesto.

Black Box Test basato sulle specifiche senza considerare il funzionamento interno.

Affidabilità Probabilità che un sistema funzioni senza errori in un determinato periodo di tempo.

Ergonomia Scienza che si occupa della relazione tra gli esseri umani e i sistemi con cui interagiscono.

Mock-up Modello visivo di alto livello di un prodotto o sistema.

Sicurezza Protezione di un sistema contro accessi o usi non autorizzati.

Scalabilità Abilità di un sistema di gestire un aumento di carico di lavoro.

Tabelle di CockBurn Strumenti usati per definire in dettaglio uno specifico caso d'uso.

Robustezza Capacità di un sistema di gestire situazioni non previste o errori senza interrompere le sue funzioni principali.

Portabilità Facilità con cui un software può essere trasferito da un ambiente di lavoro a un altro.

Front-end Interfaccia utente e tutto ciò che l'utente può vedere e interagire.

Manutenzione Facilità con cui un sistema può essere corretto, migliorato o adattato.

Personas Rappresentazioni fittizie degli utenti finali basate su dati reali.

Use Case Diagram Rappresentazione grafica delle interazioni tra gli utenti e un sistema.



Dominio DietiDeals24

DietiDeals24 Piattaforma online per la gestione di aste che offre servizi accessibili tramite applicazioni mobile, desktop o web-based.

Utente Individuo che utilizza DietiDeals24 e che può essere un venditore o un acquirente.

Account Venditore/Acquirente Profilo creato sulla piattaforma DietiDeals24 che permette di vendere o acquistare beni/servizi.

Registrazione Processo di creazione di un nuovo account su DietiDeals24.

Credenziali di Terze Parti Informazioni di accesso fornite da servizi esterni come Google, Facebook o GitHub.

Profilo Utente Pagina personale di un utente su DietiDeals24 dove può aggiungere informazioni personali come bio, link, area geografica.

Asta Procedura di vendita o acquisto dove gli utenti possono proporre offerte per beni/servizi.

Bene/Servizio Oggetto o attività offerti per la vendita su DietiDeals24.

Categoria Classificazione dei beni/servizi in vendita, utilizzata per organizzare e facilitare la ricerca all'interno della piattaforma.

Ricerca Funzionalità che permette di trovare aste attive tramite filtri come categoria e parole chiave.

Asta al Ribasso Tipo di asta con prezzo iniziale alto che diminuisce con il tempo fino a un prezzo minimo o finché un'offerta viene accettata.

Prezzo Iniziale/Minimo Valore di partenza per un'asta al ribasso e il valore segreto al di sotto del quale il venditore non è disposto a scendere.

Timer di Decremento Conto alla rovescia che determina quando il prezzo di un'asta al ribasso viene ridotto.

Asta Silenziosa Tipo di asta dove le offerte sono segrete e il venditore può accettare o rifiutare le proposte ricevute in modo discreto.

Offerta Segreta Proposta di acquisto che non viene rivelata agli altri partecipanti dell'asta silenziosa.

Notifica Messaggio informativo inviato agli utenti per comunicare eventi all'interno della piattaforma (es. esito di un'asta).

Codice Recupero Password Codice temporaneo fornito agli utenti per consentire il recupero delle credenziali d'accesso dimenticate.

Aste in Evidenza Aste selezionate o in primo piano sulla piattaforma, spesso per la loro popolarità o rilevanza.

Aste Seguite Aste che un utente ha scelto di seguire o monitorare per ricevere aggiornamenti o notifiche.

Acquista Funzione che consente agli utenti di effettuare l'acquisto di un bene o servizio.

Vendi Funzione che permette agli utenti di mettere in vendita i propri beni o servizi.

Cerca Funzionalità di ricerca che aiuta gli utenti a trovare specifiche aste sulla piattaforma.



Profilo Sezione personale dove l'utente può visualizzare e modificare le proprie informazioni, impostazioni e attività sulla piattaforma.

Segui Asta Opzione che permette di ricevere aggiornamenti sullo stato di un'asta specifica.

Offri Azione di fare un'offerta in un'asta.

Negozi Sezione della piattaforma dove gli utenti possono visualizzare le aste dell'utente venditore selezionato.

Vinto Stato che indica quando un utente ha fatto l'offerta vincente in un'asta.

Conclusa Stato di un'asta che è terminata, sia che sia stata vinta da un'offerta o che sia scaduto il tempo senza offerte vincenti.

Tipologia Asta Categorie di asta come asta al ribasso, asta silenziosa, ecc., che definiscono le regole e le modalità di svolgimento.

Titolo Denominazione o nome dato a un'asta o a un bene/servizio in vendita.

Sottocategoria Ulteriore suddivisione all'interno di una categoria per dettagliare ulteriormente il tipo di bene o servizio.

Descrizione Prodotto Dettagli e informazioni relative al bene o servizio offerto in asta.

Data Fine Asta Il momento in cui un'asta si chiude e non sono più accettate ulteriori offerte.

Importo Decrementato La quantità in cui il prezzo di un'asta al ribasso viene ridotto in ogni fase del timer di decremento.

Prezzo Minimo Segreto Il prezzo minimo nascosto in un'asta al ribasso sotto il quale il venditore non accetta offerte.

Aperse Aste attualmente in corso e aperte a offerte dagli utenti.

Chiuse Aste che sono state concluse, indipendentemente dal fatto che siano state vinte o meno.



Parte introduttiva



Il mondo digitale è in continua evoluzione, spingendo le aziende e gli individui a cercare soluzioni innovative per soddisfare le esigenze quotidiane in modo efficiente e tecnologicamente avanzato. In questo contesto, "DietiDeals24" emerge come una piattaforma all'avanguardia per la vendita e l'acquisto di aste online, rivoluzionando il modo in cui utenti e aziende interagiscono nel mercato delle aste. L'obiettivo principale di DietiDeals24 è quello di fornire un'esperienza utente fluida, intuitiva e piacevole, garantendo al contempo prestazioni elevate e affidabilità.

La piattaforma è progettata per essere estremamente user-friendly, consentendo agli utenti di registrare facilmente un nuovo account. Un account prevede due sezioni, una per la vendita e una per l'acquisto, in questo modo gli utenti con un'unica registrazione potranno utilizzare entrambi i servizi, invece di dover creare due account separati, uno per la vendita e uno per l'acquisto. Inoltre, la piattaforma supporta la registrazione tramite credenziali di terze parti, come Google, Facebook o GitHub, rendendo il processo ancora più semplice. Gli utenti hanno anche la possibilità di personalizzare il proprio profilo con informazioni quali una breve biografia, link ai propri siti web o social media, e la loro area geografica.

Per quanto riguarda le funzionalità chiave, DietiDeals24 offre diverse opzioni sia per i venditori che per gli acquirenti. Nella sezione di vendita è possibile visualizzare le proprie aste create, oltre a crearne di nuove. Sono disponibili due tipologie di aste, quelle silenziose in cui gli acquirenti inviano offerte segrete al venditore, il quale può accettare l'offerta che preferisce, e quelle al ribasso, dove il venditore stabilisce un prezzo iniziale alto che diminuisce nel tempo fino a che un acquirente non fa un'offerta o si raggiunge il prezzo minimo segreto. Ogni asta include una descrizione dettagliata del bene o servizio offerto e, optionalmente, una fotografia. Le aste sono inoltre organizzate in categorie e sottocategorie (ad esempio, elettronica, informatica, giocattoli, alimentari, servizi, ecc.), facilitando la navigazione e la ricerca da parte degli utenti. Nella sezione dedicata agli acquirenti è invece possibile visionare le aste in evidenza, le aste seguite e le categorie disponibili, con indicazione sul prezzo più basso disponibile per ognuna. E' inoltre possibile, per ogni prodotto in vendita, visualizzare il profilo del venditore, con una dettagliata visione sulle sue informazioni di base (email, città, nazione, social networks) e sui prodotti fino ad ora venduti, oltre a quelli attualmente attivi. Tutto questo permette all'utente che desidera acquistare un prodotto, di fidarsi non solo del venditore in questione ma della piattaforma stessa.

DietiDeals24 permette inoltre ricerche avanzate tramite l'uso di filtri per tipologia (asta silenziosa o al ribasso), stato (chiusa, aperta), prezzo, categoria e sottocategoria.

Il progetto DietiDeals24 è stato concepito con l'obiettivo primario di sviluppare una piattaforma robusta, performante e affidabile per la gestione di aste online. La visione è quella di creare un'applicazione che sia al tempo stesso intuitiva, rapida e piacevole da usare. Nelle seguenti pagine affronteremo nel dettaglio questi quattro principali aspetti:

1. **Requisiti del Software:** Analisi dettagliata dei requisiti di dominio, funzionali e non funzionali, modellazione dei casi d'uso, e valutazione dell'usabilità.
2. **Design del Sistema:** Esposizione dell'architettura del sistema, inclusa la descrizione delle scelte tecnologiche adottate e la progettazione dettagliata.
3. **Codice Sorgente:** Presentazione del codice sorgente sviluppato, comprensivo delle soluzioni di containerizzazione come Docker.
4. **Testing:** Approfondimento delle strategie di testing adottate e valutazione dell'usabilità sul campo.

Attraverso un approccio sistematico e metodico, questo report mira a delineare in modo esaustivo il processo di sviluppo, design e implementazione della piattaforma DietiDeals24. L'obiettivo è quello di fornire una comprensione approfondita delle scelte tecniche, architettoniche e di design adottate, nonché di evidenziare le strategie di testing e valutazione utilizzate per assicurare la massima qualità e affidabilità del sistema.



Parte I

Requisiti del

Software



1 Analisi dei Requisiti

L'analisi dei requisiti è un po' come mettere le fondamenta prima di costruire una casa. Prima di iniziare a sviluppare un software, è fondamentale capire cosa si aspettano gli utenti e cosa il software dovrà effettivamente fare. In questa fase, ci si siede con gli utenti, i clienti e gli stakeholder per discutere, ascoltare e raccogliere tutte le loro necessità e desideri. Una volta raccolte, queste informazioni diventano 'requisiti' - una lista di funzionalità e caratteristiche che il software dovrà avere. Ma non si tratta solo di ciò che il software fa (requisiti funzionali); è anche importante come lo fa, come ad esempio la velocità, la sicurezza e la facilità d'uso (requisiti non funzionali). Una volta definiti, questi requisiti diventano la guida per tutto il team di sviluppo, assicurando che il prodotto finale soddisfi veramente le esigenze di chi lo userà. La fase di analisi dei requisiti prevede la raccolta e la documentazione dei requisiti, l'identificazione dei vincoli del sistema e la definizione dei casi d'uso. L'obiettivo è di fornire una base solida per la progettazione, lo sviluppo, la verifica e la validazione del sistema software.

1.1 Requisiti di dominio

Il progetto DietiDeals24 è stato concepito come un sistema distribuito diviso in due principali componenti:

- **Front-end:** Questa componente si occupa dell'interfaccia utente e dell'interazione con l'utente finale.
- **Back-end:** Questa componente fornisce le API REST che saranno utilizzate dal front-end.

Il back-end è stato confezionato e distribuito come un container Docker. Inoltre, il sistema è ospitato su Amazon AWS, rendendolo quindi accessibile via Internet.

Per la progettazione, sono stati utilizzati strumenti CASE. Si sottolinea l'importanza dell'uso di un approccio che ha favorito l'astrazione, con l'obiettivo di facilitare il riutilizzo del codice e la futura aggiunta di nuove funzionalità. Le decisioni prese per promuovere questa astrazione verranno dettagliatamente descritte nella documentazione seguente.

In termini di tecnologie, il Gruppo di Lavoro ha scelto di sviluppare il software come un'applicazione mobile, implementata con l'ausilio di Android Studio per lo sviluppo frontend e IntelliJ per lo sviluppo backend.

1.2 Requisiti Funzionali

L'applicazione DietiDeals24 è stata progettata per rispondere alle esigenze di venditori e compratori nell'ambito delle aste online, offrendo una serie di servizi specifici. I requisiti funzionali descrivono le funzioni specifiche del software DietiDeals24.

1. Registrazione e Gestione Account

- Possibilità di registrazione e creazione di un nuovo account utente.
- Integrazione con sistemi di autenticazione di terze parti (Google, Facebook, GitHub).
- Possibilità di recupero credenziali
- Possibilità di scegliere tra la dashboard acquirente e la dashboard venditore.

2. Personalizzazione del Profilo Utente

- Aggiunta e modifica di una short bio, link a siti web/social, area geografica e altre informazioni.

3. Creazione e Gestione delle Aste

- Creazione di aste per la vendita di beni/servizi.



- Inclusione di descrizione e fotografie del bene/servizio.
- Classificazione delle aste in categorie.

4. Ricerca e Navigazione

- Funzioni di ricerca con filtri per categoria e parole chiave.
- Visualizzazione dettagli di aste e profili utenti.

5. Asta al Ribasso

- Creazione di aste al ribasso con prezzo iniziale, timer, decremento e prezzo minimo.
- Possibilità di partecipare all'asta acquistando il bene in questione.

6. Asta Silenziosa

- Organizzazione di aste silenziose con offerte segrete e scelta del venditore.
- Possibilità di partecipare all'asta offrendo per il bene in questione.

7. Altre

- Possibilità di visualizzare gli acquisti effettuati
- Possibilità di visualizzare le vendite effettuate
- Possibilità di visualizzare le aste seguite
- Possibilità di ricevere notifiche per una specifica asta

1.3 Requisiti non funzionali

L'usabilità di un'applicazione non è solo una caratteristica, ma un prerequisito fondamentale per garantire un'esperienza utente di qualità. Nel progetto DietiDeals24, poniamo grande enfasi sull'implementazione di principi di usabilità per assicurare che ogni utente possa navigare l'applicazione in modo intuitivo e piacevole. I requisiti non funzionali definiscono standard, qualità e vincoli del sistema.

1. Performance e Affidabilità

- L'applicazione deve risultare veloce di risposta e con prestazioni affidabili.

2. Usabilità e Intuitività

- L'applicazione deve avere un'interfaccia utente intuitiva e facile da navigare.
- L'applicazione deve permettere all'utente di esserne padrone entro poche ore dal primo utilizzo

3. Sicurezza e Protezione dei Dati

- Protezione dei dati personali.

4. Scalabilità

- Capacità di gestire un aumento del volume di utenti e aste.

5. Compatibilità e Integrazione

- Integrazione con servizi di terze parti.

6. Supporto e Manutenzione

- Facilità di manutenzione e aggiornamento.
- Supporto tecnico per gli utenti.

7. Legislazione e Conformità

- Conformità alle leggi e regolamenti del commercio elettronico.

1.4 Modellazione di tutti i casi d'uso richiesti

In questa sezione, vengono presentati i casi d'uso principali per il sistema. Ogni caso d'uso rappresenta un insieme specifico di funzionalità e interazioni che gli utenti possono avere con il sistema.

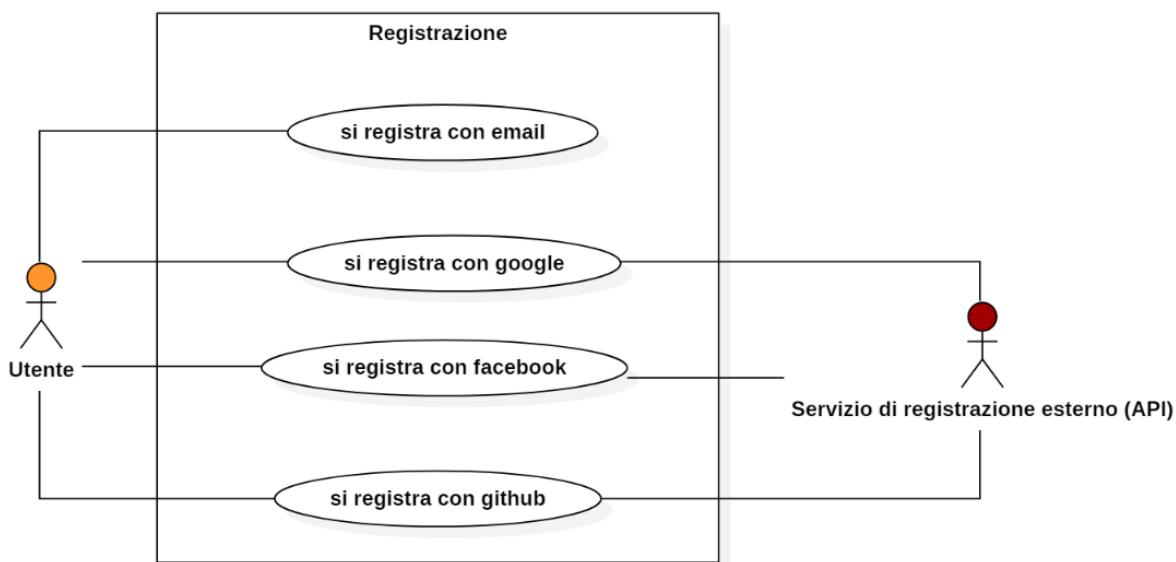


Figura 1: Use Case di Registrazione

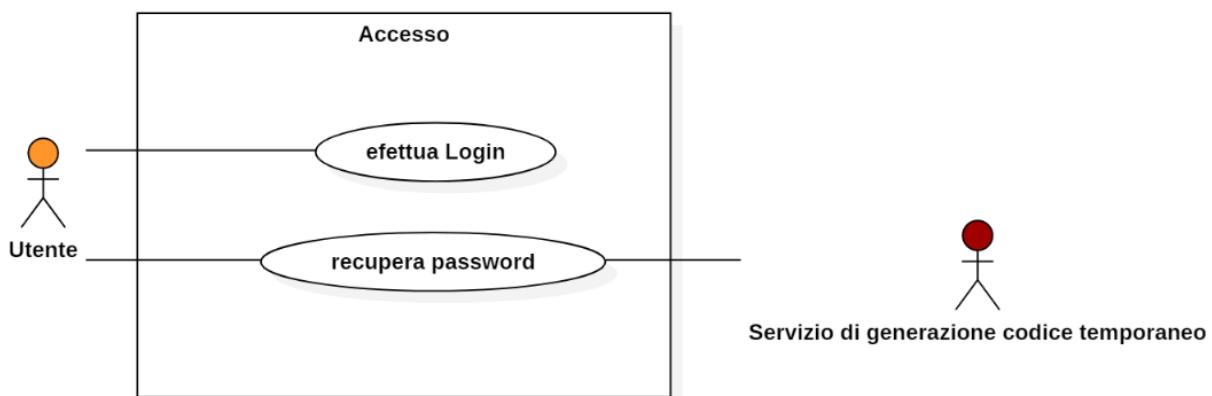


Figura 2: Use Case di Accesso

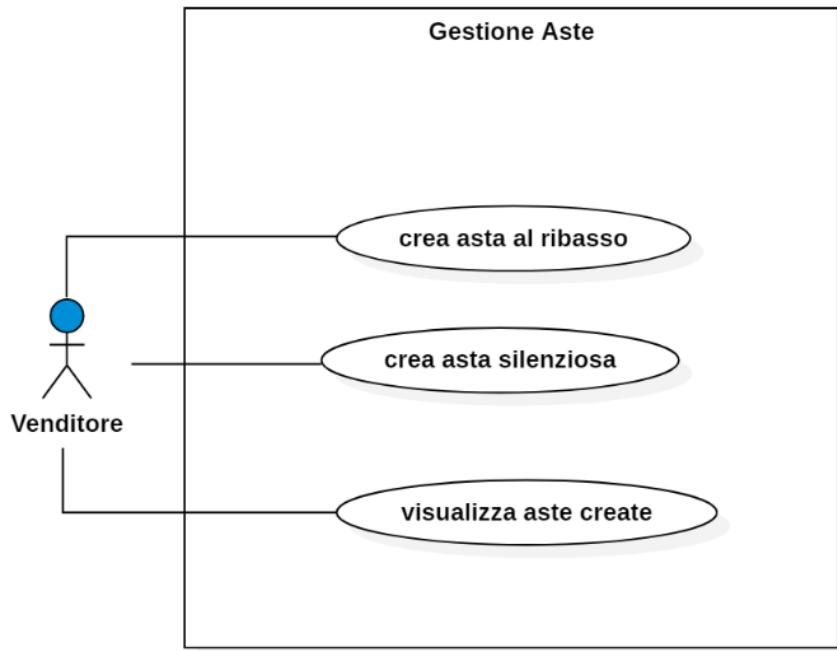


Figura 3: Use Case di Gestione Aste

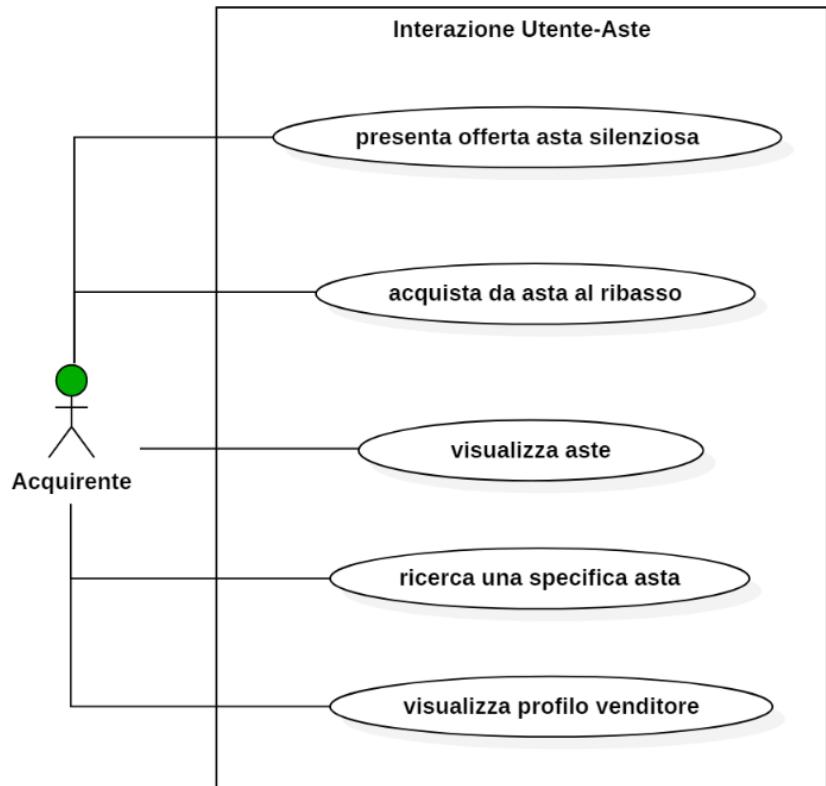


Figura 4: Use Case di Interazione Utente-Aste

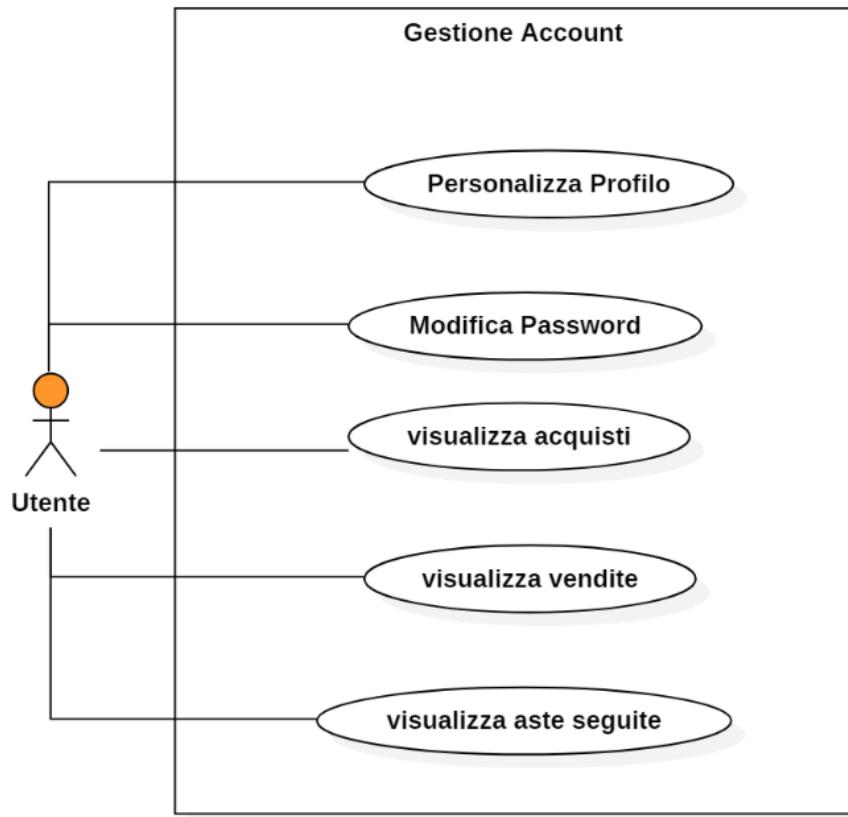


Figura 5: Use Case di Gestione Account

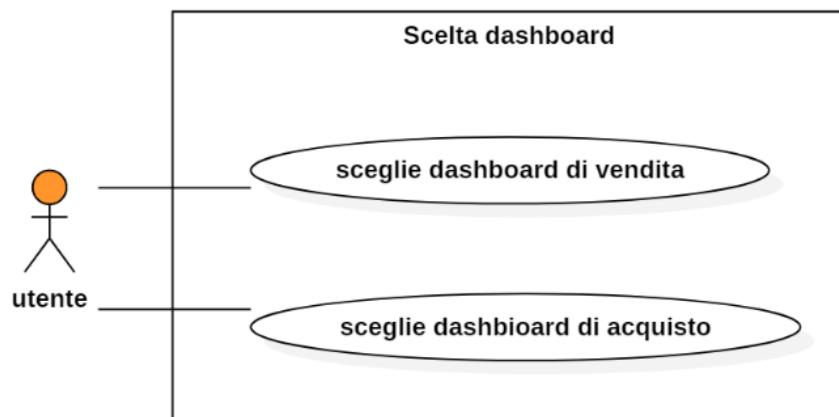


Figura 6: Use Case di Scelta Dashboard

1.5 Individuazione del target degli utenti

Il mercato delle aste online ha visto una notevole crescita negli ultimi anni, in particolare a causa della crescente popolarità del commercio elettronico e dei cambiamenti nei comportamenti di acquisto dovuti alla pandemia di COVID-19. Piattaforme come eBay e Amazon hanno dominato questo spazio, offrendo un'ampia varietà di prodotti e attirando milioni di utenti a livello globale. Una tendenza rilevante è l'incremento delle aste specializzate, che si concentrano su categorie specifiche come l'elettronica, l'informatica, o i collezionabili.



1. Demografia dei Compratori Online:

- Secondo un rapporto di Statista del 2021, la maggior parte degli utenti di e-commerce rientra nella fascia di età tra i 25 e i 34 anni, seguita da quella tra i 35 e i 44 anni.
- La distribuzione di genere nei siti di e-commerce è relativamente equilibrata, anche se in alcune categorie specifiche si possono notare differenze.

2. Stili di Vita e Interessi:

- Rapporti di mercato come quelli di eMarketer e McKinsey mostrano che i consumatori di età compresa tra i 25 e i 45 anni tendono ad essere interessati a tecnologia, fitness, moda, prodotti sostenibili e beni per la casa. Questo gruppo tende ad avere una maggiore consapevolezza ambientale e sociale, preferendo prodotti eco-friendly e aziende con pratiche etiche.

3. Comportamento di Acquisto:

- Secondo una ricerca di Nielsen, gli utenti in questa fascia di età sono più propensi a fare ricerche online prima di effettuare un acquisto, leggendo recensioni e paragonando i prezzi.
- Un rapporto di Deloitte sull'e-commerce ha evidenziato l'importanza di una user experience fluida, sicurezza nelle transazioni e servizi di supporto clienti efficaci.

Un report (4) di Research and Markets sull'interesse dei Giovani e degli High-Net-Worth Millennials, ha fornito dati sul crescente interesse dei giovani e degli High-Net-Worth Millennials nelle aste online, nonché sulle prospettive di crescita del mercato delle aste online. Questo gruppo ha speso di più nelle aste digitali nel 2020, grazie al loro crescente potere d'acquisto mentre entravano nei loro 40 anni.

Sempre secondo questo report, le aste online offrono numerosi vantaggi, come la vasta portata, la disponibilità 24 ore su 24, la facilità d'uso, la varietà di prodotti, e transazioni rapide e sicure. Si prevede che il mercato globale delle aste online raggiungerà i 7,42 miliardi di dollari nel 2023, con un tasso di crescita annuale composto (CAGR) dell'11,50% (2). Inoltre, l'avvento di internet e i progressi tecnologici, come connessioni ad alta velocità e opzioni di pagamento sicure, hanno avuto un impatto significativo sulla crescita del mercato delle aste online. L'uso di dispositivi mobili ha reso possibile partecipare a aste da qualsiasi luogo e in qualsiasi momento.

Crescita del Mercato delle Aste Online

- Il mercato delle aste online ha registrato un aumento significativo, soprattutto nei beni di lusso come arte, orologi, borse e gioielli. Questa crescita è stata in parte alimentata dalla pandemia, che ha accelerato la conversione digitale e ha attratto individui ad alto patrimonio netto (HNWI) verso le aste online. La scarsità di beni disponibili e il desiderio di investimenti tangibili hanno contribuito a questo trend(3).
- Nel 2023, Christie's ha registrato un totale di 590 milioni di euro in vendite per orologi, borse e gioielli, segnando un incremento del 43% rispetto allo stesso periodo del 2022. Gli acquirenti più influenti provengono dall'Asia, Europa, Medio Oriente, Africa e America(5).

Demografia degli Utenti

- Le aste online stanno attirando un pubblico sempre più giovane. Christie's ha registrato che il 35% dei suoi nuovi acquirenti sono millenials, e Sotheby's ha notato un forte interesse da parte di offerenti under-40 e una crescita di acquirenti sotto i 20 anni. Questo indica che i giovani stanno diventando una parte importante del mercato delle aste online(3).

- Gli utenti italiani delle aste online mostrano un trend di ringiovanimento, con un aumento significativo dei visitatori della Generazione Z (+40% in Italia rispetto al 2020), ovvero quelli tra i 18 e i 24 anni. Gli utenti tra i 25 e i 34 anni costituiscono la fascia d'età più rappresentativa, pari al 25% del totale degli offerenti (1).

Personas

Le personas sono profili finti di utenti creati per rappresentare i diversi tipi di persone che potrebbero utilizzare un prodotto o servizio. Sebbene siano immaginarie, sono spesso basate su ricerche reali e sono arricchite da dettagli demografici, comportamentali e psicologici. Questi profili aiutano i designer a visualizzare e comprendere meglio le esigenze, le aspettative e i comportamenti degli utenti. Attraverso le personas, i team di sviluppo possono immergersi nelle esperienze degli utenti, prendendo decisioni più informate durante la progettazione e lo sviluppo di un prodotto. Inoltre, le personas sono spesso accompagnate da scenari o storie che illustrano come potrebbero interagire con il prodotto o servizio in questione.

Maria Rossi

Graphic Designer Freelance

"La creatività è l'intelligenza che si diverte."

Biografia

Maria, 30 anni, è una graphic designer freelance con una passione per l'arte e il design. Lavora da casa a Milano e collabora con diversi clienti internazionali. Vive da sola, appassionata di tecnologia e design moderno.

Hobby e Interessi

Fotografia, viaggi, design d'interni.

Obiettivi

Trovare pezzi unici di design per arredare il suo studio casalingo.

Perché è un potenziale utente

- Ricerca di Oggetti Unici: Come designer, Maria è sempre alla ricerca di oggetti d'arte, mobili o accessori unici per il suo studio o per ispirazione. DietDeals24, con la sua varietà di aste, può offrire una gamma eclettica di oggetti che soddisfano il suo gusto per l'estetica e il design.
- Oportunità di Business: Potrebbe anche usare la piattaforma per vendere i suoi design o opere d'arte, raggiungendo un pubblico più ampio.

Informazioni generali

- Età: 30 anni
- Località: Milano, Italia
- Stile di Vita: Urbano, moderno, autonomo. Lavora principalmente da casa o in spazi di coworking.
- Preferenze Tecnologiche: Predilige dispositivi Apple, appassionata di software di design e applicazioni creative.
- Comportamento d'Acquisto: Cerca prodotti unici, spesso relativi all'arte e al design, con un interesse particolare per oggetti vintage o fatti a mano.

Figura 7: Personas n.1, Maria Rossi

Sara Conti

Proprietaria di un negozio di antiquariato

"Il passato è un prologo."

Biografia

Sara, 45 anni, gestisce un negozio di antiquariato a Firenze. Ha un forte interesse per la storia e gli oggetti vintage. Appassionata di storia dell'arte, ama restaurare mobili antichi.

Hobby e Interessi

Collezionare oggetti antichi, viaggiare, leggere.

Obiettivi

Trovare oggetti rari e unici per il suo negozio.

Perché è un potenziale utente

- Acquisizione di Oggetti Rari e Antichi: La piattaforma può essere un ottimo canale per Sara per trovare e acquisire pezzi rari e antichi per il suo negozio. Le aste online sono luoghi ideali per scoprire oggetti unici che potrebbero non essere disponibili altrove.
- Espansione del Network di Vendita: Sara potrebbe utilizzare DietDeals24 per vendere alcuni dei suoi articoli, raggiungendo un pubblico più vasto e diversificato rispetto al solo negozio fisico.

Figura 8: Personas n.2, Sara Conti



Indice

Parte I



Informazioni generali

- Età: 22 anni
- Località: Torino, Italia
- Stile di Vita: Dinamico, incentrato sugli studi e la vita sociale studentesca.
- Preferenze Tecnologiche: Interessato a nuove tecnologie, ma limitato da un budget studentesco.
- Comportamento d'Acquisto: Cerca offerte per attrezzature tecniche, libri di testo e articoli per hobbies a prezzi accessibili.

Giovanni Verdi

Studente universitario in Ingegneria

"Sogno in grande e osa fallire."

Biografia

Giovanni, 22 anni, è uno studente di Ingegneria a Torino. Vive in un appartamento condiviso con altri studenti.

Hobby e interessi

Tecnologia, sport, musica

Obiettivi

Acquistare attrezzature tecniche e libri a prezzi accessibili.

Perchè è un potenziale utente

- Accesso a Materiali Educativi e Tecnologici a Prezzi Accessibili: Giovanni potrebbe utilizzare la piattaforma per acquistare libri di testo, attrezzature per il suo corso di studio o persino gadget tecnologici a prezzi accessibili, adatti al suo budget da studente.
- Vendita di Libri e Materiale Usato: Inoltre, può vendere i suoi libri di testo o altri articoli non più necessari per recuperare parte dei costi e aiutare altri studenti come lui.



Luca Bianchi

Dirigente in un'azienda IT

"Ogni problema è un'opportunità in maschera."

Biografia

Luca, 38 anni, lavora come dirigente IT a Roma. È sposato e ha due figli. Ama la tecnologia e si tiene sempre aggiornato sulle ultime novità.

Hobby e interessi

Gaming, elettronica, programmare.

Obiettivi

Acquistare gadget elettronici e giochi al miglior prezzo.

Perchè è un potenziale utente

- Gadget e Tecnologia a Buon Prezzo: Luca, essendo un appassionato di tecnologia e gaming, potrebbe utilizzare DietiDeals24 per trovare gadget e giochi elettronici a prezzi vantaggiosi, approfittando delle astre per ottenere il miglior affare possibile.
- Vendita di Articoli Usati: Potrebbe inoltre vendere articoli tecnologici o elettronici usati, rendendo la piattaforma un'ottima scelta per riciclare prodotti in modo efficace.

Figura 9: Personas n.3, Giovanni Verdi

Figura 10: Personas n.4, Luca Bianchi

1.6 Descrizioni Testuali Strutturate per due casi d'uso significativi

USE CASE #Gestione Aste			
Crea asta al ribasso			
Goal in Context	Permettere al Venditore di creare un asta di tipologia al ribasso.		
Preconditions	L'utente ha effettuato l'accesso al sistema DietiDeals24 e si trova nella schermata Home di vendita.		
Success End Condition	Il Venditore crea con successo una nuova asta al ribasso.		
Failed End Condition	Il sistema ha generato un errore e il Venditore non ha potuto creare l'asta al ribasso.		
Primary, Secondary Actors	Venditore.		
Trigger	Il Venditore clicca sul box di "Crea asta al ribasso".		
Description	Step	Venditore	Sistema
	1	Clicca sul box "Crea asta al ribasso"	
	2		Genera schermata con il Form per le informazioni di base dell'asta.
	3	Compila i campi di informazioni di base del Form.	
	4	Clicca su bottone "Continua".	
Continua alla pagina successiva			



Tabella 2 – continua nella pagina successiva

USE CASE #Gestione Aste			
Crea asta al ribasso			
Extensions	Step	Venditore	Sistema
	5		Salva le informazioni di base inserite.
	6		Genera schermata con il Form per le informazioni specifiche dell'asta.
	7	Compila i campi di informazioni specifiche del Form e inserisce l'immagine del prodotto.	
	8	Clicca sul bottone "Crea Asta".	
	9		Genera un pop-up informativo che chiede la conferma a procedere alla creazione dell'asta.
	10	Clicca sul bottone "Conferma".	
	11		Salva le informazioni specifiche e l'immagine inserita.
	12		Genera una nuova asta al ribasso, assegnandogli un codice univoco.
	5.a		Visualizza un errore su tutte le parti del Form che non sono state compilate.
	Si ricollega allo step 3 del main scenario. I campi precedentemente compilati rimangono tali.		
	11.a		Visualizza un errore su tutte le parti del Form che non sono state compilate.
	Si ricollega allo step 7 del main scenario. I campi precedentemente compilati rimangono tali.		
Sub-Variations	Step	Venditore	Sistema
	3.a	Clicca sulla freccia di "Indietro".	
	4		Visualizza schermata Home di vendita.
	Si ricollega allo step 2 del main scenario. Il sistema autocompila i campi con le informazioni precedentemente inserite dal venditore.		
	4.a	Clicca sulla freccia di "Indietro".	
Continua alla pagina successiva			



Tabella 2 – continua nella pagina successiva

USE CASE #Gestione Aste		
Crea asta al ribasso		
5		Visualizza schermata Home di vendita.
Si ricollega allo step 2 del main scenario. Il sistema autocompila i campi con le informazioni precedentemente inserite dal venditore.		
7.a	Clicca sulla freccia di "Indietro".	
8		Visualizza schermata Home di vendita.
Si ricollega allo step 2 del main scenario. Il sistema autocompila i campi con le informazioni precedentemente inserite dal venditore.		
8.a	Clicca sulla freccia di "Indietro".	
Si ricollega allo step 2 del main scenario. Il sistema autocompila i campi con le informazioni precedentemente inserite dal venditore.		
10.a	Clicca sul bottone "Annulla".	
Si ricollega allo step 8 del main scenario. Il sistema mantiene i campi precedentemente compilati.		

Tabella 2: Use Case #Gestione Aste - Crea Asta al Ribasso

USE CASE Interazione Utente-Aste		
Acquista da asta al ribasso		
Goal in Context	Permettere all'acquirente di acquistare un prodotto da un'asta al ribasso.	
Preconditions	L'acquirente ha effettuato l'accesso al sistema DietiDeals24 e ha selezionato la sezione "Acquista"; sono attive delle aste; l'acquirente si trova nella sezione di asta.	
Success End Condition	L'acquirente acquista il prodotto selezionato.	
Failed End Condition	Il sistema ha generato un errore e l'acquisto non va a buon fine.	
Primary, Secondary Actors	Acquirente.	
Trigger	L'acquirente clicca sul box di un'asta.	
Description	Step	Acquirente
	1	Clicca sul box di un'asta.
Continua alla pagina successiva		



Tabella 3 – continua nella pagina successiva

USE CASE Interazione Utente-Aste			
Acquista da asta al ribasso			
Sub-Variations	Step	Acquirente	Sistema
	2		Recupera informazioni dettagliate sull'asta
	3		Visualizza schermata di dettaglio dell'asta.
	4	Clicca sul bottone "Acquista".	
	5		Genera un pop-up informativo che chiede la conferma a procedere all'acquisto.
	6	Clicca sul bottone "Conferma Acquisto".	
	7		Genera una notifica di acquisto andato a buon fine.
	8		Inserisce il prodotto acquistato nella sezione "Prodotti Acquistati" dell'Acquirente.
	9		Visualizza schermata Home di acquisto.
	4.a	Clicca sul bottone "Attiva Notifiche"	
	5		Abilita gli aggiornamenti su quella specifica asta.
	6		Mostra bottone per disabilitare le notifiche.
	Si ricollega allo step 3 del main scenario.		
	4.b	Clicca sul bottone "Disabilita Notifiche".	
	5		Disabilita gli aggiornamenti su quella specifica asta.
	6		Mostra bottone per abilitare le notifiche.
	Si ricollega allo step 3 del main scenario.		
	4.c	Clicca sulla freccia di "Indietro"	
	Si ricollega allo step 3 del main scenario.		
Continua alla pagina successiva			

**Tabella 3 – continua nella pagina successiva**

USE CASE Interazione Utente-Aste			
Acquista da asta al ribasso			
Extensions	6.a	Clicca sul bottone "Annulla"	
		Si ricollega allo step 3 del main scenario.	
	Step	Acquirente	Sistema
	7.a		Genera una notifica di acquisto non andato a buon fine.
		Si ricollega allo step 3 del main scenario.	

Tabella 3: Use Case #Interazione Utente-Aste - Acquista da asta al ribasso

1.7 Prototipazione visuale via Mock-up dell’interfaccia utente per tutti i casi d’uso

Un utilissimo strumento sono i MockUp, una rappresentazione visiva di un’interfaccia utente, spesso realizzata con strumenti grafici o software di progettazione, che non possiede funzionalità operative ma serve a mostrare l’aspetto e la struttura di base di un’interfaccia. I MockUp sono utilizzati per ottenere feedback dagli stakeholder, compresi gli utenti finali, prima che l’interfaccia venga effettivamente costruita.

Vantaggio	Descrizione
Risparmio di Tempo e Risorse	Correggere problemi di usabilità in una fase precoce è molto meno costoso rispetto alla correzione dopo che il software è stato sviluppato.
Coinvolgimento degli Utenti	I mockup permettono di coinvolgere gli utenti nel processo di progettazione, garantendo che le loro esigenze e aspettative siano soddisfatte.
Flessibilità	Essendo rappresentazioni visive senza funzionalità operative, i mockup possono essere facilmente modificati in base ai feedback ricevuti.
Comunicazione Chiara	Forniscono un riferimento visivo concreto che può aiutare a prevenire malintesi tra sviluppatori, designer e stakeholder.

Tabella 5: Vantaggi nell’utilizzare i MockUp come strumento di valutazione dell’usabilità.



DIETIDEALS24

01 COLORI

Sezione Acquista			
#54ACB4 Elementi principali	#75C4CC Elementi secondari	#000000 Testi principali	#7F7F82 Testi secondari
Sezione Vendi			
#DB591B Elementi principali	#ED893E Elementi secondari	#000000 Testi principali	#7F7F82 Testi secondari
Altro			
#167F71 Azioni Positive	#FF8A06 Warning	#FF5959 Prestare attenzione	#FFFFFF Background

DIETIDEALS24

02 TIPOGRAFIA - POPPINS

- Titolo: **semibold 30**
- Testo Principale: **semibold 20-24**
- Sottotitolo: **bold 14-16**
- Slogan: **medium 20-24**
- Scritte secondarie: **semibold 16**
- Bottoni: **medium 18**
- Form: **semibold 14**
- Intestazione Pagina: **semibold 22**

Figura 11: Studio UI

Il front-end dell'app mobile presenta:

- **Palette di colori:** #54ACB4, #75C4CC, #000000, #FFFFFF, #7F7F82, #FF8A06, #167F71, #ED893E, #DB591B, #D93737
- **Stile Tipografico:** Poppins e Roboto
- **Elementi Grafici:** Per garantire uniformità e migliorare l'usabilità, abbiamo adottato una specifica paletta di colori per gli elementi dell'app:
 - Per i bottoni principali, abbiamo utilizzato il colore #54ACB4 con testo in #FFFFFF; per i bottoni secondari il colore #FFFFFF con testo in #000000.
 - I testi selezionabili/cliccabili sono evidenziati con il colore #75C4CC se cliccati; con il colore #000000 se non cliccati.
 - Le sezioni della schermata che devono risaltare sono contrassegnate dal colore #FF8A06.
 - I testi più rilevanti e significativi sono contrassegnati con il colore #000000; mentre i testi secondari con il #7F7F82.
 - Per i testi informativi abbiamo utilizzato il colore #54ACB4; mentre per i warning il colore #FF8A06.
 - I feedback positivi sono con il colore #167F71; mentre quelli di errore con il colore #D93737.
 - Sono stati inseriti dei box rettangolari con effetto di sopraelevamento per essere cliccati.

Le dimensioni dei caratteri sono state selezionate attentamente per migliorare l'usabilità:

- I titoli utilizzano il carattere Poppins Semibold con dimensioni comprese tra 20 e 24 punti.
 - I sottotitoli sono formattati con il carattere Poppins Medium, dimensioni di 12-16 punti.
 - Le scritte minori sono realizzate con il carattere Poppins Medium, dimensioni di 12 punti.
 - Per i bottoni, abbiamo scelto il carattere Poppins con una dimensione di 18 punti.
- **Layout e Organizzazione:** Nelle schermate principali dell'applicazione, sono presenti le seguenti caratteristiche:



- Un menu a scelta rapida nella parte inferiore della schermata, contenente le shortcut principali.
- Ogni schermata secondaria è dotata di un pulsante per tornare alla schermata principale da cui si è arrivati.
- Le schermate sono sviluppate principalmente nella parte centrale e prevedono uno scorrimento verticale per accedere a contenuti aggiuntivi.
- I bottoni dell’interfaccia sono collocati nella parte inferiore della schermata, occupando l’intero spazio centralmente.

Inoltre, le sezioni di scelta nelle schermate includono una barra di ricerca nella parte superiore per agevolare la selezione delle opzioni desiderate.

- **Uso di Bottoni e Elementi Interattivi:** I bottoni rappresentano l’elemento principale di interazione nel progetto e presentano le seguenti caratteristiche:
 - Hanno una forma rettangolare con dimensioni di 245x60 pixel.
 - I bordi dei bottoni sono arrotondati con un raggio di 20 pixel per un aspetto più morbido.
 - Oltre al testo descrittivo, alcuni bottoni includono un’icona per migliorare l’interazione e la comprensione da parte dell’utente.

Altri elementi interattivi includono i testi cliccabili, che seguono le seguenti specifiche:

- Il colore del testo cliccabile è #000000 quando non è stato cliccato.
- Una volta cliccato, il colore del testo cambia in #FFFFFF per indicare l’interazione attiva con l’elemento e ha uno sfondo #75C4CC.

Da notare che gli unici campi in cui l’utente è tenuto a inserire del testo sono quelli relativi alla funzione di registrazione, ricerca e ai campi di inserimento/creazione delle aste.

- **Stile delle Forme e dei Contenitori:** In linea generale, le schermate del progetto seguono le seguenti caratteristiche:
 - Non presentano immagini di sfondo, mantenendo uno sfondo uniforme.
 - Lo sfondo ha un colore principale di sfondo #FFFFFF per un aspetto pulito e coerente.
 - Le dimensioni delle schermate sono ottimizzate per la visualizzazione su un dispositivo Android Large.
- **Consistenza del Design:** Tutte le schermate dell’applicazione condividono elementi riutilizzati al fine di garantire uniformità e coerenza del design. Questi elementi includono:
 - Bottoni
 - Menu a Scelta Rapida
 - Frecce di Navigazione
 - Box delle Informazioni sulle Aste.
- **Responsive Design:** Nella fase iniziale del progetto su Figma, non è stata implementata la progettazione responsive. Tuttavia, ci proponiamo di sviluppare la responsività dell’app durante la fase di progettazione per Android.
- **Iconografia e Simboli:** Nel progetto, non sono state utilizzate icone particolari o simboli complessi. Tutte le icone presenti sono state scelte per la loro natura autoesplicativa, al fine di garantire una chiara comprensione da parte degli utenti senza la necessità di spiegazioni aggiuntive.



- **Scelte di Navigazione:** L'utente ha diverse opzioni per navigare all'interno dell'applicazione:
 - Può accedere alle schermate principali utilizzando il menu a scelta rapida posizionato nella parte inferiore dell'interfaccia. Questo menu consente un accesso rapido alle funzionalità chiave dell'app.
 - Inoltre, l'utente può tornare indietro da qualsiasi schermata facendo clic sulla freccia di navigazione appositamente posizionata per facilitare il ritorno alla schermata precedente.
- **Feedback Utente:** Nel progetto Figma attuale, il feedback con l'utente è limitato a:
 - Il cambio di colore per i testi cliccati per indicare l'interazione attiva.
 - L'apparizione di popup in caso di errore/scelta critica per fornire informazioni aggiuntive.

Tuttavia, nell'ambito del progetto finale, ci proponiamo di migliorare l'esperienza dell'utente aggiungendo ulteriori elementi di feedback, tra cui:

- Feedback tattile per la pressione sui buttoni, per indicare in modo più evidente l'azione.
- Feedback visivo durante l'inserimento di testo per confermare l'input dell'utente.

MockUp

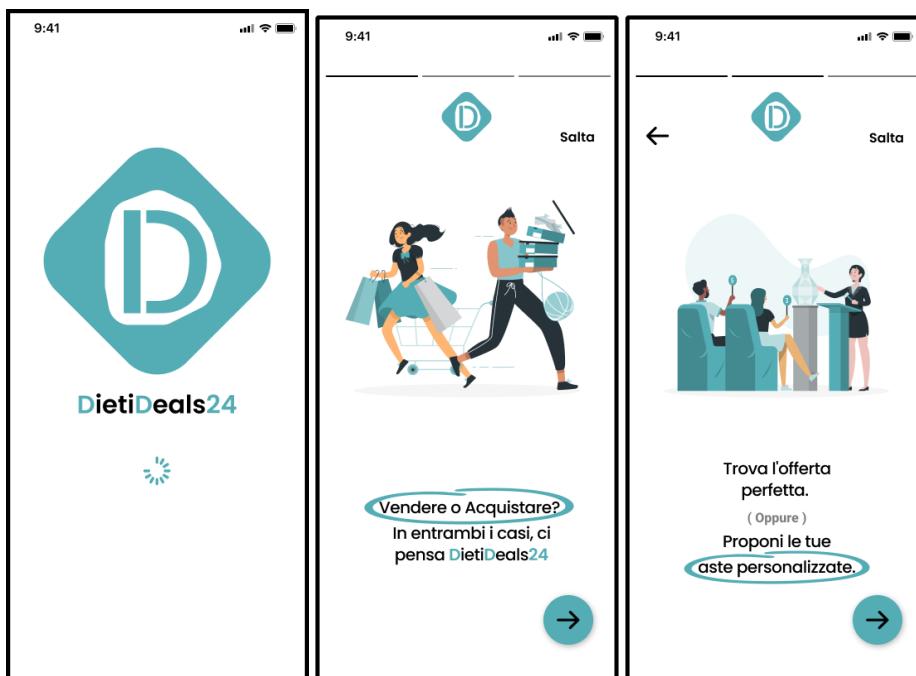


Figura 12: Schermata apertura applicazione Figura 13: Schermata informativa 1 Figura 14: Schermata informativa 2

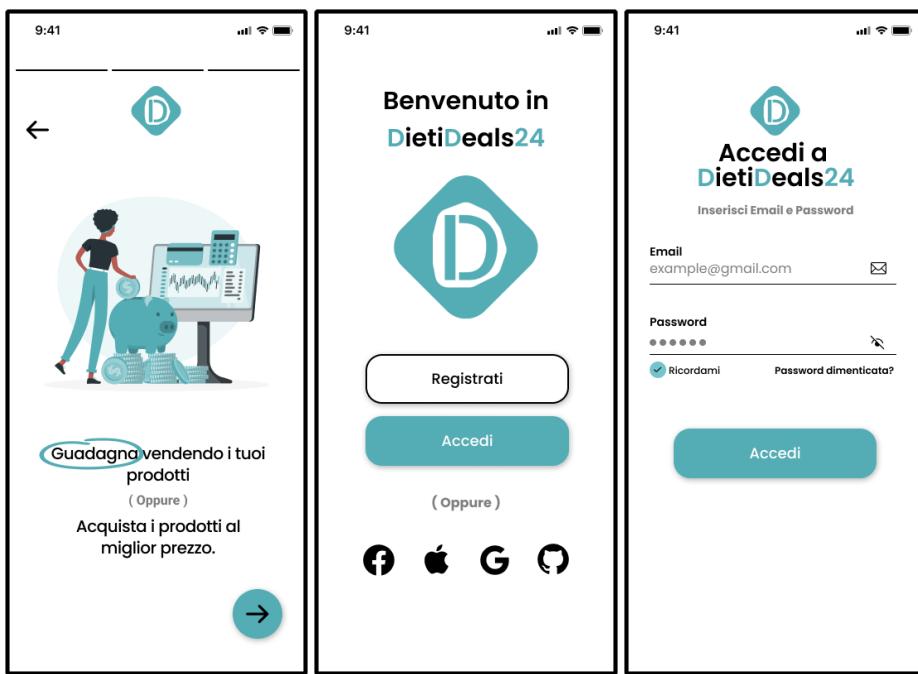


Figura 15: Schermata informativa 3

Figura 16: Schermata di benvenuto

Figura 17: Schermata di accesso

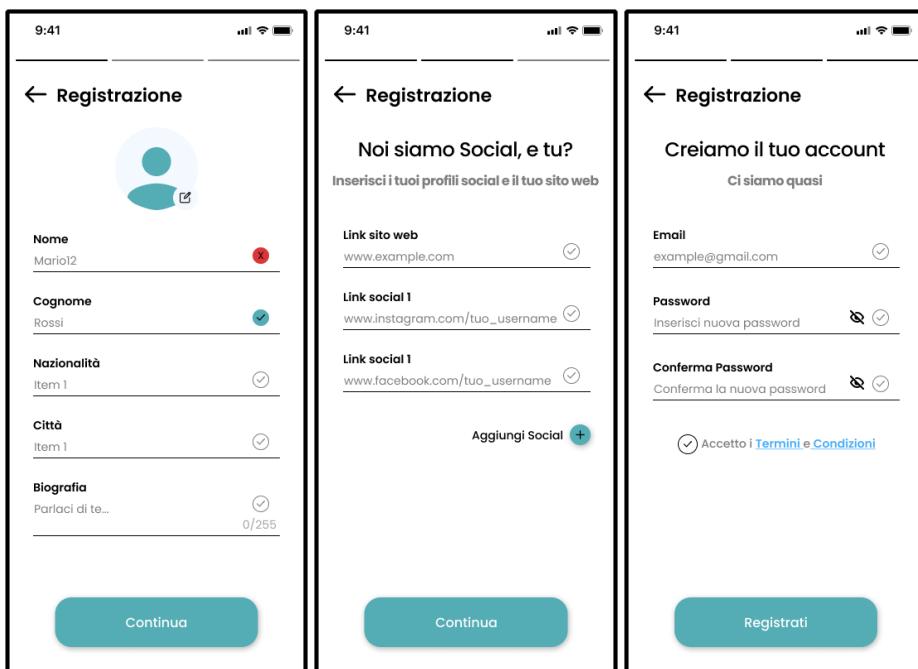


Figura 18: Schermata di registrazione 1

Figura 19: Schermata di registrazione 2

Figura 20: Schermata di registrazione 3

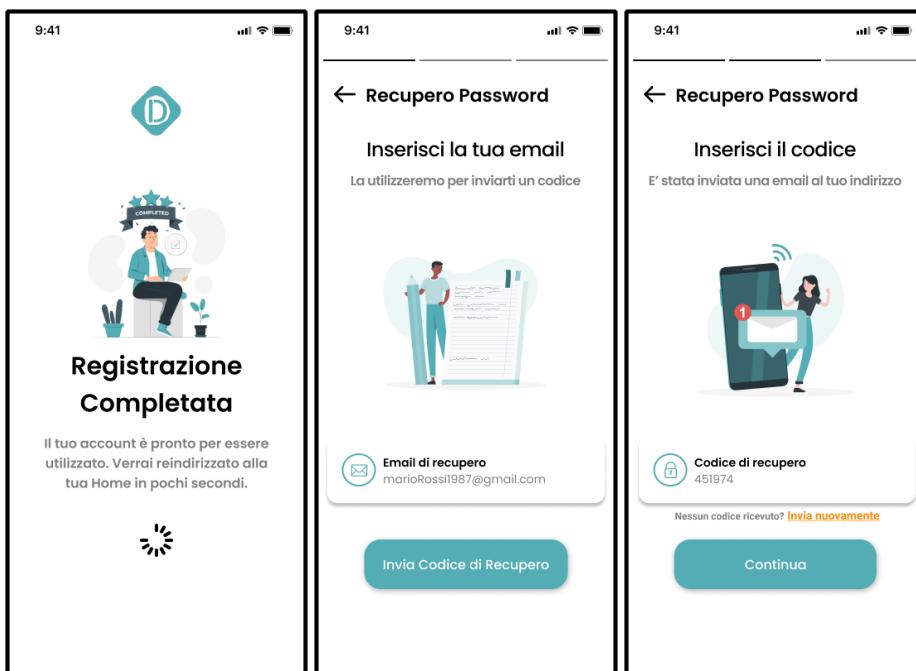
Figura 21: Schermata re-
gistrazione completataFigura 22: Schermata di
recupero password 1Figura 23: Schermata di
recupero password 2



Figura 24: Schermata di Figura 25: Schermata ter-
recupero password 3

Figura 26: Schermata ho-
mini e condizioni

**Figura 26: Schermata ho-
me acquirente**

Conditions

- Gli utenti devono registrarsi fornendo dati autentici e mantenersi responsabili della sicurezza del loro account.
- Solo gli utenti maggiorenni o che hanno l'età legale per stipulare contratti nel loro paese di residenza, possono partecipare alle aste.
- Presentando un'offerta, l'utente si impegna ad acquistare l'oggetto all'offerta proposta, se questa risulta essere la vincente.
- I vincitori dell'asta sono tenuti a effettuare il pagamento entro un termine specificato dall'annuncio dell'asta.
- La responsabilità della consegna dell'oggetto passa al venditore, a meno che non sia specificato diversamente.
- Gli utenti hanno diritto a un periodo di recesso, se applicabile legalmente nel loro paese di residenza.
- È vietato manipolare i prezzi delle aste, interferire con le transazioni degli altri utenti o pubblicare contenuti ingannevoli.

Termini

- L'uso dell'app implica l'accettazione incondizionata dei presenti termini e condizioni.
- L'azienda si riserva il diritto di modificare i termini e condizioni in qualsiasi momento.
- L'azienda si impegna a proteggere la privacy degli utenti e a utilizzare i dati personali in conformità con la politica sulla privacy pubblicata.
- Tutti i contenuti dell'app, inclusi testi, grafiche e codici, sono protetti da diritti di proprietà intellettuale.
- L'azienda non sarà responsabile per danni diretti, indiretti, incidentali, speciali o conseguenziali derivanti dall'uso o dall'impossibilità di usare l'app.
- I termini saranno governati e interpretati in conformità con le leggi del paese in cui l'azienda ha la sua sede principale, senza dare effetto a qualsiasi principio di conflitto di leggi.
- Qualsiasi controversia relativa all'uso dell'app sarà risolta tramite arbitrato o in tribunale, a seconda di quanto stabilito dall'azienda.

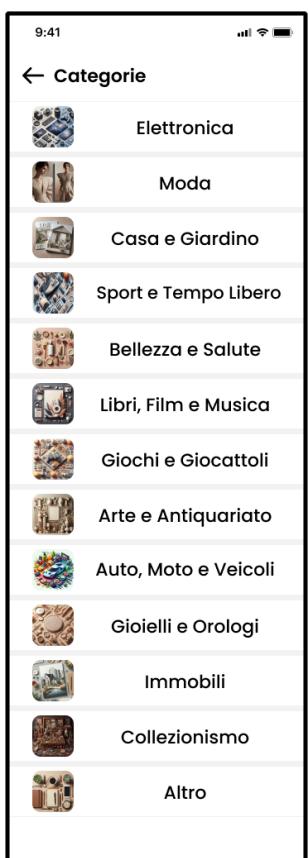


Figura 27: Schermata ca-

tegorie

Figura 28: Schermata aste
silenzioseFigura 29: Schermata aste
ribasso

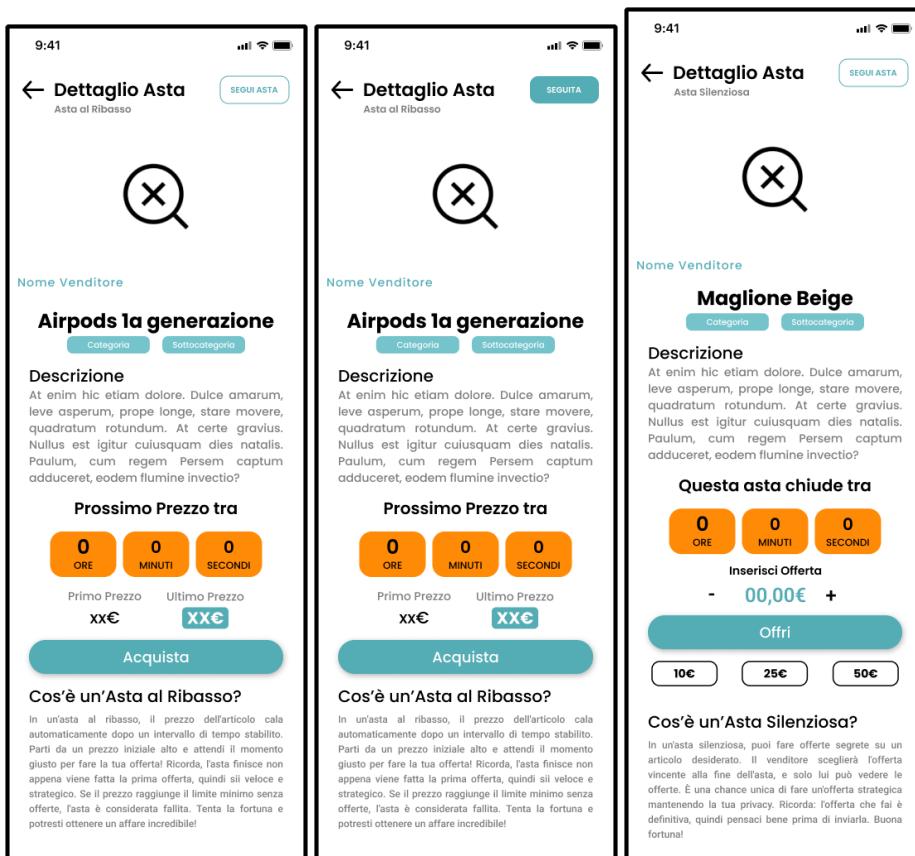


Figura 30: Schermata dettaglio asta ribasso

Figura 31: Schermata dettaglio asta ribasso2

Figura 32: Schermata dettaglio asta silenziosa

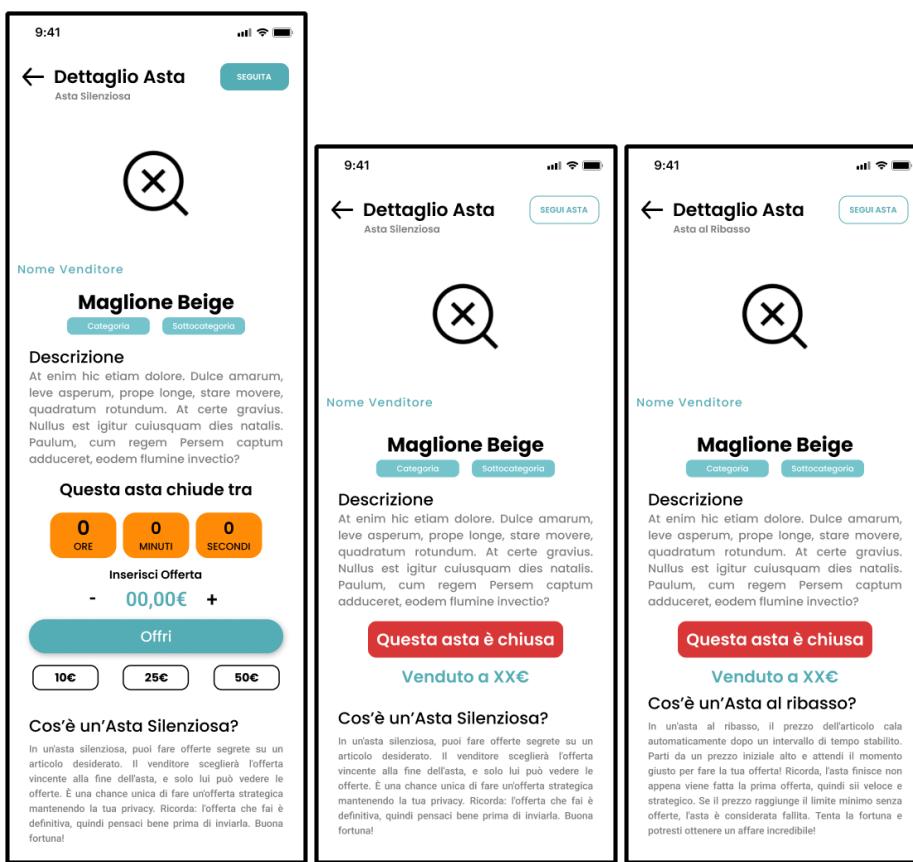


Figura 33: Schermata dettaglio asta silenziosa2

Figura 34: Schermata dettaglio asta silenziosa3

Figura 35: Schermata dettaglio asta ribasso3

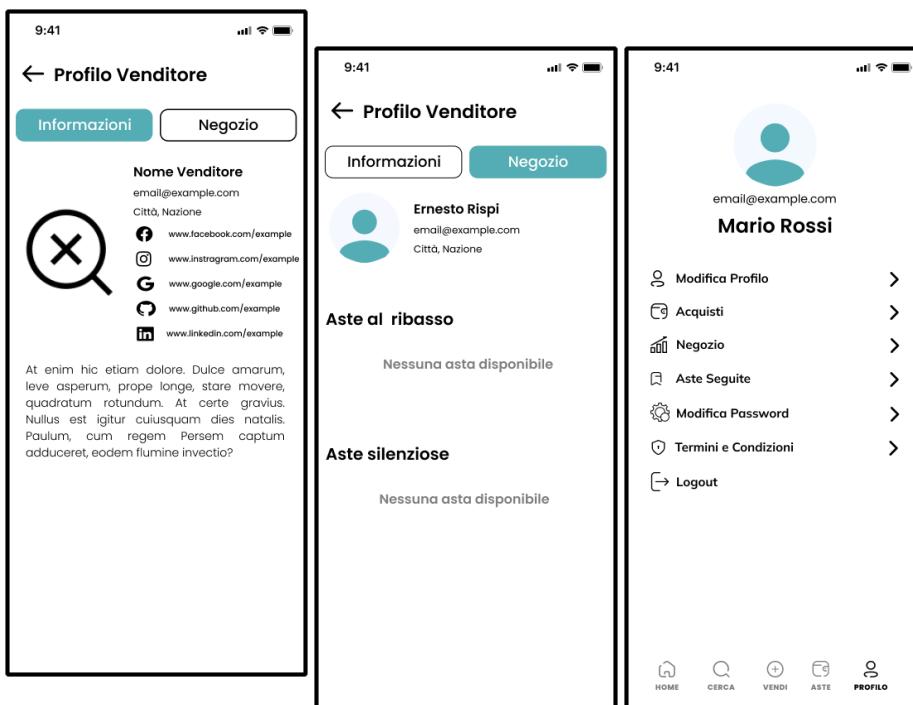


Figura 36: Schermata profilo venditore informazioni

Figura 37: Schermata profilo venditore aste al ribasso

Figura 38: Schermata profilo venditore negozio

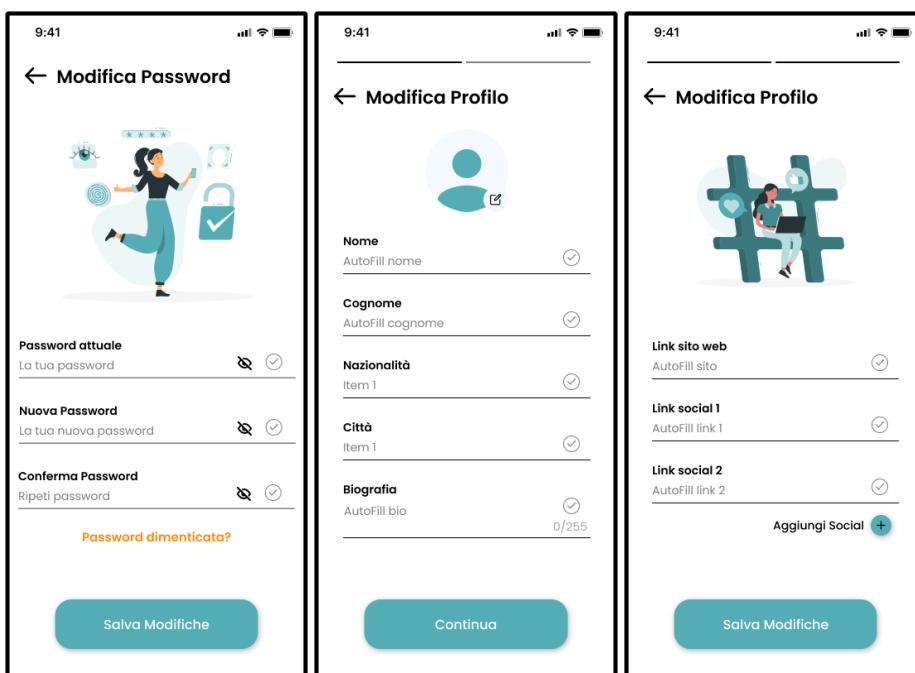


Figura 39: Schermata modifica credenziali

Figura 40: Schermata modifica profilo 1

Figura 41: Schermata modifica profilo 2

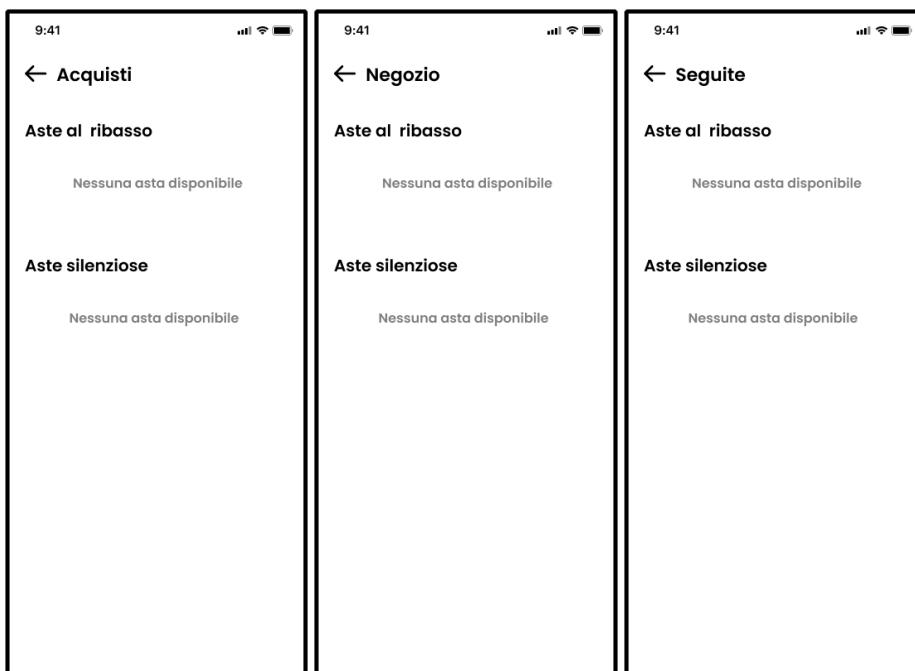


Figura 42: Schermata acquisti

Figura 43: vendite

Figura 44: Schermata seguite

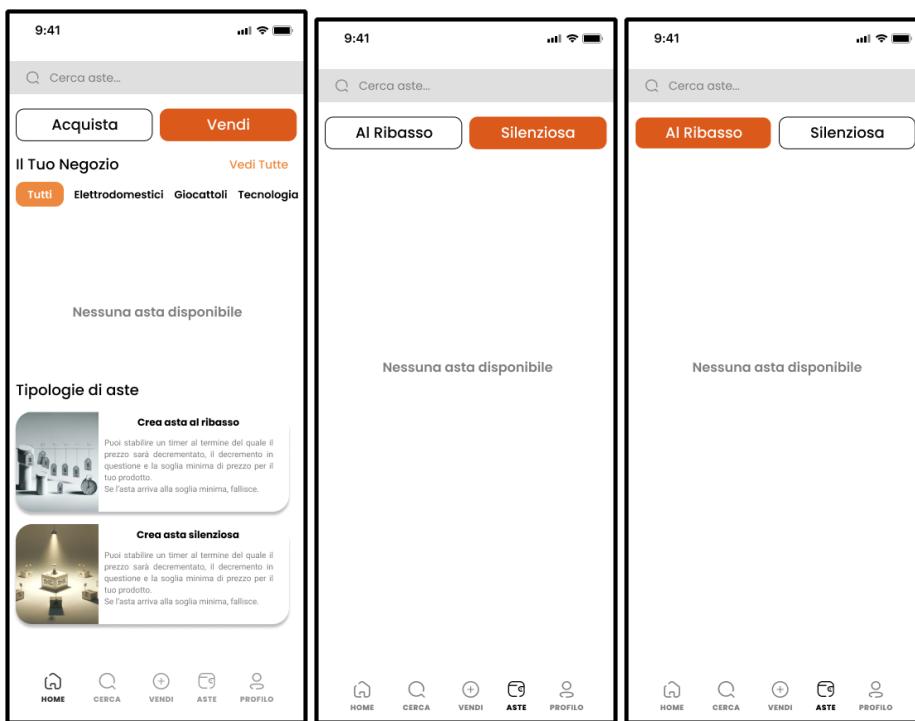


Figura 45: Schermata home venditore
Figura 46: Schermata aste
Figura 47: Schermata aste

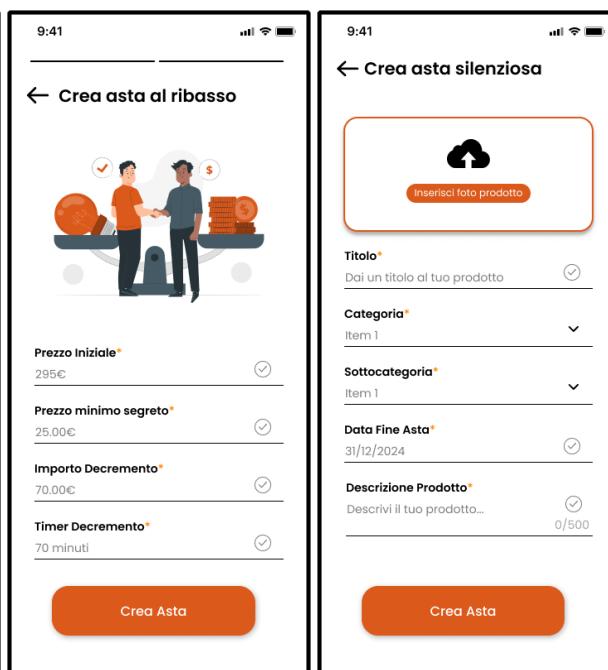


Figura 48: Schermata crea asta ribasso
Figura 49: Schermata crea asta ribasso2
Figura 50: Schermata crea asta silenziosa



Figura 51: Schermata asta ribasso



Figura 52: Schermata asta silenziosa



Figura 53: Schermata asta silenziosa chiusa



The figure consists of three side-by-side screenshots of a mobile application interface:

- Figura 54: Schermata asta chiusa** (Left): A detail page for a closed auction titled "Dettaglio Asta Asta al Ribasso". It shows a large "X" icon, a search bar, and a section for the product name. Below is a "Descrizione" block with Latin placeholder text. A green button says "Questa asta è chiusa", followed by "Venduto a XX€" and "Prezzo iniziale: XX€". A section titled "Cos'è un'Asta al Ribasso?" provides a brief explanation.
- Figura 55: Schermata cerca** (Middle): A search results page for "Aste al ribasso". It displays two sections: "Aste al ribasso" (empty) and "Aste silenziose" (empty). At the bottom are navigation icons: HOME, CERCA, VENDI, ASTE, and PROFILO.
- Figura 56: Schermata filtri** (Right): A filter settings screen titled "Filtri". It includes sections for "Tipologia" (with "Asta Silenziosa" checked), "Ordina per" (dropdown), "Prezzo" (range slider from 0€ to 1000€), "Data fine asta" (set to 23/12/2025), and "Categoria" (checkboxes for Abbigliamento, Elettrodomestici, Bellezza, Giocattoli, and Altro, with "Tutti" also available). A blue "Salva Filtri" button is at the bottom.

Figura 54: Schermata asta chiusa
Figura 55: Schermata cerca

Figura 56: Schermata filtri

9:41 Vendi - Crea asta

Vendi - Crea asta

Tipologia Asta*

Al Ribasso

Inserisci foto prodotto

Titolo*
Dai un titolo al tuo prodotto

Categoria*
Item 1

Sottocategoria*
Item 1

Descrizione Prodotto*
Descrivi il tuo prodotto... 0/500

Prezzo Iniziale*
295.00€

Prezzo minimo segreto*
25.00€

Importo Decremento*
70.00€

Timer Decremento*
70 minuti

Crea Asta

9:41 Vendi - Crea asta

Vendi - Crea asta

Tipologia Asta*

Silenziosa

Inserisci foto prodotto

Titolo*
Dai un titolo al tuo prodotto

Categoria*
Item 1

Sottocategoria*
Item 1

Data Fine Asta*
31/12/2024

Descrizione Prodotto*
Descrivi il tuo prodotto... 0/500

Crea Asta

Figura 57: Schermata Vendi

Figura 58: Schermata Vendi ribasso

Figura 59: Schermata Vendi silenziosa

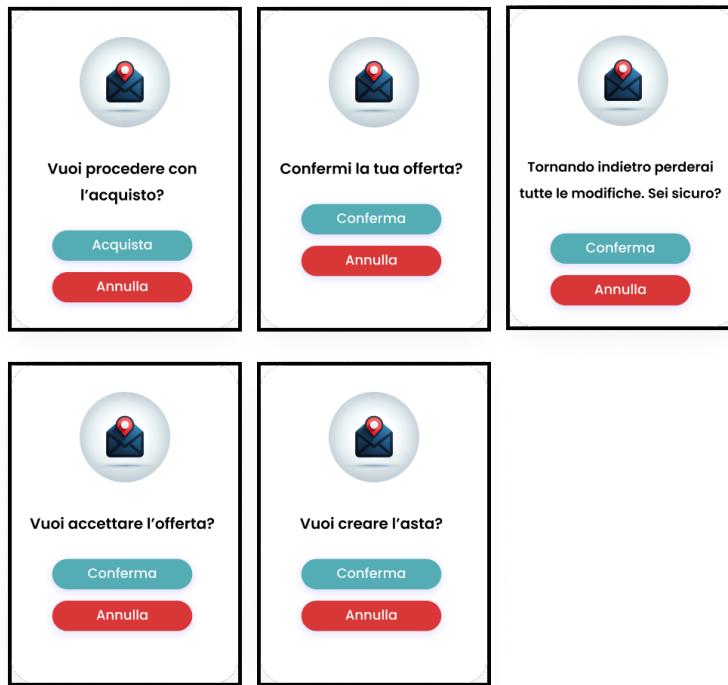


Figura 60: Schermata pop-up



Figura 61: Schermata Schede generiche



Figura 62: Schermata Schede stato asta

1.8 Valutazione dell'usabilità a priori

Una delle sfide principali nello sviluppo di software è garantire che l'interfaccia utente (UI) sia intuitiva, efficace e piacevole da usare. Qui entra in gioco il concetto di "usabilità", che si riferisce alla facilità con cui gli utenti possono utilizzare un prodotto per raggiungere obiettivi specifici.

L'usabilità a priori si riferisce alla valutazione dell'usabilità di un prodotto prima che esso venga effettivamente costruito o implementato. Questo tipo di valutazione è fondamentale per identificare e correggere eventuali problemi di usabilità in una fase precoce del processo di sviluppo, risparmiando tempo e risorse.

Valutazione euristica: ISO 9241-110

Nell'ambito della progettazione e valutazione delle interfacce utente, la norma ISO 9241 rappresenta un pilastro fondamentale. La parte 110 di questa norma, specificamente, si concentra sui principi di dialogo tra l'utente e il sistema. Questi principi sono essenziali per garantire un'interazione efficace, efficiente e



soddisfacente tra l'utente e il sistema in questione. La valutazione euristica, come tecnica di ispezione, utilizza un insieme di principi o "euristiche" per identificare potenziali problemi di usabilità. Applicando le euristiche delineate nell'ISO 9241-110, è possibile condurre una valutazione mirata che tenga conto dei principi di dialogo fondamentali per un'interazione ottimale.

In questa sottosezione, esploreremo i principi di dialogo dell'ISO 9241-110 e discuteremo di come possono essere utilizzati per condurre una valutazione euristica efficace.

Principio	Valutazione
Adeguatezza al compito	L'interfaccia soddisfa tutti i requisiti funzionali e supporta l'utente nel completamento delle sue attività in modo efficiente.
Auto descrizione	Ogni elemento dell'interfaccia è chiaro e comprensibile, fornendo un feedback adeguato per guidare l'utente.
Conformità alle aspettative	L'interfaccia segue le convenzioni standard di design e rispetta le aspettative degli utenti basate sulle loro esperienze precedenti.
Controllabilità	Gli utenti hanno pieno controllo delle loro azioni, con la capacità di annullare o ripetere le azioni e di modificare le impostazioni secondo le loro preferenze.
Tolleranza verso gli errori	Il sistema previene errori comuni e fornisce messaggi di errore chiari e costruttivi che aiutano gli utenti a recuperare dagli errori.
Design minimalista ed estetico	L'interfaccia è visivamente attraente e priva di elementi non necessari che potrebbero distrarre o confondere l'utente.
Adeguatezza all'apprendimento	Nuovi utenti possono imparare rapidamente come usare l'interfaccia, e c'è un supporto adeguato per gli utenti a tutti i livelli di esperienza.

Tabella 7: Tabella di valutazione

Definiremo ora i Test che verranno condotti e i punteggi che verranno assegnati in base agli eventi che vogliamo monitorare e infine spiegheremo delle funzioni atte a calcolare vari aspetti, in ambito di usabilità, della nostra applicazione.

Nelle tabelle della sezione 1.9 e 1.10, verranno utilizzate le seguenti sigle:

- **S** (Successo, 1): indica il completamento riuscito del compito
- **P** (Successo Parziale, 0,5): indica il completamento parziale del compito
- **F** (Fallimento, 0): indica il mancato completamento del compito
- **CC** (Click Corretti): rappresenta il numero di click eseguiti correttamente
- **CE** (Click Eseguiti): rappresenta il numero di click eseguiti (totale)
- **TI** (Tempo Impiegato): indica il tempo impiegato per completare il compito
- **E** (Esito): rappresenta l'esito del compito



Inoltre, i dati raccolti verranno utilizzati per formulare i seguenti risultati:

- **Tasso di successo (Success Rate):** Questa metrica misura la percentuale di compiti che gli utenti completano con successo. Un tasso di successo più alto indica un'usabilità migliore.
Formula: $(\text{Numero di compiti completati con successo}) / (\text{Numero totale di compiti tentati})$
- **Tempo impiegato per il completamento (Time on Task):** Questa metrica misura quanto tempo gli utenti impiegano per completare compiti specifici. Un tempo più breve indica un'usabilità migliore.
Formula: Tempo medio impiegato per il completamento di un compito
- **Errori utente (User Errors):** Questa metrica misura il numero di errori commessi dagli utenti durante l'utilizzo dell'interfaccia. Un numero più basso di errori indica un'usabilità migliore.
Formula: $(\text{Numero totale di errori commessi dagli utenti}) / (\text{Numero totale di compiti tentati})$
- **Soddisfazione dell'utente (User Satisfaction):** Questa metrica valuta il grado di soddisfazione degli utenti nell'utilizzo dell'interfaccia. È spesso misurata su una scala, ad esempio, da 1 a 5 o da 1 a 7, dove punteggi più alti indicano una maggiore soddisfazione.
Formula: Misurata tramite sondaggi, questionari o scale di valutazione
- **Scalabilità dell'interfaccia (Scalability):** Questa metrica misura quanto delle funzionalità dell'interfaccia sono state effettivamente utilizzate dagli utenti. Una maggiore scalabilità indica un'usabilità migliore.
Formula: $(\text{Numero di funzionalità o schermate utilizzate}) / (\text{Numero totale di funzionalità o schermate disponibili})$
- **Efficienza (Efficiency):** Questa metrica combina il tasso di successo e il tempo impiegato per misurare quanto velocemente gli utenti riescono a completare i compiti.
Formula: $(\text{Compiti completati con successo}) / (\text{Tempo totale impiegato})$
- **Facilità di apprendimento (Learnability):** Questa metrica esamina quanto sia facile per gli utenti imparare a utilizzare l'interfaccia. Può essere misurata osservando quanto tempo impiegano gli utenti per imparare a svolgere compiti di base.
Formula: Misurata attraverso test di apprendimento o valutazioni dell'utente. Nel nostro caso abbiamo assegnato un punteggio di 1 se il valore si scosta di $\pm 5s$ dal valore atteso; un punteggio di 0.5 se il valore si scosta di $\pm 10s$ dal valore atteso; un punteggio di 0 se il valore si scosta dal valore atteso.

1.9 Test di usabilità: prime prove

I test sono organizzati in tre categorie principali: funzionalità generali, funzionalità dedicate agli acquirenti e funzionalità specifiche per i venditori. Questa categorizzazione è riflessa dettagliatamente nelle tabelle 8, 9 e 10. In queste tabelle, sono elencati i compiti chiave che un utente dovrebbe essere in grado di eseguire utilizzando l'applicazione, insieme ai dati che ci aspettiamo di raccogliere da un utente che ha familiarità con il software.



COMPITO	COMPITI GENERALI	CLICK STIMATI	TEMPO STIMATO(s)
1	Registrazione	20	90
2	Accesso	7	28
3	Recupero password	11	58
4	Personalizzazione del profilo	6	38
5	Modifica Password	10	50
6	Visualizza acquisti	6	38
7	Visualizza vendite	6	38
8	Visualizza aste seguite	6	38

Tabella 8: Tabella Compiti Generali

COMPITO	COMPITI ACQUIRENTE	CLICK STIMATI	TEMPO STIMATO(s)
9	Presenta offerta asta silenziosa	11	50
10	Acquista da asta al ribasso	10	50
11	Visualizza aste	8	32
12	Ricerca una specifica asta	9	33
13	Visualizza profilo venditore	10	39

Tabella 9: Tabella Compiti Acquirente

COMPITO	COMPITI VENDITORE	CLICK STIMATI	TEMPO STIMATO(s)
14	Crea asta al ribasso	19	98
15	Crea asta silenziosa	16	50
16	Visualizza aste create	9	35

Tabella 10: Tabella Compiti Venditore



Calcolo Usabilità

Utente	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 1	20	21	100 s	1	20	26	140 s	0.5	20	20	88 s	1	20	23	87 s	0.5
Compito 2	7	9	34 s	1	7	10	43 s	1	7	7	27 s	1	7	8	31 s	1
Compito 3	11	12	75 s	0.5	11	14	90 s	0.5	11	11	60 s	1	11	11	80 s	1
Compito 4	6	7	39 s	1	6	8	46 s	0.5	6	6	35 s	1	6	7	37 s	1
Compito 5	10	12	50 s	1	10	11	105 s	0.5	10	10	50 s	1	10	10	57 s	1
Compito 6	6	6	38 s	1	6	6	40 s	1	6	6	38 s	1	6	6	36 s	1
Compito 7	6	6	38 s	1	6	7	44 s	1	6	6	38 s	1	6	6	37 s	1
Compito 8	6	7	40 s	1	6	6	40 s	1	6	6	38 s	1	6	6	36 s	1

Tabella 11: Calcolo Usabilità Compiti Utente

Acquirente	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 9	11	11	48 s	1	11	15	60 s	0.5	11	11	45 s	1	11	12	49 s	1
Compito 10	10	10	47 s	1	10	14	58 s	0.5	10	10	45 s	1	10	10	47 s	1
Compito 11	8	8	30 s	1	8	12	40 s	1	8	8	28 s	1	8	9	34 s	1
Compito 12	9	11	35 s	0.5	9	13	45 s	0.5	9	9	30 s	1	9	9	31 s	1
Compito 13	10	10	40 s	1	10	14	50 s	0.5	10	10	30 s	1	10	10	37 s	1

Tabella 12: Calcolo Usabilità Compiti Aqcuirente



Venditore	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 14	19	21	100 s	1	19	24	120 s	0	19	19	94 s	1	19	19	99 s	1
Compito 15	16	16	53 s	1	16	19	72 s	0.5	16	16	48 s	1	16	17	50 s	1
Compito 16	9	11	41 s	0.5	9	9	47 s	1	9	9	35 s	1	9	9	36 s	1

Tabella 13: Calcolo Usabilità Compiti Venditore

In base ai dati raccolti, abbiamo elaborato i seguenti risultati:

	Success Rate	Time On Task	Users Error	Efficiency	Learnability
MARIA	13	50.5	0.82	0.26	0.5
SARA	6	65	2.75	0.09	0
GIOVANNI	16	46.19	0	0.35	1
LUCA	15	49	0.5	0.31	1
ASPETTATIVE	14	45.94	2	0.30	0.75
MEDIA	12.5	52.67	1.02	0.25	0.625

Tabella 14: Calcolo Usabilità - Risultati totali

1.10 Test di usabilità: prove dopo i miglioramenti

Durante i test di usabilità del prototipo, e grazie all'elaborazione dei risultati ottenuti e alle segnalazioni riportate dai tester, sono state riscontrate le seguenti problematiche:

- **COLORI:** Tre tester su quattro hanno riscontrato difficoltà nel percepire le sensazioni appropriate dai colori utilizzati. In due casi, i colori principali hanno evocato sensazioni di disagio e disorientamento.
- **TIPOGRAFIA:** La differenziazione di font nelle varie schermate ha creato confusione tra i tester.
- **INFORMAZIONI ECCESSIVE:** È stato evidenziato che molti box di introduzione contenevano troppi dettagli, risultando eccessivamente complessi e confusionari.
- **FOCUS ERRATO:** Due tester su quattro hanno avuto difficoltà a comprendere immediatamente il focus di alcune schermate, a causa dell'utilizzo di colori non adeguati o poco intuitivi.
- **FEEDBACK:** In alcuni casi, i tester hanno espresso insoddisfazione per l'assenza totale o parziale di feedback, sia percettivi (ad esempio, colori e testo) che visivi (come i popup).

In seguito a queste considerazioni, sono state apportate le seguenti modifiche:

- **COLORI:** È stata introdotta una nuova paletta di colori, che ha ricevuto l'approvazione di tutti i tester, trasmettendo sensazioni di rilassatezza, armonia e comfort.



- **TIPOGRAFIA:** È stato adottato un unico tipo di font (Poppins), con variazioni solo nella dimensione e, occasionalmente, nel colore.
- **INFORMAZIONI ECCESSIVE:** Le schermate sono state semplificate e le informazioni ridotte nei punti in cui erano superflue.
- **FOCUS ERRATO:** Sono stati introdotti colori specifici con significati ben definiti, permettendo a tutti i tester di comprenderne il senso.
- **FEEDBACK:** Sono stati aggiunti vari popup informativi e i colori ora forniscono un feedback che viene percepito correttamente dai tester.
- **ALTRO:** Sono stati implementati ulteriori miglioramenti per rendere l'esperienza utente più fluida e intuitiva. Questo è stato possibile riducendo il numero di schermate e gestendo diverse azioni all'interno di un'unica interfaccia. Gli utenti hanno dimostrato di poter imparare e padroneggiare le funzionalità dell'applicazione in un tempo ragionevolmente breve.

In seguito alle analisi effettuate, abbiamo aggiornato i valori attesi relativi all'utilizzo del software da parte di utenti esperti. Queste revisioni sono dettagliatamente riportate nelle tabelle 40, 41 e 42, dove sono elencati i compiti fondamentali insieme ai valori aggiornati, riflettendo così le modifiche recentemente implementate nell'applicazione.

COMPITO	COMPITI GENERALI	CLICK STIMATI	TEMPO STIMATO(s)
1	Registrazione	19	80
2	Accesso	7	25
3	Recupero password	11	70
4	Personalizzazione del profilo	6	28
5	Modifica Password	10	52
6	Visualizza acquisti	6	35
7	Visualizza vendite	6	35
8	Visualizza aste seguite	6	35

Tabella 15: Tabella Compiti Generali



COMPITO	COMPITI ACQUIRENTE	CLICK STIMATI	TEMPO STIMATO(s)
9	Presenta offerta asta silenziosa	11	45
10	Acquista da asta al ribasso	10	45
11	Visualizza aste	8	30
12	Ricerca una specifica asta	9	29
13	Visualizza profilo venditore	10	45

Tabella 16: Tabella Compiti Acquirente

COMPITO	COMPITI VENDITORE	CLICK STIMATI	TEMPO STIMATO(s)
14	Crea asta al ribasso	17	125
15	Crea asta silenziosa	15	50
16	Visualizza aste create	10	40

Tabella 17: Tabella Compiti Venditore

Calcolo usabilità

Utente	Maria				Sara				Giovanni				Luca						
	Compito	CC	CE	T	E	Compito	CC	CE	T	E	Compito	CC	CE	T	E	Compito	CC	CE	T
Compito 1	20	20	80 s	1		20	22	90 s	1		20	20	88 s	1		20	23	87 s	1
Compito 2	7	7	30 s	1		7	7	43 s	1		7	7	27 s	1		7	8	31 s	1
Compito 3	11	11	62 s	1		11	11	68 s	1		11	11	60 s	1		11	11	70 s	1
Compito 4	6	7	40 s	1		6	8	46 s	1		6	6	35 s	1		6	7	32 s	1
Compito 5	10	11	52 s	1		10	11	72 s	0.5		10	10	50 s	1		10	10	60 s	1
Compito 6	6	6	36 s	1		6	6	40 s	1		6	6	38 s	1		6	6	36 s	1
Compito 7	6	6	36 s	1		6	7	44 s	1		6	6	38 s	1		6	6	36 s	1
Compito 8	6	6	36 s	1		6	6	40 s	1		6	6	38 s	1		6	6	36 s	1

Tabella 18: Calcolo Usabilità UI Migliorata Compiti Utente



Acquirente	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 9	11	11	41 s	1	11	12	50 s	0.5	11	11	45 s	1	11	12	44 s	1
Compito 10	10	10	45 s	1	10	12	40 s	1	10	10	45 s	1	10	10	51 s	1
Compito 11	8	8	37 s	1	8	8	40 s	1	8	8	28 s	1	8	9	34 s	1
Compito 12	9	9	30 s	1	9	10	45 s	1	9	9	30 s	1	9	9	31 s	1
Compito 13	10	10	32 s	1	10	10	37 s	0.5	10	10	30 s	1	10	10	37 s	1

Tabella 19: Calcolo Usabilità UI Migliorata Compiti Acquirente

Venditore	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 14	19	20	92 s	1	19	20	85 s	0.5	19	19	97 s	1	19	19	94 s	1
Compito 15	16	16	49 s	1	16	16	68 s	1	16	16	50 s	1	16	16	50 s	1
Compito 16	9	11	47 s	1	9	9	47 s	0.5	9	9	35 s	1	9	9	37 s	1

Tabella 20: Calcolo Usabilità UI Migliorata Compiti Venditore

In base ai dati raccolti, abbiamo elaborato i seguenti risultati:

	Success Rate	Time ON Task	Users Error	Efficiency	Learnability
MARIA	16	46.56	0.29	0.34	1
SARA	13	53.44	2	0.24	1
GIOVANNI	16	45.88	0	0.35	1
LUCA	16	47.88	0.44	0.33	1
ASPETTATIVE	14	48.06	2	0.29	0.9
MEDIA	15.25	48.44	0.68	0.32	1

Tabella 21: Calcolo Usabilità UI Migliorata - Risultati totali

1.11 Considerazioni finali usabilità

L'approccio sistematico ai test di usabilità ha consentito di identificare e risolvere efficacemente i problemi di usabilità dell'applicazione. Le modifiche apportate hanno portato a miglioramenti tangibili, evidenziati dai risultati dei test post-modifica. Questo processo dimostra l'importanza di un ciclo iterativo di



valutazione e miglioramento nell'ambito dello sviluppo software, specialmente in termini di usabilità e esperienza utente. Le soluzioni implementate hanno reso l'applicazione non solo più funzionale ma anche più piacevole da usare, aumentando così la probabilità di una maggiore adozione e soddisfazione da parte degli utenti finali.

La differenza tra le due tabelle 14 e 46 è significativa e riflette l'impatto delle modifiche apportate all'interfaccia utente (UI) dell'applicazione. Ecco una comparazione dettagliata:

1. Tasso di Successo (Success Rate):

- Prima delle Modifiche: Il tasso di successo variava notevolmente tra i vari utenti, con una media complessiva di 12.5 compiti completati per persona.
- Dopo le Modifiche: Si nota un marcato miglioramento nel tasso di successo, con una media quasi perfetta di 15.25 compiti completati per persona. Questo indica che le modifiche hanno reso i compiti più facili da completare per la maggior parte degli utenti.

Miglioramento complessi del 18% circa.

2. Tempo Impiegato per il Completamento (Time on Task):

- Prima delle Modifiche: La media del tempo impiegato era di 52.67 secondi con un discostamento dalle aspettative di circa 7 secondi, suggerendo una certa inefficienza o difficoltà nell'eseguire i compiti.
- Dopo le Modifiche: Il tempo medio è diminuito a 48.44 secondi, indicando che gli utenti erano in grado di completare i compiti più rapidamente, un segnale di migliorata usabilità considerando che il valore di discosta dalle aspettative di 0.38 secondi.

Miglioramento dell'8% circa.

3. Errori degli Utenti (Users Error):

- Prima delle Modifiche: Il numero medio di errori per utente per ogni schermata era 1.02, indicando una certa frequenza di errori nell'utilizzo dell'applicazione.
- Dopo le Modifiche: Si è registrata una notevole riduzione degli errori, scendendo a una media di 0.68 errori per schermata per utente. Questo suggerisce che le modifiche hanno reso l'interfaccia più intuitiva e meno soggetta a confusione.

Miglioramento del 50% circa.

4. Efficienza (Efficiency):

- Prima delle Modifiche: L'efficienza media era di 0.25, indicando una certa efficacia ma con margine di miglioramento.
- Dopo le Modifiche: L'efficienza è aumentata a 0.32, il che indica che gli utenti erano in grado di completare i compiti con maggiore successo e in minor tempo.

Miglioramento del 21% circa.

5. Facilità di Apprendimento (Learnability):

- Prima delle Modifiche: La media era 0.75, suggerendo che alcuni utenti avevano difficoltà nell'apprendere l'utilizzo dell'interfaccia.
- Dopo le Modifiche: La facilità di apprendimento è aumentata a 0.9, indicando che le modifiche hanno reso l'interfaccia molto più facile da comprendere e usare per gli utenti.

**Miglioramento del 38% circa.**

In sintesi, la comparazione tra le due tabelle mostra che le modifiche apportate all'interfaccia utente hanno avuto un impatto positivo marcato su tutti gli aspetti della usabilità, migliorando significativamente l'esperienza complessiva degli utenti.



2 Specifica dei Requisiti

Questa sezione del documento si concentra sulla specifica dei requisiti del sistema. Qui verranno dettagliate le classi, gli oggetti e le relazioni di analisi attraverso un diagramma delle classi. Inoltre, verranno presentati i diagrammi di sequenza di analisi per due casi d'uso significativi al fine di comprendere meglio l'interazione tra gli attori e il sistema. Infine, verrà realizzata una prototipazione funzionale dell'interfaccia grafica utilizzando un diagramma di stato (statechart) al fine di visualizzare visivamente il comportamento del sistema.

2.1 Classi, oggetti e relazioni di analisi: Class Diagram

In questa sezione, approfondiremo la specifica dei requisiti attraverso un diagramma delle classi. Descriveremo le classi, gli oggetti e le relazioni chiave che costituiscono il nucleo del sistema. Questo diagramma fornisce una panoramica chiara della struttura del sistema e delle sue principali componenti.

Euristica Three-Object-Type

L'europea *Three-Object-Type* si riferisce a un modello architettonico nel design del software, particolarmente utile nella programmazione orientata agli oggetti. Questo modello distingue tra tre tipi fondamentali di oggetti: *Entità*, *Controllo* e *Confine*.

1. **Entità (Entity):** Gli oggetti di tipo *Entità* rappresentano i concetti fondamentali del dominio del problema, come dati di clienti o prodotti in un'applicazione commerciale. Sono caratterizzati dalla loro durata lunga e dalla loro indipendenza dalle specifiche funzioni dell'applicazione.
2. **Controllo (Control):** Gli oggetti di *Controllo* coordinano le attività e gestiscono la logica di business. Funzionano come intermediari tra gli oggetti *Entità* e *Confine*, dirigendo il flusso delle operazioni e assicurando che le azioni siano eseguite correttamente.
3. **Confine (Boundary):** Gli oggetti di tipo *Confine* gestiscono l'interazione tra il sistema e il mondo esterno. Questo include la gestione delle interfacce utente, delle comunicazioni di rete e dell'interazione con database o altri sistemi.

Questo modello promuove una chiara separazione delle responsabilità tra diverse parti del sistema, facilitando la manutenzione e l'espansione del software.

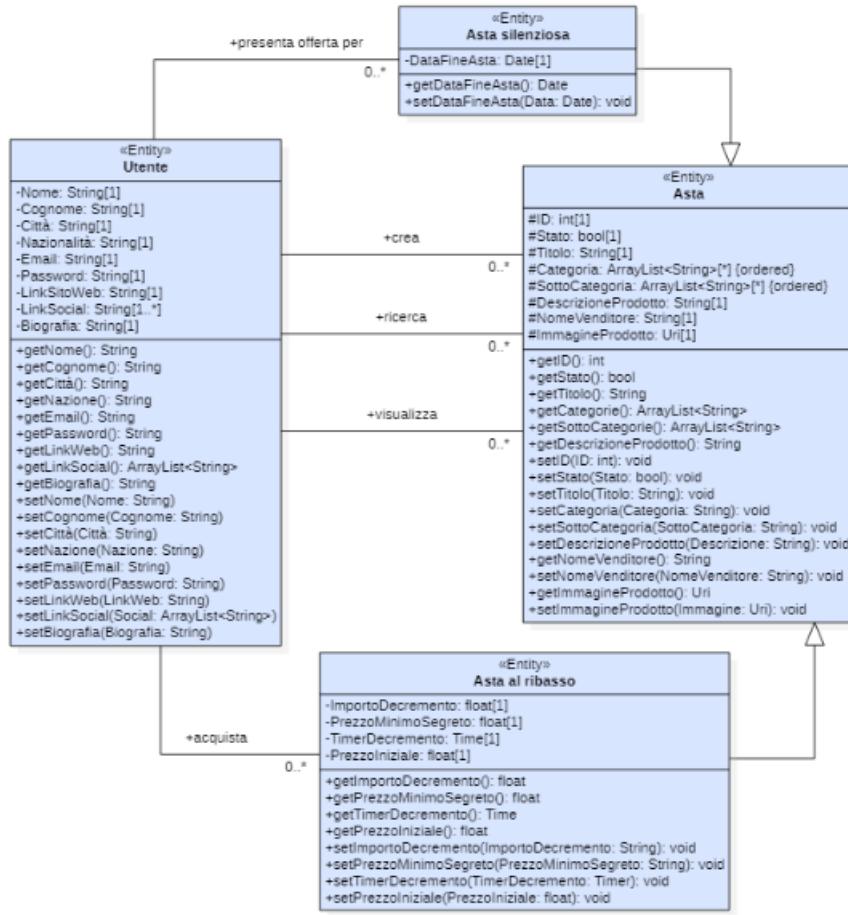


Figura 63: ClassDiagram - Entità software

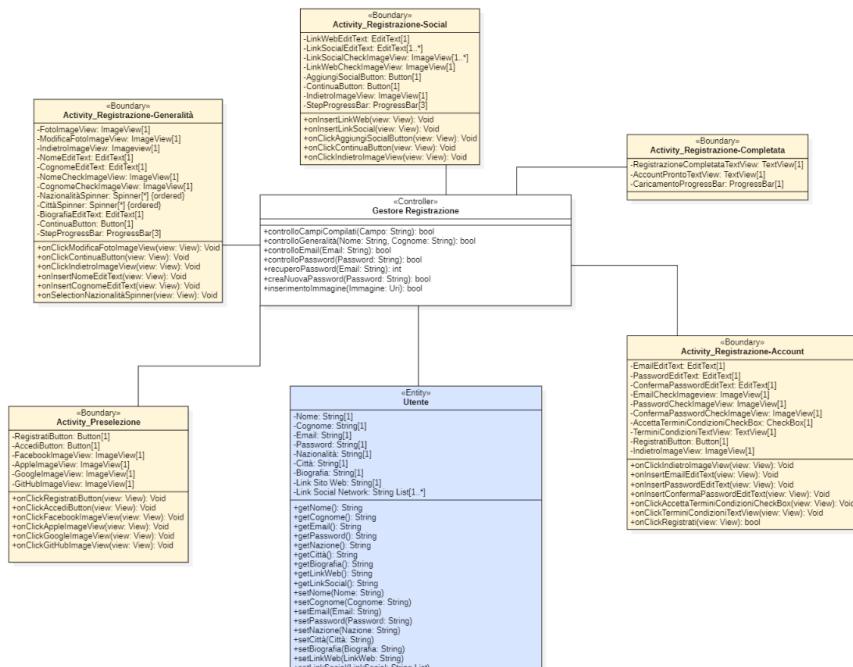


Figura 64: ClassDiagram - Registrazione

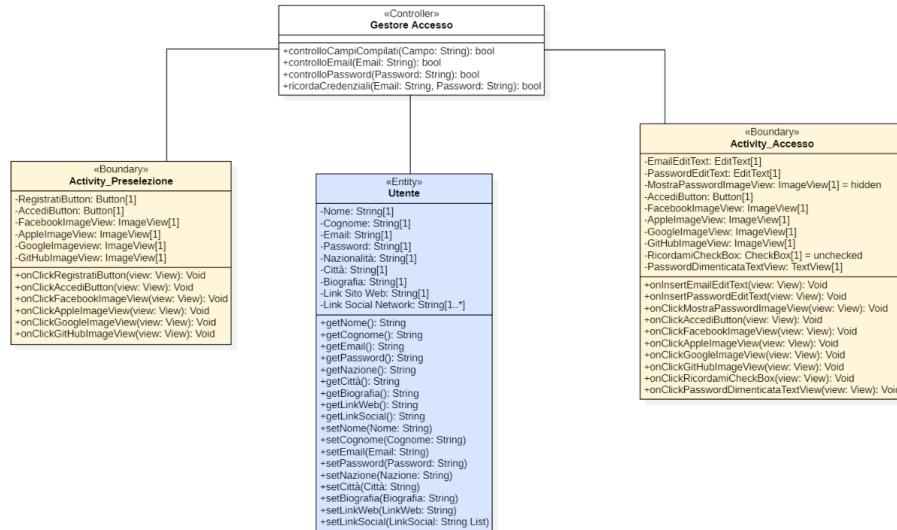


Figura 65: ClassDiagram - Accesso

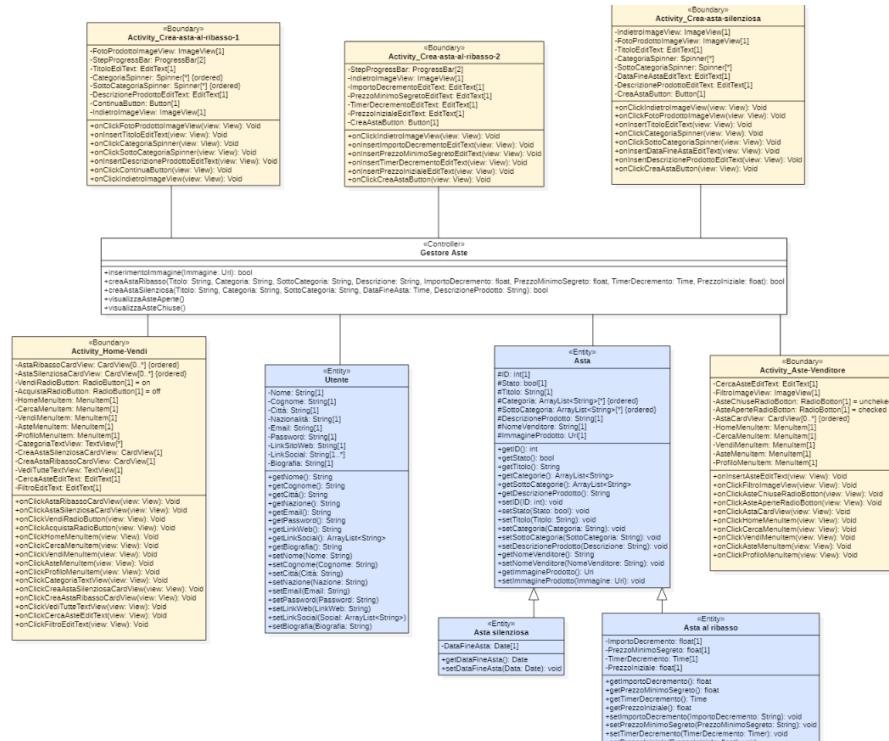


Figura 66: ClassDiagram - Gestore Aste

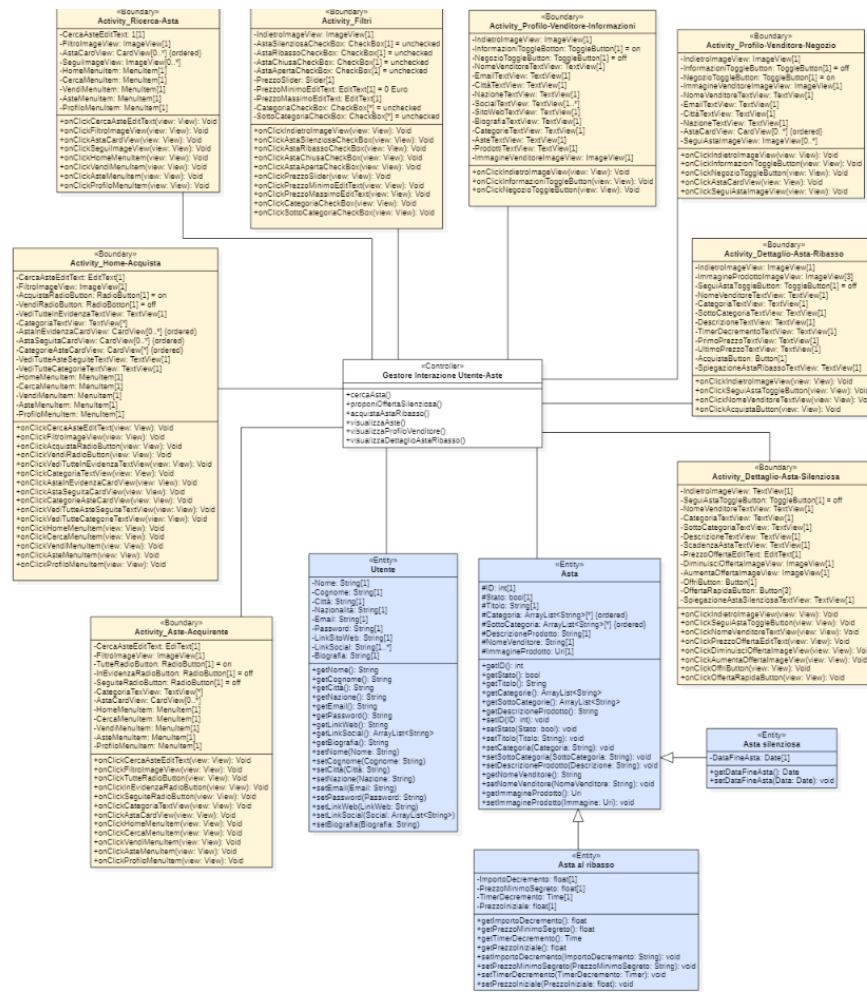


Figura 67: ClassDiagram - Gestore Utente-Aste

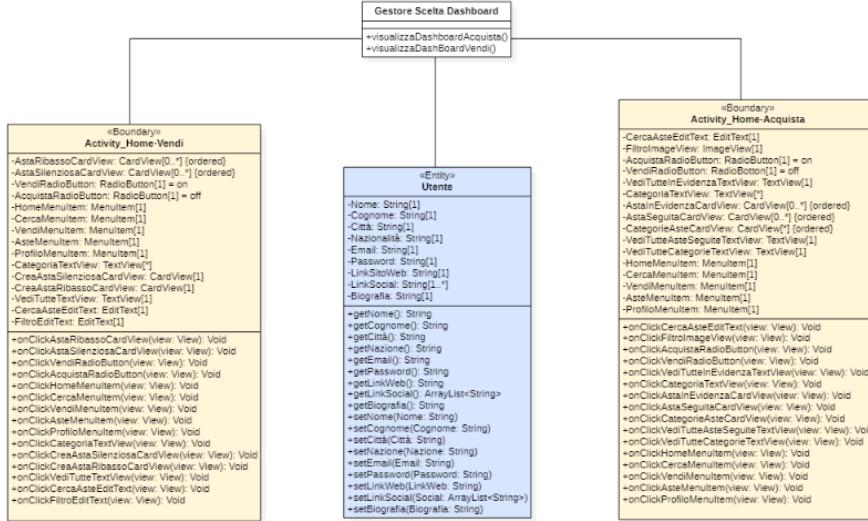


Figura 68: ClassDiagram - Scelta Dashboard

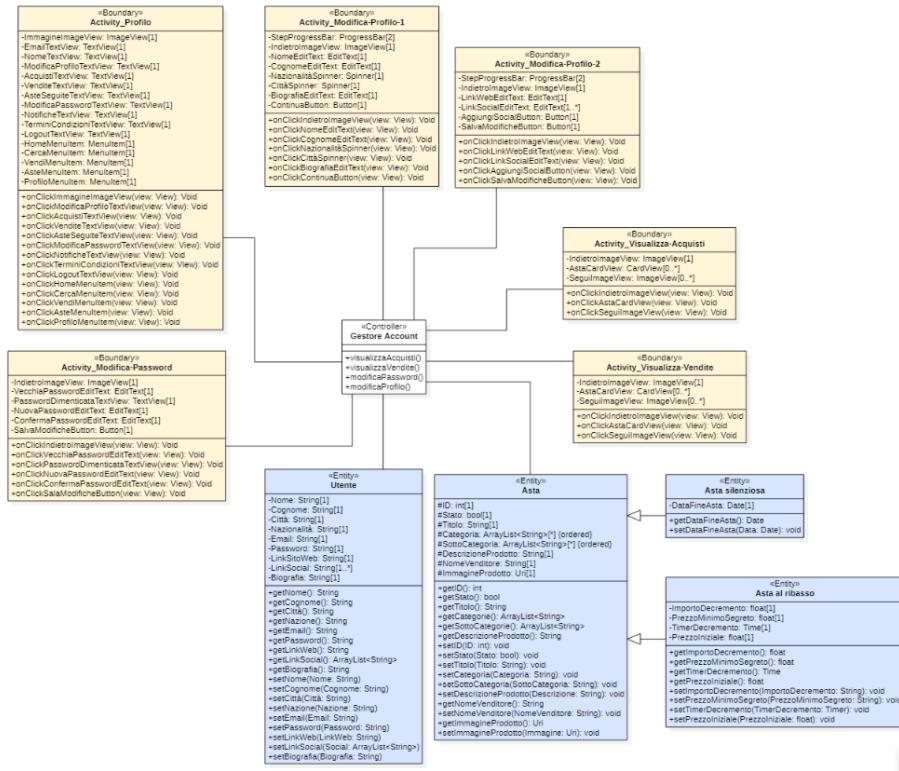


Figura 69: ClassDiagram - Gestore Account

2.2 Diagrammi di sequenza di analisi per casi d'uso significativi: Sequence Diagram

Nella seguente parte, esamineremo i diagrammi di sequenza di analisi per casi d'uso significativi. Questi diagrammi illustreranno in dettaglio come gli attori interagiscono con il sistema in scenari specifici. Ci concentreremo sull'ordine delle operazioni e sulle comunicazioni tra gli attori e il sistema.

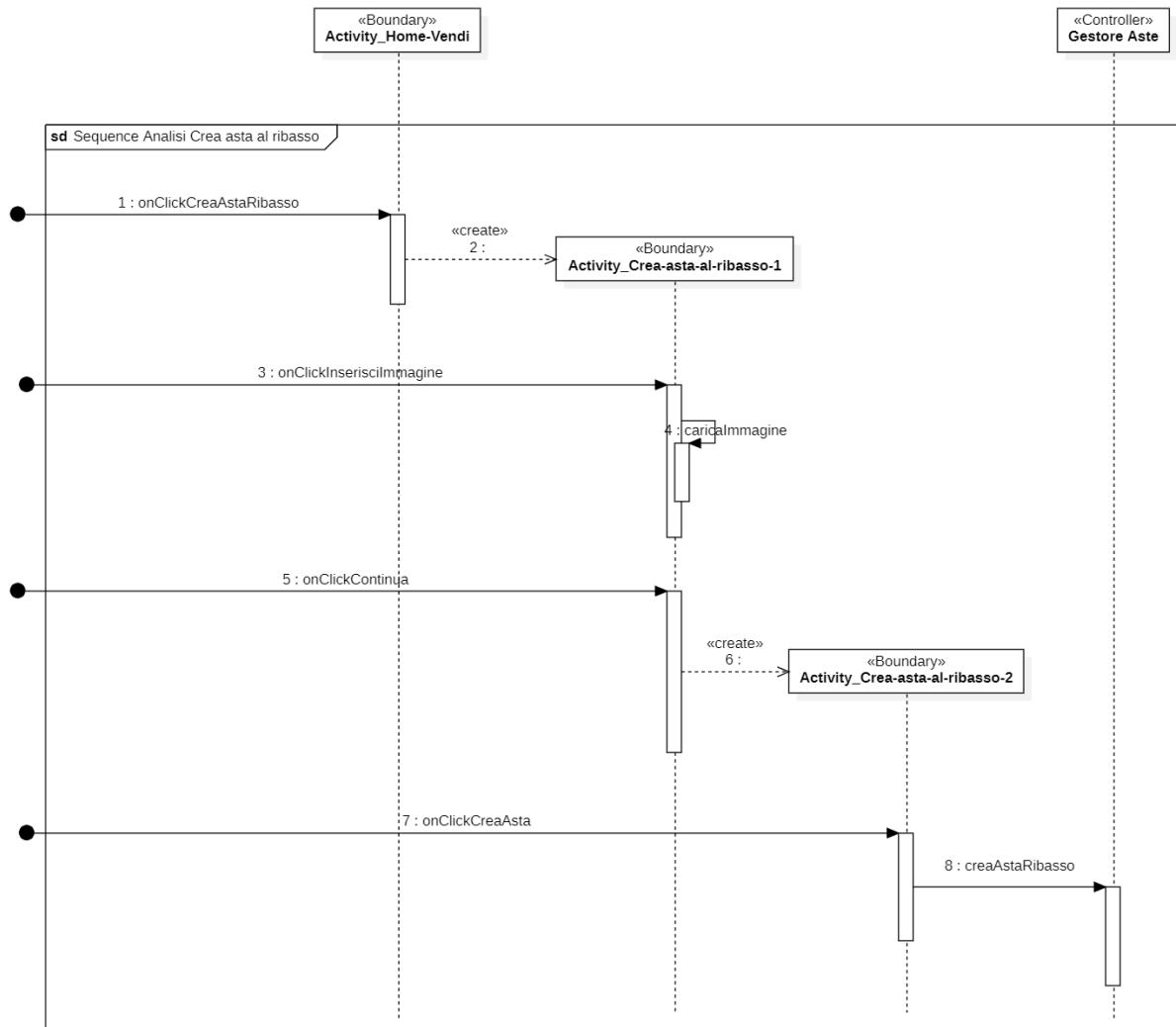


Figura 70: SequenceDiagram - Crea asta al ribasso

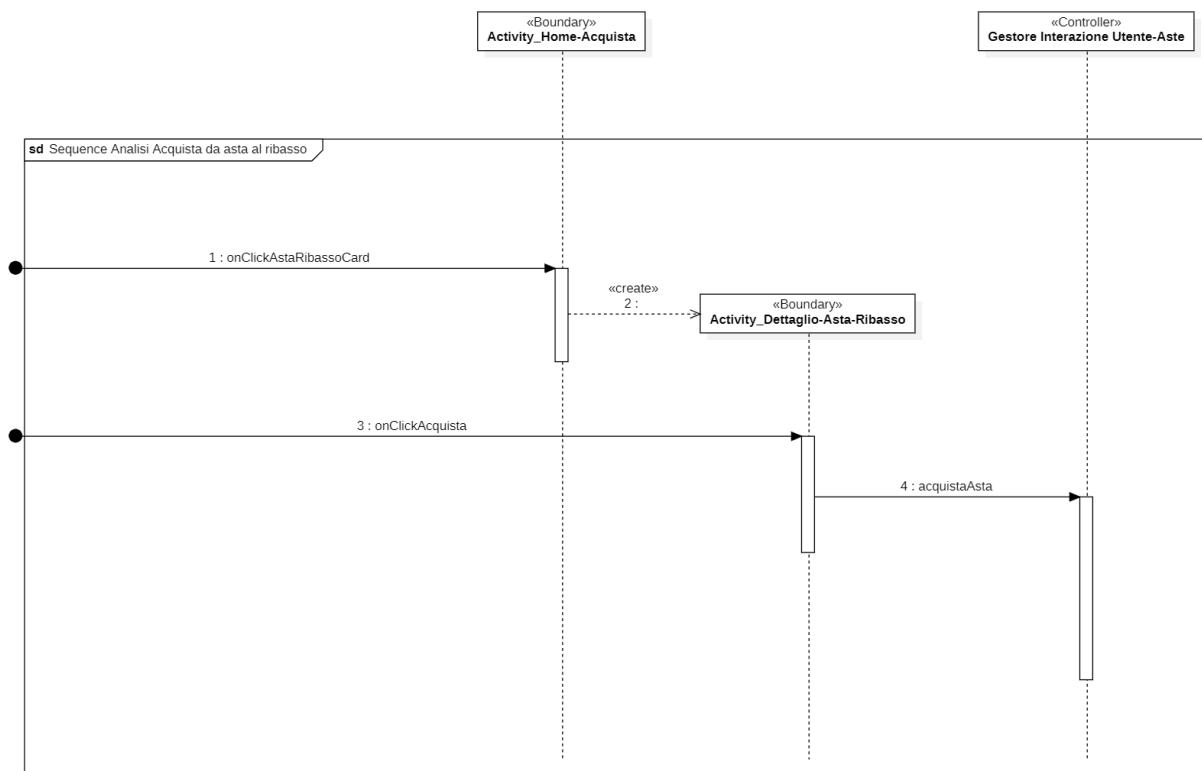


Figura 71: SequenceDiagram - Acquista da asta al ribasso

2.3 Prototipazione funzionale via statechart dell’interfaccia grafica: Statechart

Infine, presenteremo una prototipazione funzionale dell’interfaccia grafica utilizzando un diagramma di stato (statechart). Questo diagramma visualizzerà il comportamento dinamico del sistema, illustrando come reagisce alle interazioni degli utenti e agli eventi. Questa rappresentazione visuale ci aiuterà a comprendere meglio il flusso delle attività all’interno del sistema e a valutare l’usabilità dell’interfaccia utente.

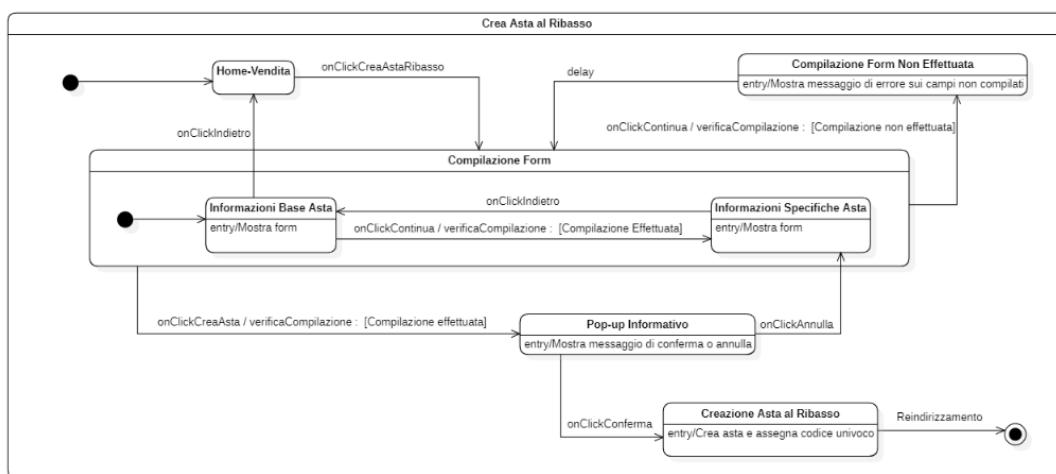


Figura 72: StatechartDiagram - Crea asta al ribasso

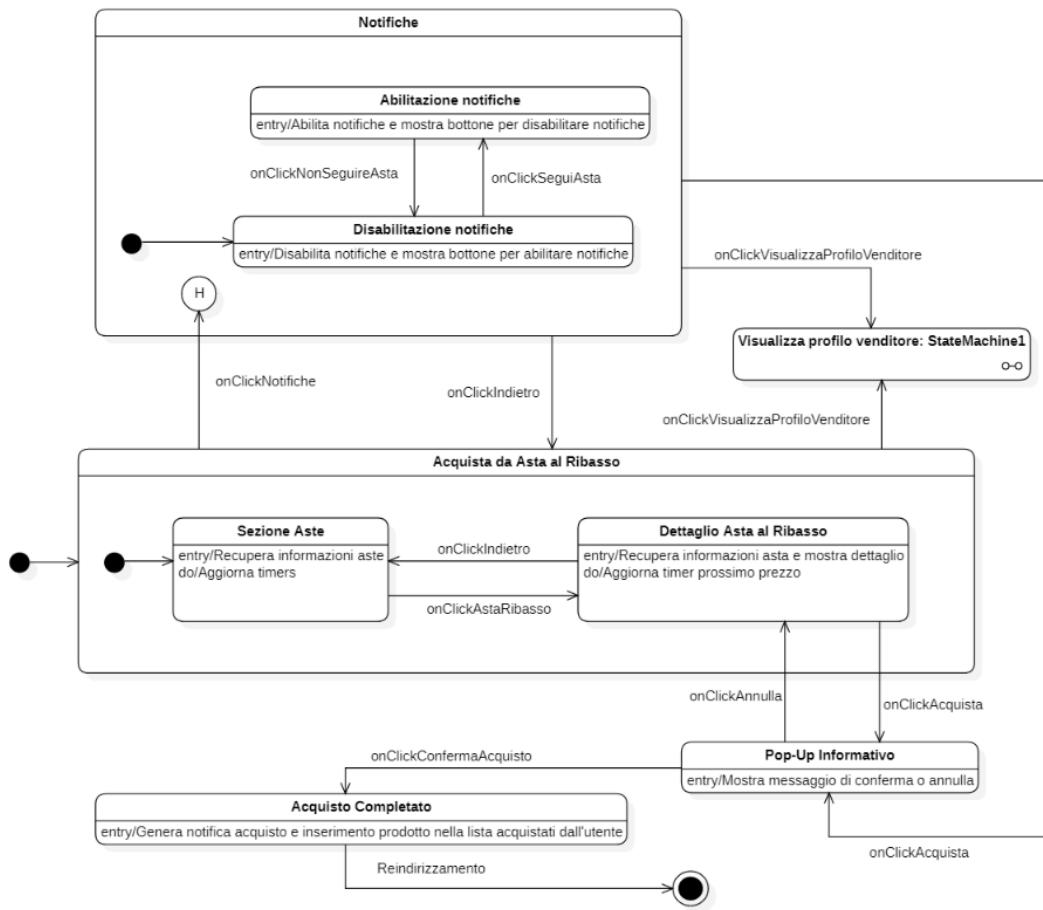


Figura 73: StatechartDiagram - Acquista da asta al ribasso



Parte II

Design del sistema



In questa sezione del documento, ci concentreremo sull'analisi dell'architettura del sistema, definendo esplicitamente i criteri di design che ne guidano lo sviluppo. Esamineremo le scelte tecnologiche adottate per implementare il sistema e discuteremo le motivazioni che hanno portato a tali decisioni. Inoltre, presenteremo il diagramma delle classi di design, che offre una rappresentazione visuale della struttura del sistema e delle relazioni tra le diverse componenti. Infine, esamineremo i diagrammi di sequenza di design, che mostreranno come il sistema interagisce con gli attori in scenari specifici.

3 Analisi dell'architettura con esplicita definizione dei criteri di design

In questa sezione, analizzeremo l'architettura del sistema e definiremo chiaramente i criteri di design che orientano lo sviluppo. Questi criteri costituiranno le linee guida fondamentali per garantire coerenza, scalabilità ed efficacia nel design complessivo del sistema.

3.1 Criteri di Design

I cinque aspetti menzionati - performance, affidabilità, costi di manutenzione, e usabilità - rappresentano elementi cruciali nel ciclo di vita dello sviluppo del software. Ognuno di essi svolge un ruolo fondamentale nel determinare il successo e la sostenibilità di un'applicazione o di un sistema software. Di seguito è fornita una panoramica dettagliata di ciascun aspetto:

1. **Performance:** Il software deve garantire agli utenti una corretta politica di concorrenza durante la fase di acquisto da un'asta al ribasso. Inoltre deve riuscire a gestire l'uso simultaneo dell'applicazione da parte di un considerevole numero di utenti.
2. **Affidabilità:** Il software deve risultare sicuro e deve riuscire a riprendersi da qualsiasi errore riportandomi sempre ad una situazione iniziale dove posso ripetere l'azione soggetta ad errore.
3. **Costi di Manutenzione:** Il codice deve risultare pulito e ben suddiviso nei vari file e cartelle, garantendo una corretta leggibilità e facilitazione per aggiornamenti e manutenzione ordinaria.
4. **Usabilità:** In base ai test di usabilità e allo studio fatto nelle sezioni precedenti, è garantita all'utente un'ottima usabilità e facilità d'uso dell'applicazione.
5. **Costi:** In base alle tecnologie utilizzate, ci aspettiamo costi di gestione relativamente bassi.

Ciascuno di questi aspetti è interconnesso. Ad esempio, una buona performance può ridurre i costi di manutenzione nel lungo termine, mentre un design intuitivo e un'alta usabilità possono migliorare l'affidabilità percepita riducendo gli errori da parte degli utenti. La gestione efficace di questi aspetti richiede un approccio equilibrato e una pianificazione attenta durante tutte le fasi dello sviluppo del software.

3.2 Architettura DietiDeals24

Il sistema DietiDeals24 è stato progettato seguendo un'architettura esterna a tre livelli, con chiara separazione tra i livelli di presentazione, applicazione e dati. Ognuno dei livelli può accedere esclusivamente a al layer di livello sottostante, rendendo l'architettura del sistema DietiDeals24 chiusa. Questa architettura offre una struttura solida per il sistema, garantendo una gestione efficiente dei dati e delle funzionalità, inoltre permette di garantire una adeguata sicurezza delle informazioni in quanto il client non può accedere direttamente al database.

Di seguito presentiamo una schematizzazione dell'architettura esterna del sistema DietiDeals24:



- **Client Tier**(Mobile Application Android): Il livello di presentazione è responsabile dell’interfaccia utente e comprende il frontend per Android. Questi componenti forniscono agli utenti un’interfaccia intuitiva e interattiva per interagire con il sistema DietiDeals24.
- **Business Tier**(Server Springboot e RestAPI): Il livello applicazione gestisce la logica di business del sistema. Include il backend e i controller associati ai frontend. Questo livello si occupa di elaborare le richieste degli utenti, gestire la logica delle operazioni, e garantire il corretto flusso di dati tra il frontend e il backend.
- **Database Tier**(Postgresql Database e Amazon S3): Il livello dati è responsabile della gestione dei dati del sistema. Comprende il database postgres in cui vengono archiviati i test, i dati degli utenti e tutte le informazioni pertinenti. Questo livello garantisce l’accesso sicuro e affidabile ai dati e ne gestisce la persistenza.

Architettura Server e Sicurezza dei Dati

Nello sviluppo del nostro software, abbiamo optato per un’architettura orientata al cloud, selezionando Amazon Web Services (AWS) per ospitare il server con le API REST e lo strumento di containerizzazione Docker. Il servizio di AWS scelto è Amazon Elastic Compute Cloud (EC2), un servizio IaaS che fornisce l’infrastruttura essenziale, lasciandoci la libertà di gestire gli altri aspetti del sistema.

Il back-end del nostro software gioca un ruolo fondamentale nella gestione e sicurezza dei dati. Ogni richiesta inviata da un client viene elaborata dal back-end prima di interagire con il database. Questa architettura non solo impedisce l’accesso diretto al database, garantendo la sicurezza dei dati, ma agisce anche come una barriera protettiva per l’integrità delle informazioni.

L’impiego di REST API migliora significativamente l’integrazione con altre applicazioni e servizi, ampliando la versatilità del sistema. Il server è supportato da SpringBoot, un framework efficiente per lo sviluppo di applicazioni web e servizi REST.

Per rafforzare ulteriormente la sicurezza, abbiamo integrato Java Persistence API (JPA). JPA non solo facilita l’interazione con il database attraverso un’astrazione ORM (Object-Relational Mapping), ma contribuisce anche alla sicurezza dei dati. Utilizzando JPA, si riduce il rischio di iniezioni SQL e si garantisce che le operazioni sul database siano eseguite in modo sicuro e conforme agli standard.

Di seguito è riportata una schematizzazione dell’architettura server:

- **Controller:** Classi Controller per gestire le richieste REST, richiedere i dati al database e inviare le risposte al client. Grazie alle JPA converte i Model in DTO e invia al client solo le informazioni pertinenti.
- **Service:** Classi Service per la logica di business e collegano i controller con le repository.
- **Repository:** Classi Repository per l’interazione con il database tramite JPA sfruttando le operazioni CRUD.
- **Model** - Classi Model che rappresentano le entità del dominio.
- **DTO** - Data Transfer Objects per il trasferimento di dati. Rappresentano le classi del model ma con solo gli attributi necessari con i dati fondamentali da mandare al client.

Endpoints e Operazioni CRUD

Abbiamo implementato le operazioni CRUD (Create, Read, Update, Delete) per gestire le richieste API. Queste operazioni offrono una maggiore flessibilità e funzionalità nel nostro sistema API, consentendo ai front-end di interagire con le nostre risorse in modi più efficaci e intuitivi. Nella sezione 13 sono dettagliate le chiamate Api Rest implementate.



Architettura Client

Nella progettazione dell'applicazione Android, abbiamo adottato una struttura parallela a quella del client desktop, implementando il modello MVP (Model-View-Presenter). Questo modello si adatta particolarmente bene all'ambiente Android, offrendo una distinzione chiara tra il Model, che gestisce i dati, la View, che si occupa della presentazione, e il Presenter, che funge da ponte tra i due. A differenza del tradizionale MVC (Model-View-Controller), nel MVP la View non interagisce direttamente con il Model, e il Presenter assume la responsabilità di collegare i dati del Model con la View. Questo permette una scrittura di test più agevole e migliora la manutenibilità e la leggibilità del codice.

Per la parte grafica, utilizzeremo XML, che consente una personalizzazione dettagliata dell'interfaccia utente. Ogni schermata dell'app sarà rappresentata da una "Activity", ad eccezione della schermata iniziale e delle sue funzioni collegate. Elementi come la barra di navigazione e la testata della pagina saranno costanti, incorporati in una schermata riutilizzabile attraverso l'uso di "Fragment" interscambiabili. Questo design favorisce una navigazione fluida e intuitiva, migliorando significativamente l'esperienza utente.

Di seguito è riportata una schematizzazione dell'architettura client:

- **Presenters**(Presenters per la gestione della logica di presentazione): facilita la comunicazione tra le diverse View e i Model. Inoltre, sono i Presenter che si occupano di invocare i metodi di Network per stabilire la comunicazione con il server.
- **Views**(Activity e Fragment per la visualizzazione delle interfacce utente): vengono definite le varie activity che mostrano i dati graficamente e gestiscono le interazioni dell'utente.
- **Model**(Classi per la gestione delle entità): implementazione delle entità
- **Network**: Contiene i metodi per la gestione delle chiamate di rete e l'elaborazione delle richieste HTTP.
- **Utils**: Classi di utilità, principalmente collegano il presenter con il network.

Architettura Database

Per l'archiviazione e la gestione dei dati, abbiamo scelto di utilizzare PostgreSQL, un sistema di gestione di basi di dati relazionali avanzato e open source, noto per la sua robustezza e affidabilità. PostgreSQL offre un'eccellente conformità agli standard SQL, supporto per query complesse e funzionalità di estensione avanzate, che sono ideali per gestire i dati strutturati in modo efficace e sicuro. In aggiunta, per la gestione e il trasferimento sicuro di file, abbiamo integrato Amazon S3. Questo server ci permette di gestire il trasferimento di file di grandi dimensioni, come documenti e immagini, in modo efficiente e sicuro. L'uso combinato di PostgreSQL per i dati strutturati e Amazon S3 per file immagini o di grandi dimensioni offre una soluzione di archiviazione dati completa, ottimizzando le prestazioni e garantendo la sicurezza dei dati.

Architettura Cloud

Per l'hosting del nostro server e servizi, abbiamo scelto Amazon Web Services (AWS) e, in particolare, il servizio Amazon Elastic Compute Cloud (EC2). AWS EC2 offre un'infrastruttura cloud scalabile e affidabile, che ci permette di distribuire e gestire facilmente le nostre applicazioni nel cloud. La flessibilità di EC2 ci consente di scegliere tra diversi tipi di istanze ottimizzate per vari carichi di lavoro, garantendo così che le risorse siano sempre allineate con le nostre esigenze di prestazioni e costi. La scelta di AWS EC2 come piattaforma cloud ci offre inoltre vantaggi come la resilienza, la disponibilità globale, la sicurezza avanzata e l'integrazione con altri servizi AWS, che sono fondamentali per garantire un servizio di hosting efficace e affidabile per le nostre applicazioni.



L'architettura cloud, quindi, Utilizza un server EC2 di AWS che ospita un container Docker, all'interno del quale sono eseguiti il server dell'applicazione e i database. In particolare il server EC2 AWS ospita il container Docker, il quale a sua volta ospita dockerfile per il server e il database.

4 Descrizione/motivazione delle scelte tecnologiche adottate e strumenti di implementazione

In questa sezione, esamineremo le scelte tecnologiche adottate per l'implementazione del sistema. Forneremo una descrizione dettagliata delle tecnologie, dei linguaggi di programmazione e dei framework selezionati, oltre a illustrare le motivazioni che hanno guidato tali scelte.

SpringBoot

SpringBoot è stato scelto come framework per lo sviluppo del back-end del sistema. Le motivazioni per questa scelta includono:

- **Facilità di sviluppo:** SpringBoot semplifica lo sviluppo di applicazioni Java basate su microservizi, consentendo una rapida implementazione.
- **Gestione delle dipendenze:** SpringBoot offre un sistema di gestione delle dipendenze integrato, semplificando la configurazione dei componenti.
- **Ampia comunità e supporto:** SpringBoot è supportato da una vasta comunità di sviluppatori, con una ricca documentazione e risorse online.
- **Integrazione con Docker:** SpringBoot è altamente compatibile con container Docker, consentendo una facile distribuzione e scalabilità.

Android

Sviluppare un'app su Android offre numerosi vantaggi. Primo tra tutti, Android domina il mercato globale degli smartphone, offrendo agli sviluppatori l'accesso a una vasta e variegata base di utenti. Questa piattaforma si distingue per la sua elevata personalizzazione, permettendo una grande libertà creativa nello sviluppo di funzionalità uniche e innovative. Inoltre, la stretta integrazione con l'ecosistema di servizi Google fornisce strumenti potenti e facilmente integrabili, come Google Maps e Google Drive.

Dal punto di vista della pubblicazione, il Google Play Store si caratterizza per un processo più veloce e meno restrittivo rispetto ad altre piattaforme, facilitando la pubblicazione e la distribuzione delle app. Infine, il carattere open source di Android e il suo ampio supporto comunitario forniscono risorse preziose e opportunità di apprendimento per gli sviluppatori, rendendo Android una scelta eccellente sia per progetti innovativi che per quelli tradizionali.

Android Studio è stato scelto come ambiente di sviluppo integrato (IDE) per la creazione dell'applicazione mobile. Le ragioni dietro questa scelta includono:

- **Compatibilità nativa con Android:** Android Studio offre un ambiente di sviluppo nativo per Android, garantendo una maggiore efficienza nella creazione di applicazioni per questa piattaforma.
- **Ampia comunità e supporto:** La comunità di sviluppatori Android è estesa, con una vasta gamma di risorse e supporto disponibili online.
- **Integrazione con emulatori Android:** Android Studio fornisce emulatori Android altamente configurabili, semplificando il processo di testing e debug dell'app su diverse versioni di Android e dispositivi.



- **Aggiornamenti frequenti:** Android Studio è costantemente aggiornato per adattarsi alle ultime tecnologie e best practice nello sviluppo per Android.

Docker

Docker è stato scelto come tecnologia per la distribuzione e la gestione dei componenti del sistema. Le motivazioni dietro questa scelta includono:

- **Containerizzazione:** Docker consente di creare container leggeri e isolati che includono tutte le dipendenze necessarie per l'esecuzione del software, garantendo la coerenza tra ambienti di sviluppo, test e produzione.
- **Scalabilità:** Docker semplifica la scalabilità orizzontale dei componenti del sistema, consentendo di gestire facilmente carichi di lavoro crescenti.
- **Portabilità:** I container Docker possono essere eseguiti su diverse piattaforme senza problemi di compatibilità, garantendo una maggiore portabilità del software.

PostgreSQL

L'adozione di un database relazionale è ideale per applicazioni che necessitano di una gestione dei dati organizzata e consistente. Questi database offrono una struttura rigida, che è perfetta per garantire l'integrità e l'affidabilità dei dati. Inoltre, il loro modello basato su tabelle, righe e colonne facilita le interrogazioni complesse e relazioni tra dati, rendendoli ideali per applicazioni aziendali e sistemi che richiedono report dettagliati e analisi.

PostgreSQL è stato scelto come sistema di gestione di basi di dati relazionali per il nostro progetto. Le motivazioni di questa scelta sono basate sui seguenti punti chiave:

- **Affidabilità e Integrità dei Dati:** PostgreSQL è noto per la sua forte enfasi sull'affidabilità, la coerenza dei dati e il rispetto degli standard SQL, che sono essenziali per qualsiasi applicazione critica.
- **Supporto per Funzionalità Avanzate:** Offre un supporto avanzato per funzionalità complesse come join complessi, sottoquery, transazioni, viste materializzate e una varietà di tipi di dati, che sono indispensabili per la gestione efficiente di grandi set di dati.
- **Estensibilità e Supporto per Procedure Memorizzate:** PostgreSQL permette una notevole estensibilità, incluso il supporto per procedure memorizzate, trigger e funzioni definite dall'utente, che consente una maggiore flessibilità nella gestione della logica del database.
- **Sicurezza Robusta:** Fornisce robuste funzionalità di sicurezza, comprese l'autenticazione basata sui ruoli, la crittografia SSL e il controllo degli accessi granulare, garantendo la sicurezza dei dati sensibili.
- **Compatibilità:** PostgreSQL è compatibile con una vasta gamma di linguaggi di programmazione e piattaforme, rendendolo una scelta flessibile per diverse architetture di software.

Server FTP e Amazon S3

Nel nostro progetto, per la gestione delle immagini caricate tramite l'applicazione, abbiamo inizialmente considerato l'uso di un server FTP. Tuttavia, dopo un'attenta valutazione, abbiamo optato per l'utilizzo di Amazon S3 (Simple Storage Service), un servizio di storage di oggetti fornito da AWS. Questa scelta è stata guidata dalla necessità di garantire maggiore scalabilità, sicurezza e affidabilità nel salvataggio dei file.

Amazon S3 offre numerosi vantaggi rispetto a un tradizionale server FTP, tra cui:



- **Scalabilità:** Amazon S3 gestisce automaticamente l'aumento del carico di lavoro e della quantità di storage necessaria, adattandosi dinamicamente alle esigenze dell'applicazione.
- **Sicurezza dei Dati:** Fornisce robuste opzioni di sicurezza per la protezione dei dati, comprese la crittografia dei dati in transito e a riposo.
- **Protezione da Perdite di Dati:** Garantisce una durabilità elevata, riducendo significativamente il rischio di perdite di dati.
- **Facilità di Accesso:** Permette un accesso semplice e rapido ai file, con diverse opzioni di configurazione per la gestione dell'accesso pubblico o privato.

Invece di salvare fisicamente le immagini sul server, memorizziamo le immagini caricate nel bucket S3 e salviamo l'URL corrispondente nel nostro database PostgreSQL. Questo approccio non solo migliora la gestione e l'accessibilità dei file ma contribuisce anche a semplificare la struttura del nostro sistema, rendendolo più efficiente e facilmente scalabile.

Java Persistence API (JPA)

Java Persistence API (JPA) è uno standard dell'industria per la mappatura oggetto-relazionale (ORM) per la piattaforma Java. È una parte della specifica Java EE e Java SE ed è usata per semplificare lo sviluppo di applicazioni che interagiscono con i database. Di seguito sono elencate alcune delle caratteristiche principali e i vantaggi dell'uso di JPA:

- **Astrazione del Livello di Persistenza:** JPA fornisce un'interfaccia ORM che astrae la logica di persistenza dallo strato del database, permettendo agli sviluppatori di interagire con gli oggetti anziché con le query SQL dirette.
- **Indipendenza dal Database:** Grazie alla sua natura ORM, JPA consente di scrivere codice indipendente dal database. Ciò significa che lo stesso codice può funzionare con diversi database, facilitando la portabilità e la scalabilità delle applicazioni.
- **Mappatura Oggetto-Relazionale:** Permette di mappare facilmente le tabelle dei database in classi Java, rendendo più intuitiva la gestione dei dati e riducendo l'errore umano nella traduzione tra il modello relazionale e quello oggetto.
- **Gestione delle Entità:** JPA gestisce il ciclo di vita delle entità Java, inclusa la persistenza, l'aggiornamento, la cancellazione e la ricerca, attraverso Entity Manager.
- **Query Language:** JPA introduce JPQL (Java Persistence Query Language), che è simile a SQL ma opera sul modello oggetto piuttosto che direttamente sulle tabelle, offrendo una maggiore flessibilità e potenza nelle query.
- **Caching:** Migliora le prestazioni dell'applicazione attraverso meccanismi di caching, riducendo il carico sul database e migliorando i tempi di risposta delle query.

AWS (Amazon Web Services)

Amazon Web Services è stata scelta come piattaforma cloud per l'hosting del sistema. Le ragioni dietro questa scelta includono:

- **Affidabilità e scalabilità:** AWS offre una vasta gamma di servizi altamente affidabili e scalabili che consentono di gestire con efficacia il traffico e le risorse del sistema.
- **Ampia disponibilità regionale:** AWS ha una presenza globale con numerosi data center in diverse regioni, garantendo alta disponibilità e ridondanza.



- **Servizi di sicurezza avanzati:** AWS offre un ampio set di strumenti e servizi per garantire la sicurezza delle applicazioni e dei dati ospitati sulla piattaforma.

Servizi di AWS utilizzati:

- EC2
- Amazon S3
- Elastic IP (51.21.58.218)

5 Esempio di funzionamento del software

L'applicazione comprende i seguenti componenti:

- **Frontend Android:** Interfaccia utente dell'applicazione per gli utenti.
- **Server Spring Boot con API REST e JPA:** Gestisce la logica e l'accesso ai dati.
- **Database PostgreSQL:** Memorizza i dati strutturati dell'applicazione.
- **Amazon S3:** Utilizzato per lo storage di file e immagini.
- **EC2 e Docker:** Fornisce capacità di calcolo scalabile e containerizzazione dell'applicazione.

Ecco il flusso di utilizzo dell'applicazione, partendo dall'interazione dell'utente:

1. **Interazione Iniziale dell'Utente:** L'utente avvia l'applicazione sul proprio dispositivo Android e interagisce con l'interfaccia utente. Questo può includere azioni come accedere, visualizzare o modificare dati, caricare immagini, ecc.
2. **Richieste al Server:** Quando l'utente compie un'azione che richiede dati dal server o deve salvare dei dati, l'app Android invia una richiesta al server Spring Boot tramite API REST. Questa richiesta viaggia attraverso Internet verso il server hostato su EC2.
3. **Elaborazione del Server:** Il server Spring Boot riceve la richiesta e la elabora. Se la richiesta richiede un'interazione con il database (come recuperare o salvare dati), il server utilizza JPA per comunicare con il database PostgreSQL. Se la richiesta implica il caricamento o il recupero di immagini, il server interagisce con Amazon S3.
4. **Database PostgreSQL e Amazon S3:** PostgreSQL gestisce tutte le richieste relative ai dati strutturati, mentre Amazon S3 gestisce le richieste relative ai file e alle immagini. Questi servizi possono essere ospitati separatamente e comunicano con il server Spring Boot secondo necessità.
5. **Risposta al Frontend:** Una volta che il server ha elaborato la richiesta (che potrebbe includere salvare o recuperare dati da PostgreSQL o S3), invia una risposta indietro al frontend Android.
6. **Visualizzazione dei Dati all'Utente:** Il frontend Android riceve la risposta dal server e aggiorna l'interfaccia utente di conseguenza. Questo potrebbe includere visualizzare nuovi dati, confermare il successo di un'azione o mostrare errori.

6 Diagramma delle classi di design

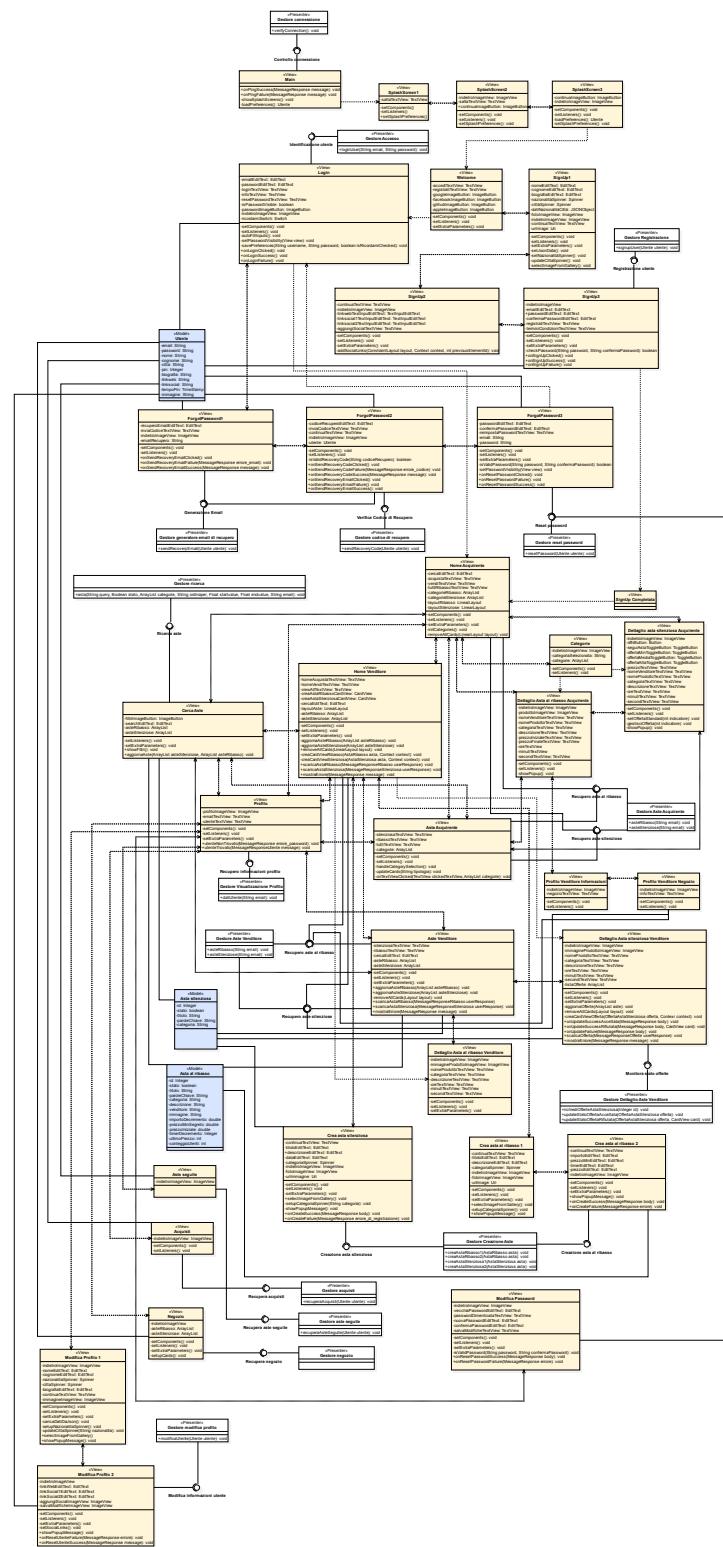


Figura 74: Diagramma delle classi di design Front-end

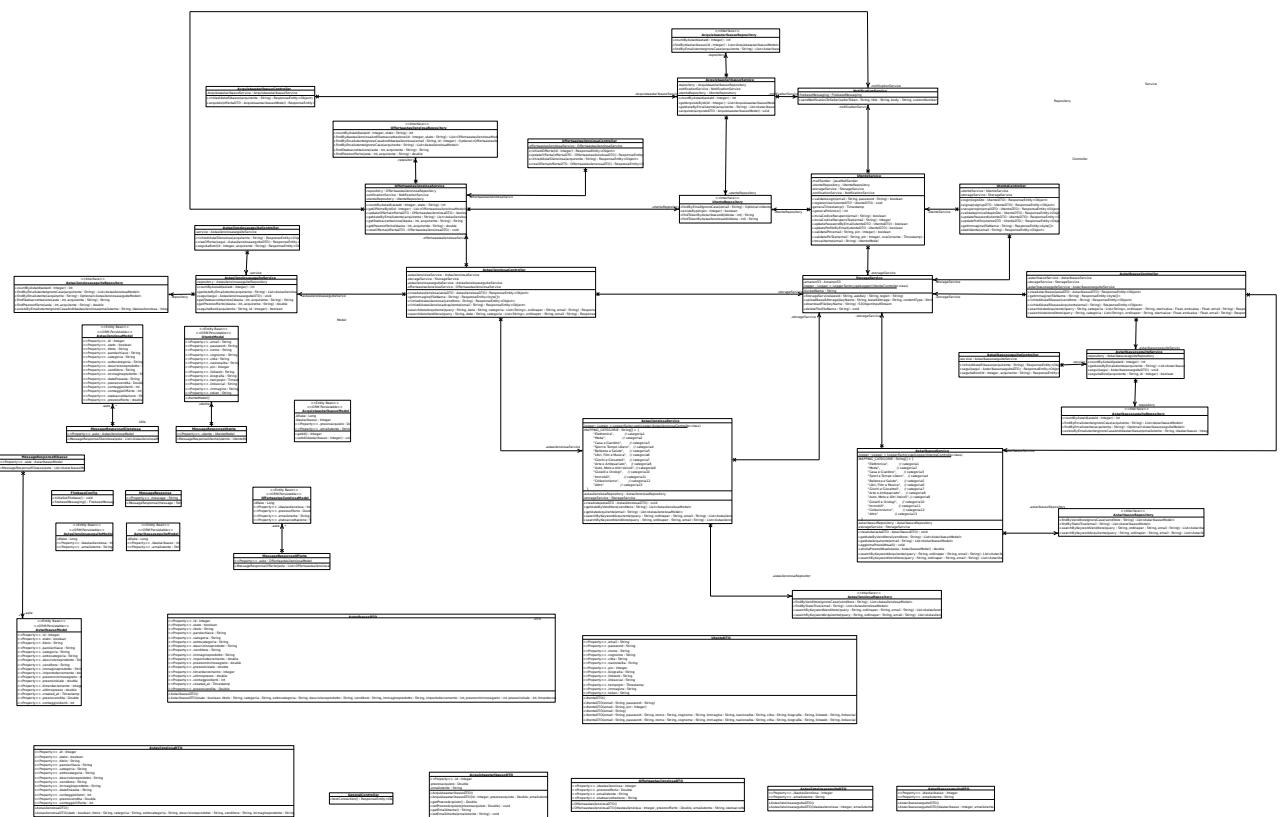


Figura 75: Diagramma delle classi di design Back-end

7 Diagrammi di sequenza di design per i casi d'uso

Esamineremo i diagrammi di sequenza di design, che illustrano in dettaglio come il sistema interagisce con gli attori in scenari specifici. Questi diagrammi mostreranno il flusso delle attività all'interno del sistema e come le varie componenti comunicano tra loro per raggiungere gli obiettivi dei casi d'uso.

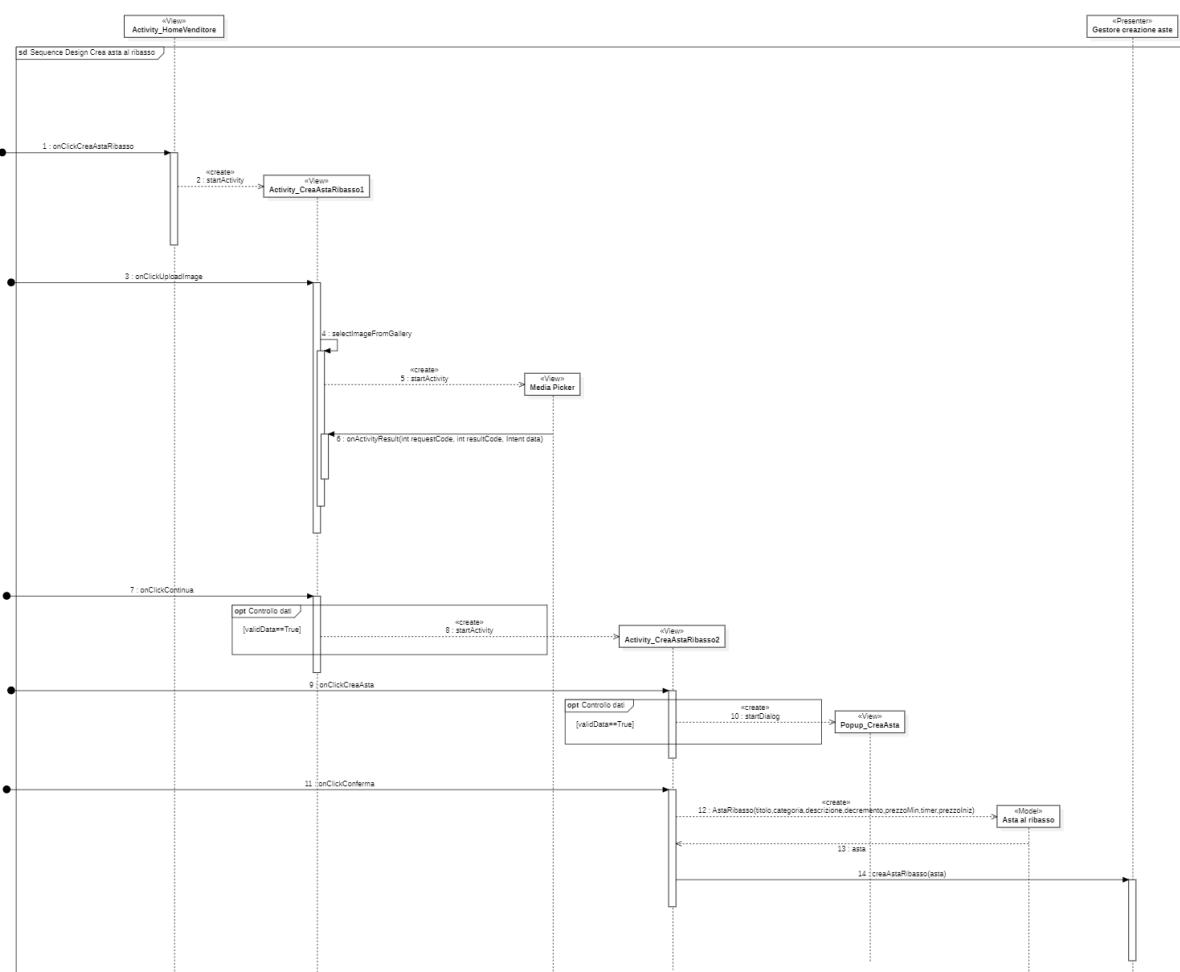


Figura 76: Diagramma di sequenza di design per il caso d'uso Crea asta al ribasso

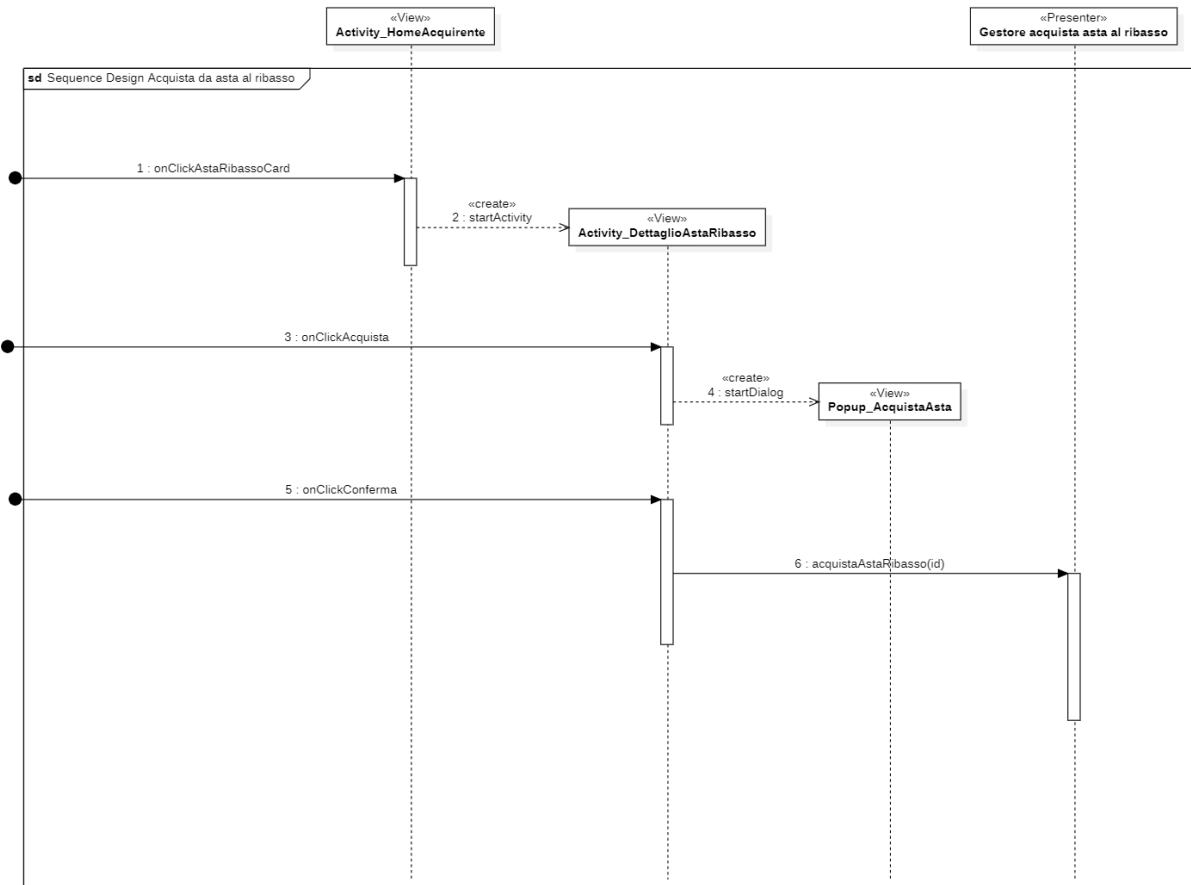


Figura 77: Diagramma di sequenza di design per il caso d'uso Acquista asta al ribasso

8 Database schema

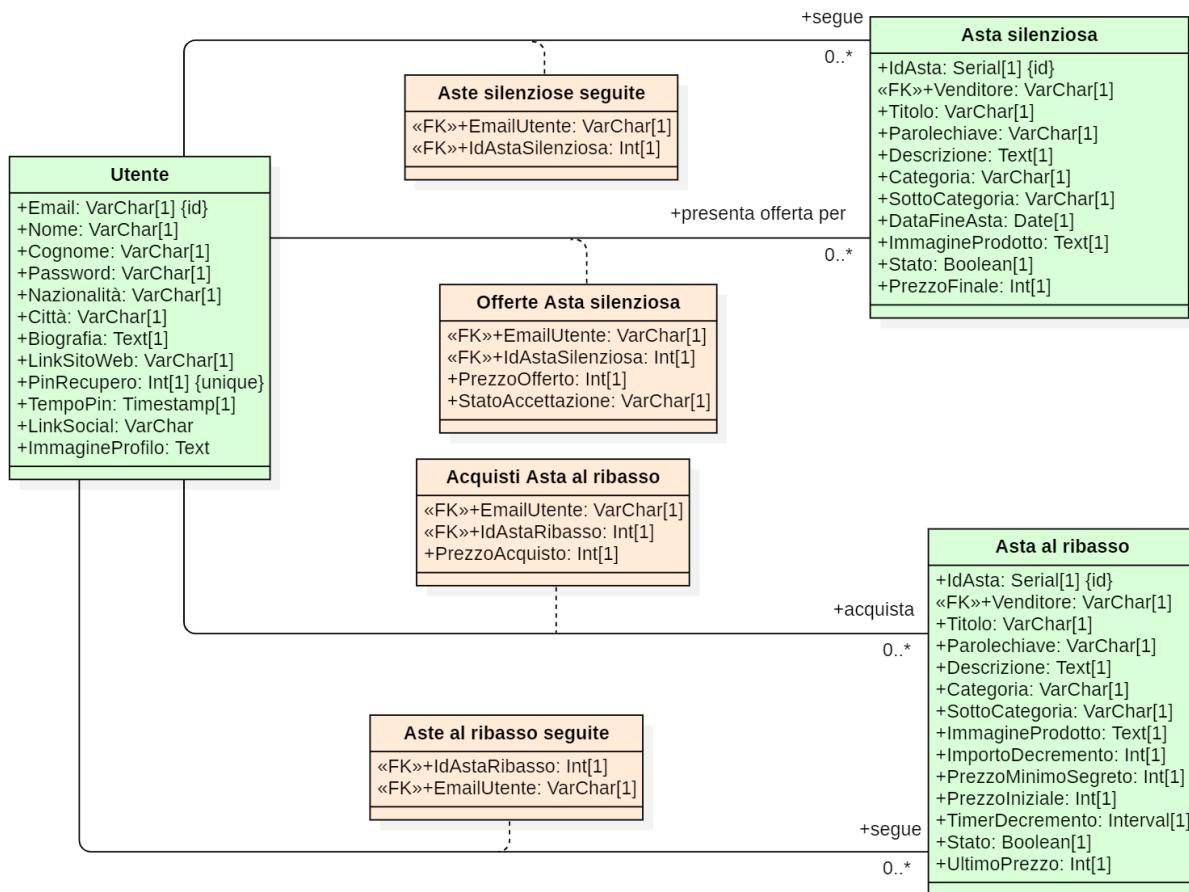


Figura 78: Database schema



Parte III

Codice Sorgente



9 Codice Sorgente sviluppato, comprensivo di eventuale Dockerfile

Il codice sorgente costituisce il nucleo del progetto e comprende tutte le componenti necessarie per l'esecuzione dell'applicazione o del servizio. Il file Dockerfile fornisce istruzioni per la creazione di un'immagine Docker che può essere utilizzata per distribuire e eseguire l'applicazione in un ambiente containerizzato. Nello specifico, in questa sezione andremo a presentare le soluzioni implemetative adottate per sviluppare alcune funzionalità principali dell'applicativo come:

- Registrazione utente
- Password dimenticata
- Creazione aste
- Visualizzazione delle aste
- Finalizzazione acquisto o presentazione offerta per le aste
- Gestione offerte aste da parte del venditore
- Notifiche di conclusione asta
- Ricerca aste nel sistema con filtri attivi

La registrazione degli utenti costituisce il primo passo essenziale per accedere all'applicazione. Durante questo processo, gli utenti possono creare un account fornendo dettagli come nome, email, password, immagine del profilo e una breve biografia. Al termine della registrazione, vengono automaticamente inviate notifiche push e email di benvenuto per garantire una migliore esperienza agli utenti appena registrati.

Nel caso in cui gli utenti dimentichino la password, è disponibile la funzione di recupero password. Tramite un pin di recupero inviato alla loro email, gli utenti possono reimpostare la password in modo sicuro e veloce.

Una volta registrati, gli utenti hanno accesso alla creazione e alla visualizzazione delle aste. Possono creare nuove aste inserendo dettagli come nome, descrizione, immagini e prezzo di partenza. Le aste sono presentate tramite cardview, che forniscono una panoramica chiara e immediata delle informazioni principali.

Per partecipare alle aste, gli utenti possono acquistare prodotti o presentare offerte. Possono anche gestire le offerte ricevute sulle aste che hanno creato, valutandole e decidendo se accettarle o rifiutarle.

Al termine delle aste create dagli utenti, vengono inviate notifiche push e email per informarli sui risultati dell'asta, garantendo una comunicazione tempestiva e trasparente.

Infine, per facilitare la ricerca delle aste desiderate, è stata implementata una funzione di ricerca avanzata. Utilizzando parole chiave e filtri, gli utenti possono trovare rapidamente le aste di loro interesse, migliorando l'efficienza e la soddisfazione complessiva dell'esperienza utente.

In sintesi, queste funzionalità sono progettate per offrire agli utenti un'esperienza completa, intuitiva e sicura all'interno dell'applicazione, migliorando la fruibilità e la facilità d'uso.

10 File di build automatica

File build Gradle modulo:



```
1 plugins {
2     id 'com.android.application'
3     id 'com.google.gms.google-services'
4 }
5
6 android {
7     namespace 'com.example.dd24client'
8     compileSdk 34
9
10    defaultConfig {
11        applicationId "com.example.dd24client"
12        minSdk 26
13        targetSdk 34
14        versionCode 1
15        versionName "1.0"
16
17        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
18
19
20        manifestPlaceholders = ['appAuthRedirectScheme': 'fb123456789']
21    }
22
23    buildTypes {
24        release {
25            minifyEnabled false
26            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
27        }
28    }
29    compileOptions {
30        sourceCompatibility JavaVersion.VERSION_1_8
31        targetCompatibility JavaVersion.VERSION_1_8
32    }
33 }
34
35 dependencies {
36     implementation 'androidx.activity:activity-ktx:1.8.2'
37     implementation 'androidx.fragment:fragment-ktx:1.6.2'
38     implementation 'androidx.appcompat:appcompat:1.6.1'
39     implementation 'com.google.android.material:material:1.11.0'
40     implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
41     implementation 'commons-io:commons-io:2.8.0'
42
43     testImplementation 'junit:junit:4.13.2'
44     testImplementation 'org.junit.jupiter:junit-jupiter:5.8.1'
45     testImplementation 'org.mockito:mockito-core:3.12.4'
46
47     androidTestImplementation 'androidx.test.ext:junit:1.1.5'
48     androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
49
50     implementation 'com.squareup.retrofit2:retrofit:2.9.0'
51     implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
52     coreLibraryDesugaring 'com.android.tools:desugar_jdk_libs:2.0.4' // Aggiunta per il supporto di LocalDate
53     implementation 'com.google.android.gms:play-services-auth:21.0.0' //accesso tramite google
54     implementation 'com.facebook.android:facebook-android-sdk:16.3.0' //accesso tramite facebook
55     implementation 'net.openid:appauth:0.11.1' //accesso tramite apple
56     implementation 'com.github.bumptech.glide:glide:4.11.0'
57     annotationProcessor 'com.github.bumptech.glide:compiler:4.11.0'
58 }
```



```
59     implementation 'com.google.firebaseio:firebase-auth:22.3.1'
60     implementation platform('com.google.firebaseio:firebase-bom:32.7.2')
61     implementation 'com.google.firebaseio:firebase-analytics:20.0.0'
62     implementation 'de.hdodenhof:circleimageview:3.1.0'
63     implementation 'com.google.firebaseio:firebase-messaging:23.0.0'
64     implementation 'com.google.code.gson:gson:2.8.8'
65
66 }
```

File build Gradle project:

```
1 buildscript {
2     ext.kotlin_version = '1.8.0'
3     repositories {
4         google()
5         mavenCentral()
6     }
7     dependencies {
8         classpath 'com.android.tools.build:gradle:8.3.1'
9
10        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
11        classpath 'com.google.gms:google-services:4.4.1'
12    }
13 }
14
15 plugins {
16     id 'com.android.application' version '8.3.1' apply false
17     id 'com.android.library' version '8.3.1' apply false
18     id 'com.google.gms.google-services' version '4.4.1' apply false
19     id "org.sonarqube" version "4.4.1.3373"
20 }
21
22
23 sonarqube {
24     properties {
25         property "sonar.projectKey", "INGSW2324"
26         property "sonar.projectName", "INGSW2324"
27         property "sonar.host.url", "http://localhost:9000"
28         property "sonar.login", "sqp_fa6d7076eb71d33f3207d72dadbb515fcccfaeba"
29     }
30 }
```

Dockerfile:

```
1 # Utilizza un'immagine base di Java.
2 FROM openjdk:17
3
4 # Imposta la directory di lavoro all'interno del container
5 WORKDIR /app
6
7 # Copia il file JAR dall'host locale al filesystem del container
8 COPY build/libs/ServerDD24-0.0.1-SNAPSHOT.jar app.jar
9 COPY src/main/java/com/example/serverdd24/serviceAccountKey.json src/main/java/com/
10 example/serverdd24/serviceAccountKey.json
11
12 # Espone la porta su cui l'applicazione sarà accessibile
13 EXPOSE 8080
14
15 # Comando per eseguire l'applicazione
CMD ["java", "-jar", "app.jar"]
```

Oltre al dockerfile presentiamo anche una lista di comandi che abbiamo utilizzato per caricare il server springboot sul container docker presente su ec2:



```

1 COMANDO PER INVIARE CARTELLA AL SERVER EC2: pscp -r -i "C:\Users\biagi\Downloads
2 \WindowsKey.ppk" "C:\Users\biagi\Desktop\INGSW\ServerDD24" ubuntu@51.21.10.214:/home/ubuntu/ingswServer
3
4 COMANDO PER CREARE IMMAGINE SERVER SPRINGBOOT: docker build -t springboot-
5 container .
6
7 COMANDO PER AVVIARE CONTAINER SERVER SPRINGBOOT: docker run -d --name springboot
8 -app --network db-springboot-network -e DATABASE_URL=jdbc:postgresql://
9 dietideals-db:5432/postgres -e DATABASE_USERNAME=postgres -e DATABASE_PASSWORD=
BiagioErasmo2024 -p 8080:8080 springboot-container:latest

```

11 Evidenza dell'uso di strumenti di versioning

Nel corso dello sviluppo del nostro software, GitHub e Overleaf hanno svolto un ruolo cruciale, fungendo da piattaforme centrali per la collaborazione e la gestione del codice sorgente e della documentazione del progetto. In particolar modo, l'utilizzo di GitHub ha apportato numerosi benefici al nostro processo di sviluppo:

- Collaborazione Efficace:** Abbiamo utilizzato i repository GitHub per condividere e sincronizzare il lavoro di tutti gli sviluppatori coinvolti nel progetto. Questo ha facilitato un ambiente di lavoro collaborativo, dove ogni membro del team poteva contribuire in modo efficiente.
- Gestione del Codice:** Il sistema di branching e merging offerto da GitHub ci ha permesso di lavorare su diverse funzionalità in parallelo, riducendo i conflitti di codice e migliorando la qualità complessiva del software.

The screenshot shows a GitHub repository page for 'INGSW'. The repository has 2 branches and 0 tags. The 'About' section describes DietiDeals24 as a platform for managing online auctions. The 'Contributors' section lists 'biagioSc' and 'CS-Era'.

Figura 79: Github



12 Report di qualità del codice, generati da SonarQube o simili (nel caso solo per il back-end)

Durante il processo di sviluppo del software, è fondamentale garantire la qualità del codice attraverso l'analisi approfondita e sistematica. In questo contesto, l'utilizzo di strumenti di analisi statica come SonarQube riveste un ruolo cruciale nell'identificare potenziali problemi e migliorare la robustezza, la sicurezza e la manutenibilità del software.

L'analisi del codice svolta utilizzando SonarQube ha restituito risultati generalmente positivi, evidenziando una buona qualità del codice nelle aree principali del progetto. Tuttavia, durante l'esame dettagliato, sono emersi alcuni problemi minori.

In questa sezione, esamineremo in dettaglio i risultati dell'analisi del codice, concentrandoci sugli aspetti positivi e sui piccoli problemi identificati. Attraverso l'analisi approfondita di tali problemi e l'implementazione delle relative soluzioni, miriamo a garantire un codice di alta qualità e una migliore esperienza complessiva degli utenti.

Problemi relativi all'Utilizzo di HTTP

Uno dei problemi rilevati durante l'analisi del codice riguarda l'utilizzo diretto del protocollo HTTP anziché di HTTPS per le comunicazioni sicure. L'uso di HTTP espone l'applicazione a rischi di sicurezza, come l'intercettazione dei dati sensibili durante la trasmissione. Nello sviluppo di questa applicazione si è comunque deciso di servirsi del protocollo HTTP in quanto per le specifiche della nostra app al momento l'uso del protocollo HTTPS può essere rimandato.

Problemi di Manutenibilità dovuti all'Utilizzo di Stringhe Dirette

Un altro problema rilevato riguarda l'utilizzo diretto di stringhe nei passaggi dei parametri invece di utilizzare costanti o variabili stringhe. Questa pratica ha reso il codice meno manutenibile a livello di analisi, in quanto le stringhe dirette possono essere soggette a problemi. Per questioni di tempistiche si è deciso di sostituire solo nella maggior parte del codice di sostituire le stringhe con variabili string.

Spiegazione della copertura dei test a 0.0% su SonarQube

La copertura dei test segnalata come 0.0% su SonarQube può essere attribuita al fatto che i test sono stati aggiunti al progetto in un momento successivo rispetto all'ultima analisi eseguita.

In conclusione siamo comunque soddisfatti del report ricevuto, grazie al quale abbiamo apportato notevoli modifiche e miglioramenti generali al codice sviluppato.

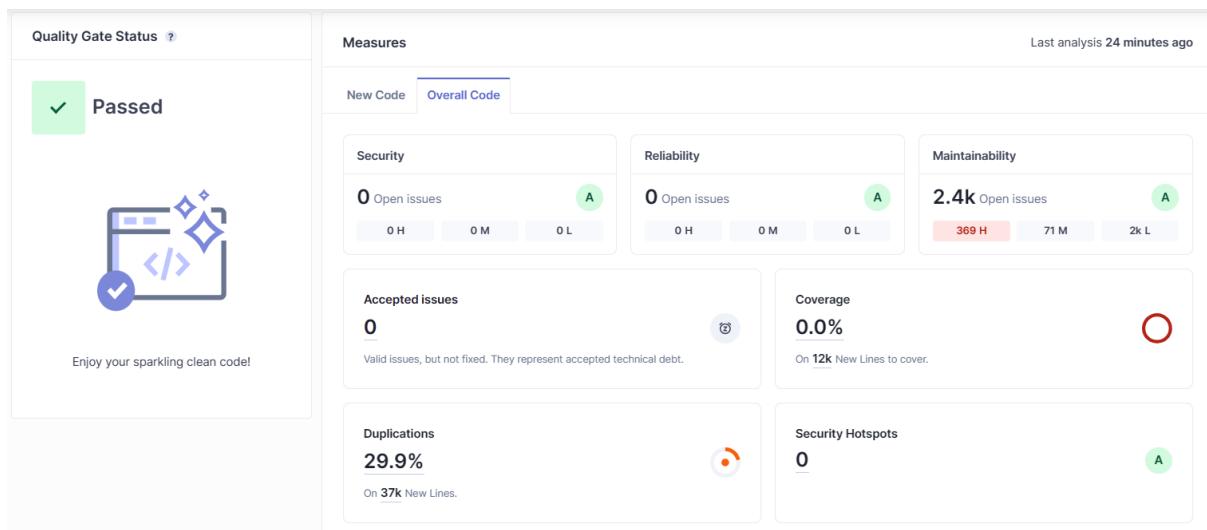


Figura 80: Sonarquebe - Analisi qualità codice

13 Chiamate API

L'analisi delle chiamate API è fondamentale per comprendere il funzionamento e le interazioni di un sistema software. In questa sezione, forniremo una panoramica delle principali chiamate API utilizzate nell'applicazione, insieme a una breve descrizione di ciascuna.

Chiamate API

/testConnection	Effettua un ping per testare la connessione.
/api/utente/login	Effettua il login dell'utente.
/api/utente/signup	Registra un nuovo utente.
/api/utente/recuperopin	Gestisce il recupero del PIN per l'utente.
/api/utente/validatepin	Valida il PIN inserito dall'utente per il recupero.
/api/utente/updatePassword	Aggiorna la password dell'utente.
/api/astaribasso/richiedi	Richiede informazioni su un'asta al ribasso.
/api/astasilenziosa/richiedi	Richiede informazioni su un'asta silenziosa.
/api/utente/datiutente	Ottiene i dati dell'utente.
/api/utente/updateProfilo	Aggiorna il profilo dell'utente.
/api/astaribasso/crea	Crea un'asta al ribasso.
/api/astasilenziosa/crea	Crea un'asta silenziosa.
/api/asteribassoacquisto/richiedi	Ottiene informazioni sugli acquisti relativi all'asta al ribasso.

Continua alla pagina successiva

**Tabella 22 – continua nella pagina successiva**

Chiamate API	
/api/astesilenziosaofferta/richiedi	Ottiene informazioni sulle offerte relative all'asta silenziosa.
/api/asteribassoseguite/richiedi	Ottiene informazioni sulle aste al ribasso seguite dall'utente.
/api/astesilenziosaseguite/richiedi	Ottiene informazioni sulle aste silenziose seguite dall'utente.
/api/astesilenziosaofferta/richiediofferte	Ottiene informazioni sulle offerte relative a un'asta silenziosa.
/api/astesilenziosaofferta/updateOfferta	Aggiorna lo stato di un'offerta per un'asta silenziosa.
/api/astaribasso/searchVenditore	Effettua una ricerca per aste al ribasso di un venditore.
/api/astasilenziosa/searchVenditore	Effettua una ricerca per aste silenziose di un venditore.
/api/astaribasso/searchAcquirente	Effettua una ricerca per aste al ribasso di un acquirente.
/api/astasilenziosa/searchAcquirente	Effettua una ricerca per aste silenziose di un acquirente.
/api/astaribasso/richieditutte	Ottiene informazioni su tutte le aste al ribasso di un acquirente.
/api/astasilenziosa/richeditutte	Ottiene informazioni su tutte le aste silenziose di un acquirente.
/api/astesilenziosaofferta/creaOfferta	Crea un'offerta per un'asta silenziosa.
/api/asteribassoacquisto/acquisto	Effettua un acquisto per un'asta al ribasso.
/api/asteribassoseguite/seguì	Segue un'asta al ribasso.
/api/astesilenziosaseguite/seguì	Segue un'asta silenziosa.
/api/asteribassoseguite/seguitabool	Controlla se l'utente segue un'asta al ribasso.
/api/astesilenziosaseguite/seguitabool	Controlla se l'utente segue un'asta silenziosa.



Parte IV

Testing



14 Testing e valutazione sul campo dell'usabilità

In questa sezione ci occuperemo di software testing, ovvero di analisi dinamica e osservazione del comportamento del prodotto realizzato. In particolare svolgeremo testing a livello di unità (Unit testing) al fine di rilevare errori di logica o dati nel modulo. Verranno considerati i seguenti obiettivi:

- Dimostrazione che le classi funzionano correttamente.
- In presenza di bugs, dimostrazione che non si verificheranno nuovamente.
- Se il codice viene cambiato, dimostrazione che tutto funziona ancora correttamente.
- Se qualcosa non funziona più, vogliamo conoscerne la causa il prima possibile.

Al fine di creare l'ambiente adatto per l'esecuzione dei test, utilizzeremo JUnit, un utile framework per la generazione di scaffolding.

14.1 Individuazione e descrizione dei metodi

Metodo	Input	Output	Classe	Posizione	Descrizione
validateLogin	String email, String password	boolean	UtenteService	serverdd24/Service/	Verifica se l'email esiste nel database e se la password corrisponde.
validatePin	String email, Integer pin	boolean	UtenteService	serverdd24/Service/	Verifica se l'email esiste nel database, se il pin corrisponde e se non è scaduto.
caricaDatiJson	Context context, String filename	void	UtilsFunction	clientdd24/Utils	Utilizzato per caricare i dati delle Nazionalità, Città e Categorie.
savePreferences	String email, String password	void	Activity06Login	clientdd24/Views	Utilizzato per ricordare i valori di email e password per i successivi login.

Tabella 23: Metodi candidati per Unit Testing

14.2 Metodo validateLogin

Questo metodo, nonostante la sua semplicità, svolge un ruolo cruciale nell'autenticazione degli utenti, un aspetto fondamentale per la sicurezza di qualsiasi sistema software. Un errore in questa funzione potrebbe avere conseguenze gravi, come permettere l'accesso a utenti non autorizzati o negare l'accesso a utenti legittimi.



```

1 public boolean validateLogin(String email, String password) {
2     UtenteModel utenteModel = utenteRepository.findByEmailIgnoreCase(email)
3         .orElseThrow(() -> new ResponseStatusException(HttpStatus.NOT_FOUND, "
4             Utente non trovato per l'email fornita."));
5     if (!utenteModel.getPassword().equals(password)) {
6         throw new ResponseStatusException(HttpStatus.UNAUTHORIZED, "Credenziali non
7             valide.");
8     }
9     return true;
10 }
```

Individuazione delle classi di equivalenza

Tra le varie tecniche di testing, l'uso delle classi di equivalenza rappresenta un approccio efficace ed efficiente per la definizione dei casi di test.

Le classi di equivalenza si basano sull'idea che un insieme di input possa essere raggruppato in classi, all'interno delle quali ci si aspetta che il software si comporti in modo simile. Questo permette di ridurre il numero di casi di test necessari, mantenendo al contempo un'alta copertura del codice.

Sono state individuate quattro classi atte a coprire tutte le possibili combinazioni di input che il metodo *validateLogin* può ricevere, garantendo così una copertura completa del testing.

Nome Classe	Descrizione	Validità
[CEemail1]	Email esistenti nel database	Valida
[CEemail2]	Email non esistenti nel database	Non valida
[CEpassword1]	Password corrispondente all'email nel database	Valida
[CEpassword2]	Password non corrispondente all'email nel database	Non valida

Tabella 24: Classi di Equivalenza del metodo validateLogin

NOTA: Sono escluse le classi di email/password non conformi alle regex e/o NULL e/o Vuote in quanto il Client provvede ad inoltrare solo stringhe sintatticamente valide.

Strategia di Testing Black-box

La strategia adottata per il testing Black-box è N-WECT. L'approccio N-WECT ci permette di testare solo alcune delle combinazioni a cui siamo interessati. In particolare, le combinazioni di [CEemail2,CEpassword1] e [CEemail2,CEpassword2] non sono necessarie, poiché nel momento in cui l'utente non esiste sul database, qualunque classe di password porterebbe alla stessa condizione di utente non trovato. Uniremo queste condizioni in un'unica combinazione come rappresentato dalla tabella 25 nel seguente modo: [CEemail2,CEpasswordX], dove $X = 1 \text{ or } X = 2$.

TC n.	Combinazione	Descrizione	Validità
1	[CEemail1,CEpassword1]	Email esistente nel database, password corretta	Valida
2	[CEemail1,CEpassword2]	Email esistente nel database, password non corretta	Non valida
3	[CEemail2,CEpasswordX]	Email non esistente nel database, password qualsiasi	Non valida

Tabella 25: Combinazioni delle Classi di Equivalenza del metodo validateLogin



Ambiente di Test e Codice JUnit

Per testare adeguatamente il metodo `validateLogin`, è necessario creare un ambiente di test che simuli le condizioni reali in cui il metodo sarà utilizzato. Poiché questo metodo interagisce con un database per validare le credenziali degli utenti, dobbiamo popolare un database di test con dati fittizi per simulare scenari d'uso realistici.

Email	Password
email_esistente@test.com	Password1234corretta

Tabella 26: Dati di test per il metodo validateLogin presenti sul database

```
1  @Test
2  public void validateLoginTestCase1() {
3      String email = "email_esistente@test.com";
4      String password = "Password1234corretta";
5
6      boolean result = utenteService.validateLogin(email, password);
7      Assertions.assertTrue(result);
8 }
```

Listing 1: Test Case 1 per validateLogin

```
1  @Test
2  public void validateLoginTestCase2() {
3      testInvalidCredentials("email_esistente@test.com", "password_non_corretta",
4      "Credenziali non valide");
5
6
7      private void testInvalidCredentials(String email, String password, String
8      expectedMessage) {
9          Exception exception = assertThrows(ResponseStatusException.class, () ->
10         utenteService.validateLogin(email, password));
11
12         String actualMessage = exception.getMessage();
13         Assertions.assertTrue(actualMessage.contains(expectedMessage));
14     }
15 }
```

Listing 2: Test Case 2 per validateLogin

```
1  @Test
2  public void validateLoginTestCase3() {
3      testInvalidCredentials("email_non_esistente@test.com", "password_non_esistente",
4      "Utente non trovato per l'email fornita.");
5
6
7      private void testInvalidCredentials(String email, String password, String
8      expectedMessage) {
9          Exception exception = assertThrows(ResponseStatusException.class, () ->
10         utenteService.validateLogin(email, password));
11
12         String actualMessage = exception.getMessage();
13         Assertions.assertTrue(actualMessage.contains(expectedMessage));
14     }
15 }
```

Listing 3: Test Case 3 per validateLogin

Strategia di Testing White-box

Il testing white box, noto anche come testing strutturale, è un approccio al testing del software che si concentra sulla struttura interna del codice. A differenza del testing black box, che si concentra solo sui risultati di output per un dato input, il testing white box prende in considerazione come il software esegue una specifica operazione.

Nel contesto del metodo `validateLogin`, il testing white box ci permette di esaminare come il metodo gestisce le diverse classi di equivalenza e le combinazioni di queste. Possiamo verificare se tutte le possibili vie di esecuzione nel codice sono state testate, compresi i casi limite e le condizioni di errore.

Le strategie di testing white box più rilevanti per questo specifico metodo sono:

- **Node Coverage:** in questo modo ci assicuriamo che ogni riga di codice nel metodo viene eseguita durante il testing.
- **Branch Coverage:** data la presenza di predicati, questa strategia ci permette di verificare che ogni percorso decisionale sia eseguito per entrambi i risultati possibili (vero/falso).

La strategia di **Modified Condition Coverage** non è necessaria in questo contesto, in quanto il metodo in questione non ha molteplici condizioni che devono essere valutate insieme.

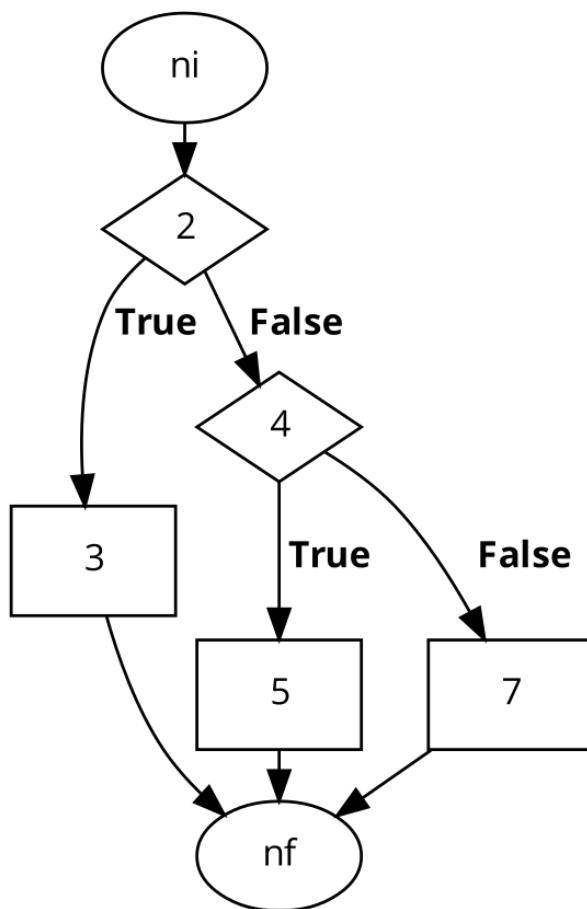


Figura 81: Control Flow Graph di `validateLogin`.



Node Coverage

TC n.	Oracolo	Insieme dei nodi coperti
1	Coprire il nodo 7	A={2,4,7}
2	Coprire il nodo 5	B={2,4,5}
3	Coprire il nodo 3	C={2,3,7}

Tabella 27: Node Coverage per il metodo validateLogin

Per calcolare il Test Effectiveness Ratio, dobbiamo prima unire gli insiemi dei nodi coperti dai tre casi di test. L'unione di questi insiemi è data da:

$$InsiemeTotale = A \cup B \cup C = \{2, 3, 4, 5, 7\}$$

Quindi, il Test Effectiveness Ratio è dato dal numero di nodi coperti diviso per il numero totale di nodi:

$$TestEffectivenessRatio = \frac{\#nodicoperti}{\#noditotali} = \frac{5}{5} = 1 = 100\%$$

Branch Coverage

Sono state individuate le seguenti condizioni:

1. 2-True e 2-False
2. 4-True e 4-False

Verifichiamo il livello di copertura delle condizioni con i Test Case identificati.

TC n.	Oracolo	Branch coperti
1	Coprire branch 2-False e 4-False	2-False e 4-False
2	Coprire branch 2-False e 4-True	2-False e 4-True
3	Coprire branch 2-True	2-True

Tabella 28: Branch Coverage per il metodo validateLogin

$$TestEffectivenessRatio = \frac{\#branchcoperti}{\#branchtotali} = \frac{2}{2} = 1 = 100\%$$

Possiamo concludere che i Test Case definiti per il metodo *validateLogin* sono risultati adeguati a coprire la totalità degli oggetti sia in Node Coverage che in Branch Coverage.

14.3 Metodo validatePin

Per aumentare la sicurezza del nostro prodotto, ogni utente che si troverà ad effettuare un cambio password dovrà passare per una procedura di validazione di un pin inoltrato tramite email. E' di fondamentale importanza testare questo metodo al fine di verificare che gli utenti con il pin corretto riescano ad effettuare il cambio password e d'altra parte, che utenti non in possesso di un pin valido non riescano a modificare la password dell'account. Inoltre, è necessario testare anche la sua validità temporale, garantendo che un PIN valido non possa essere riutilizzato dopo un certo periodo di tempo.



```

1 public boolean validatePin(String email, Integer pin) {
2     UtenteModel utenteModel = utenteRepository.findByEmailIgnoreCase(email)
3         .orElseThrow(() -> new ResponseStatusException(HttpStatus.NOT_FOUND, "
4             Utente non trovato per l'email fornita."));
5     Timestamp oraCorrente = new Timestamp(System.currentTimeMillis());
6     Timestamp limiteScadenza = new Timestamp(oraCorrente.getTime() - 60000);
7     if (!utenteModel.getpin().equals(pin)) {
8         throw new ResponseStatusException(HttpStatus.UNAUTHORIZED, "Pin non valido.");
9     }
10    } else if (utenteModel.getpin().equals(pin) && utenteModel.getTempopin().before(
11        limiteScadenza)) {
12        throw new ResponseStatusException(HttpStatus.UNAUTHORIZED, "Pin scaduto.");
13    }
14    return true;
15 }
```

Individuazione delle classi di equivalenza

Per il metodo *validatePin* sono state individuate 5 classi di equivalenza come mostrato nella tabella di seguito.

Nome Classe	Descrizione	Validità
[CEemail1]	Email esistente nel database	Valida
[CEemail2]	Email non esistente nel database	Non valida
[CEpin1]	Pin corrispondente all'email nel database e non scaduto	Valida
[CEpin2]	Pin non corrispondente all'email nel database	Non valida
[CEpin3]	Pin corrispondente all'email nel database ma scaduto	Non valida

Tabella 29: Classi di Equivalenza del metodo validatePin

NOTA: Sono escluse le classi di email/pin non conformi alle regex e/o NULL e/o Vuote in quanto il Client provvede ad inoltrare solo valori validi.

Strategia di Testing Black-box

La strategia adottata per il testing Black-box è N-WECT in quanto le combinazioni ottenibili con [CEemail2] e una tra le 3 classi di [CEpin] non sono necessarie, poiché nel momento in cui l'utente non esiste sul database, qualunque classe di pin porterebbe alla stessa condizione di utente non trovato. Uniremo queste condizioni in un'unica combinazione come rappresentato dalla tabella 30 nel seguente modo: [CEemail2,CEpinX], dove $X=1 \text{ or } X=2 \text{ or } X=3$.

TC n.	Combinazione	Descrizione	Oracolo
1	[CEemail1,CEpin1]	Email esistente nel database, pin corretto e non scaduto	True
2	[CEemail1,CEpin2]	Email esistente nel database, pin non corretto	"Pin non valido"
3	[CEemail1,CEpin3]	Email esistente nel database, pin corretto ma scaduto	"Pin scaduto"
4	[CEemail2,CEpinX]	Email non esistente nel database, pin qualsiasi	"Utente non trovato"

Tabella 30: Combinazioni delle Classi di Equivalenza del metodo validatePin



Ambiente di Test e Codice JUnit

Per testare adeguatamente il metodo `validatePin`, è essenziale creare una entry nel database per l'utente. In particolare siamo interessati solo alla sua email, poichè il pin di recupero e la data di scadenza non sono informazioni che conosciamo a priori ma vengono automaticamente generate.

```
1  @Test
2  public void validatePinTestCase1() {
3      String email = "email_esistente@test.com";
4
5      Integer pin = utenteService.inviaCodiceRecuperoTest(email);
6
7      boolean result = utenteService.validatePin(email, pin);
8      Assertions.assertTrue(result);
9 }
```

Listing 4: Test Case 1 per validatePin

```
1  @Test
2  public void validatePinTestCase2() {
3      String email = "email_esistente@test.com";
4
5      Integer pin = 0;
6
7      Exception exception = assertThrows(ResponseStatusException.class, () ->
8          utenteService.validatePin(email, pin));
9
10     String expectedMessage = "Pin non valido.";
11     String actualMessage = exception.getMessage();
12
13     Assertions.assertTrue(actualMessage.contains(expectedMessage));
14 }
```

Listing 5: Test Case 2 per validatePin

```
1  @Test
2  public void validatePinTestCase3() {
3      String email = "email_esistente@test.com";
4
5      Integer pin = utenteService.inviaCodiceRecuperoTest(email);
6
7      Timestamp now = new Timestamp(System.currentTimeMillis() + (60 * 1000));
8      Timestamp expiredTime = new Timestamp(now.getTime() + 61000); // 61000
9      millisecondi = 61 secondi
10
11      // Act and Assert
12      assertThrows(ResponseStatusException.class, () -> utenteService.
13          validatePinTest(email, pin, expiredTime), "Expected validatePin to throw, but it
14          didn't");
15 }
```

Listing 6: Test Case 3 per validatePin

```
1  @Test
2  public void validatePinTestCase4() {
3      String email = "email_non_esistente@test.com";
4      Integer pin = 123456; // Un pin qualsiasi
5
6      Exception exception = assertThrows(ResponseStatusException.class, () ->
7          utenteService.validatePin(email, pin));
8
9      String expectedMessage = "Utente non trovato per l'email fornita.";
```

```

9     String actualMessage = exception.getMessage();
10    Assertions.assertTrue(actualMessage.contains(expectedMessage));
11 }

```

Listing 7: Test Case 4 per validatePin

Strategia di Testing White-box

Le strategie di testing white box più rilevanti per questo specifico metodo sono:

- **Node Coverage:** in questo modo ci assicuriamo che ogni riga di codice nel metodo viene eseguita durante il testing.
- **Branch Coverage:** data la presenza di predici, questa strategia ci permette di verificare che ogni percorso decisionale sia eseguito per entrambi i risultati possibili (vero/falso).
- **Modified Condition Coverage:** data la presenza di un predico in cui devono essere soddisfatte due condizioni tramite l'operatore logico AND (`&&`), risulta necessario effettuare anche questo tipo di test per accertarci che ogni possibile valore delle due proposizioni venga coperto.

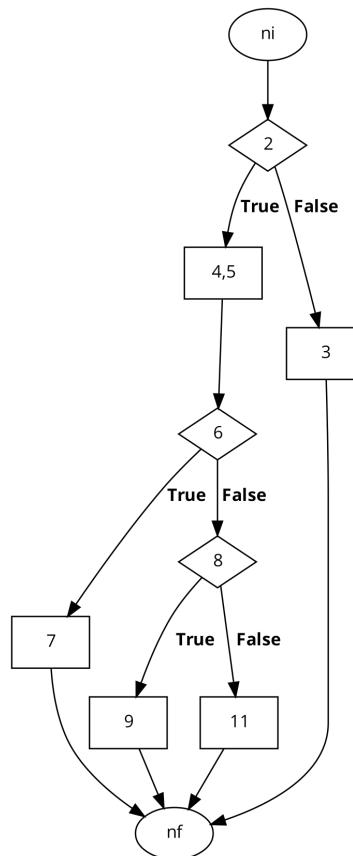


Figura 82: Control Flow Graph di validatePin.



Node Coverage

TC n.	Oracolo	Insieme dei nodi coperti
1	Coprire nodo 11	A={2,4,5,6,8,11}
2	Coprire nodo 7	B={2,4,5,6,7}
3	Coprire nodo 9	C={2,4,5,6,8,9}
4	Coprire nodo 3	D={2,3}

Tabella 31: Node Coverage per il metodo validatePin

$$InsiemeTotale = A \cup B \cup C \cup D = \{2, 3, 4, 5, 6, 7, 8, 9, 11\}$$

$$TestEffectivenessRatio = \frac{\#\text{nodicoperti}}{\#\text{nodi totali}} = \frac{9}{9} = 1 = 100\%$$

Branch Coverage

Sono state individuate le seguenti condizioni:

1. 2-True e 2-False
2. 6-True e 6-False
3. 8-True e 8-False

Verifichiamo il livello di copertura delle condizioni con i Test Case identificati.

TC n.	Oracolo	Branch coperti
1	Coprire branch 2-True e 8-False	2-True, 6-False, 8-False
2	Coprire branch 2-True e 6-True	2-True, 6-True
3	Coprire branch 2-True e 8-True	2-True, 6-False, 8-True
4	Coprire branch 2-False	2-False

Tabella 32: Branch Coverage per il metodo validatePin

$$TestEffectivenessRatio = \frac{\#\text{branchcoperti}}{\#\text{branchtotali}} = \frac{3}{3} = 1 = 100\%$$

Modified Condition Coverage

Amdremo a valutare tutte le possibili coppie dei valori di verità di "Pin corrispondente" e "Pin scaduto" (Pc,Ps), relativi al branch 8, in particolare:

1. (True,True)
2. (False,False)



3. (True,False)

4. (False,True)

TC n.	Oracolo	Risultato ottenuto	Coppia coperta
1	True	Corrispondente all'Oracolo	(True,False)
2	"Pin non valido"	/	Nessuna
3	"Pin scaduto"	Corrispondente all'Oracolo	(True,True)
4	"Utente non trovato per l'email fornita"	/	Nessuna

Tabella 33: Modified Condition Coverage per il metodo validatePin

Non abbiamo raggiunto la copertura per tutte le coppie ma questo è dovuto solo al fatto che quando la proposizione "Pin corrispondente" assume valore "False", il branch 8 non viene proprio raggiunto in quanto questa condizione è gestita singolarmente dal branch 6.

14.4 Metodo caricaDatiJson

Testare questo metodo risulta estremamente importante in quanto ha un impatto diretto su possibili nuovi utenti della piattaforma. In particolare, il metodo si occupa di caricare i dati di Nazionalità, Città e Categorie da un file Json. Il mancato corretto caricamento risulterebbe nell'impossibilità di completare la fase di registrazione, in cui Nazionalità e Città sono obbligatorie, oppure la non corretta visualizzazione delle categorie.

```

1  public JSONObject caricaDatiDaJson(Context context, String filename) {
2      if (context != null && filename != null) {
3          AssetManager assetManager = context.getAssets();
4          try (InputStream is = assetManager.open(filename)) {
5              int size = is.available();
6              byte[] buffer = new byte[size];
7              int bytesRead = is.read(buffer);
8              if (bytesRead != -1) {
9                  String jsonStr = new String(buffer, 0, bytesRead,
10                     StandardCharsets.UTF_8);
11                 return new JSONObject(jsonStr);
12             } else {
13                 Log.e("SignupActivity1", "Nessun dato letto dal file");
14                 Toast.makeText(context, "Errore nel caricamento dei dati", Toast.LENGTH_LONG).show();
15             }
16         } catch (IOException e) {
17             Log.e("SignupActivity1", "Errore nella lettura del file", e);
18             Toast.makeText(context, "Errore nel caricamento dei dati", Toast.LENGTH_LONG).show();
19         } catch (JsonSyntaxException | JsonIOException | JSONException e) {
20             Log.e("SignupActivity1", "Errore nel parsing del JSON", e);
21             Toast.makeText(context, "Errore nel parsing dei dati", Toast.LENGTH_LONG).show();
22         }
23     }
24     return null;
25 }
```



Individuazione delle classi di equivalenza

Per il metodo *caricaDatiDaJson* sono state individuate le seguenti classi di equivalenza mostrate in tabella.

Nome Classe	Descrizione	Validità
[CEcontext1]	Context valido	Valida
[CEcontext2]	Context null	Non valida
[CEfilename1]	File esistente e codice Json valido	Valida
[CEfilename2]	File non esistente	Non valida
[CEfilename3]	File null	Non valida
[CEfilename4]	File esistente e codice Json non valido	Non valida

Tabella 34: Classi di Equivalenza del metodo caricaDatiDaJson

Strategia di Testing Black-box

Nel contesto del metodo *caricaDatiDaJson*, la strategia N-SECT potrebbe essere la scelta più adeguata. Il motivo è che questo metodo ha solo due parametri e le interazioni tra le classi di equivalenza non sono di forte interesse. Pertanto, N-SECT fornirebbe una copertura di test adeguata senza la necessità di un numero eccessivo di casi di test.

TC n.	Classe	Descrizione	Oracolo
1	[CEcontext1]	Context valido	JSONObject
2	[CEcontext2]	Context null	null
3	[CEfilename1]	File esistente, Json valido	JSONObject
4	[CEfilename2]	File non esistente	IOException
5	[CEfilename3]	File null	null
6	[CEfilename4]	File esistente e codice Json non valido	JSONException

Tabella 35: Test Cases del metodo caricaDatiDaJson

Codice JUnit

```

1  @Test
2  public void caricaDatiDaJsonTestCase1() {
3      // Context valido, File esistente, Json valido
4      UtilsFunction obj = new UtilsFunction();
5      JSONObject result = obj.caricaDatiDaJson(mockContext, "nazioni_e_citta.json"
6  );
7      assertNotNull(result);
}

```

Listing 8: Test Case 1 per caricaDatiDaJson



```
2     public void caricaDatiDaJsonTestCase2() {
3         // Context null, File esistente, Json valido
4         UtilsFunction obj = new UtilsFunction();
5         assertThrows(NullPointerException.class, () -> obj.caricaDatiDaJson(null, "nazioni_e_citta.json"));
6     }
```

Listing 9: Test Case 2 per caricaDatiDaJson

```
1 @Test
2     public void caricaDatiDaJsonTestCase3() {
3         // Context valido, File non esistente
4         UtilsFunction obj = new UtilsFunction();
5         assertThrows(IOException.class, () -> obj.caricaDatiDaJson(mockContext, "nonExistingFile.json"));
6     }
```

Listing 10: Test Case 3 per caricaDatiDaJson

```
1 @Test
2     public void caricaDatiDaJsonTestCase4() {
3         // Context valido, File null
4         UtilsFunction obj = new UtilsFunction();
5         assertThrows(NullPointerException.class, () -> obj.caricaDatiDaJson(
6             mockContext, null));
    }
```

Listing 11: Test Case 4 per caricaDatiDaJson

```
1 @Test
2     public void caricaDatiDaJsonTestCase5() {
3         // Context valido, File esistente, Json non valido
4         UtilsFunction obj = new UtilsFunction();
5         assertThrows(JSONException.class, () -> obj.caricaDatiDaJson(mockContext, "invalid.json"));
6     }
```

Listing 12: Test Case 5 per caricaDatiDaJson

Strategia di Testing White-box

Il metodo in analisi contiene diverse istruzioni catch che gestiscono vari tipi di eccezioni. Nel contesto del testing, i blocchi *try-catch* possono essere considerati come punti di decisione simili agli *if-else*. Ogni blocco *catch* rappresenta un percorso diverso che il codice può prendere in risposta a una eccezione. Quindi tramite la strategia di Branch-Coverage ci assicuriamo che ogni ramo venga eseguito almeno una volta, verificando che le eccezioni vengano gestite correttamente.

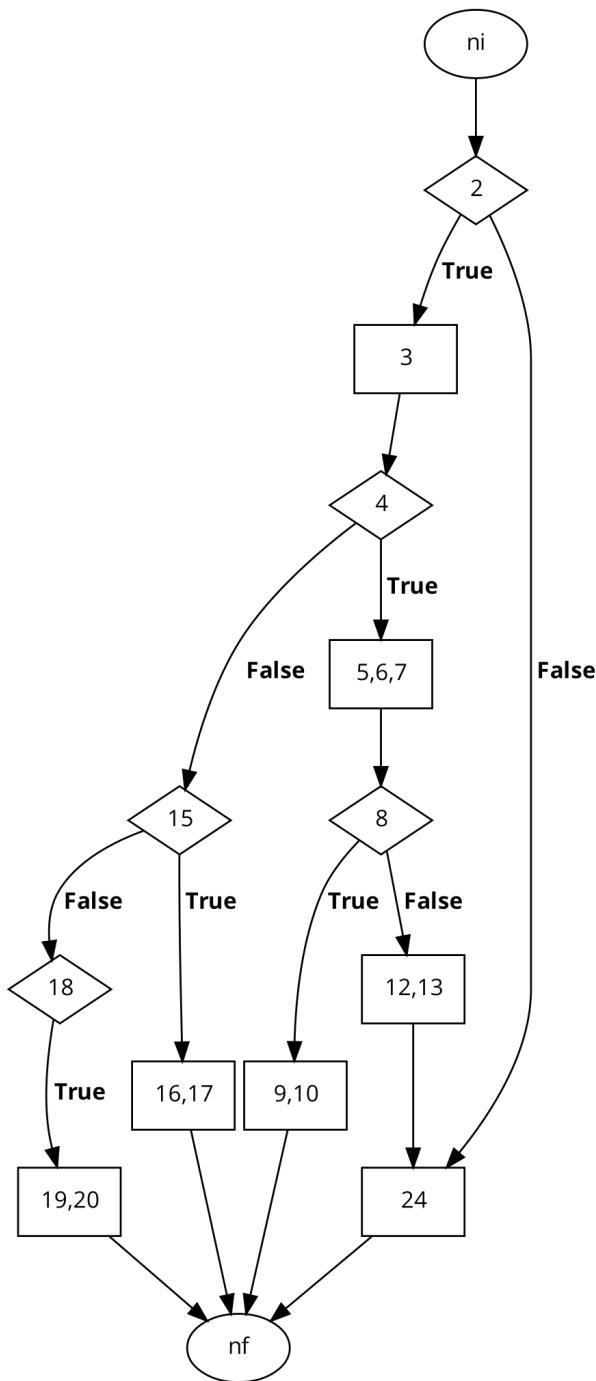


Figura 83: Control Flow Graph di caricaDatiDaJson.

Branch Coverage

Sono stati individuati 3 branch di interesse per i nostri test, di cui una condizione *if-else* e due condizioni catch:

1. 2-True e 2-False
2. 15-Catch per IOException
3. 18-Catch per JSONException



TC n.	Oracolo	Branch Coperti
1	Coprire branch 2-True	2,4,8 oppure 2,4,15 oppure 2,4,15,18
2	Coprire catch 15	2,4,15
3	Coprire branch 2-False	2
4	Coprire catch 18	2,4,15,18

Tabella 36: Branch Coverage per il metodo caricaDatiDaJson

$$TestEffectivenessRatio = \frac{\#branchcoperti}{\#branchtotali} = \frac{3}{3} = 1 = 100\%$$

14.5 Metodo savePreferences

Risulta importante svolgere il testing di questo metodo in quanto in caso di problemi, i prodotti non mostrerebbero alcuni tipi di informazioni.

```

1 private void savePreferences(String email, String password, boolean
2     isRicordamiChecked) {
3     SharedPreferences sharedpreferences = getSharedPreferences("MyAppPreferences",
4         Context.MODE_PRIVATE);
5     SharedPreferences.Editor editor = sharedpreferences.edit();
6
7     if (email == null || email.isEmpty() || password == null || password.isEmpty())
8     {
9         // Se email o password sono null o vuote, non salviamo le preferenze
10        return;
11    }
12
13    if (isRicordamiChecked) {
14        editor.putString("email", email);
15        editor.putString("Password", password);
16        editor.putBoolean("Ricordami", true);
17    } else {
18        editor.remove("email");
19        editor.remove("Password");
20        editor.putBoolean("Ricordami", false);
21    }
22
23    editor.apply();
24 }
```

Individuazione delle classi di equivalenza

Per il metodo *savePreferences* sono state individuate le seguenti classi di equivalenza mostrate in tabella.



Nome Classe	Descrizione	Validità
[CEemail1]	Stringa non null e non vuota	Valida
[CEemail2]	Stringa null	Non valida
[CEemail3]	Stringa vuota	Non valida
[CEpassword1]	Stringa non null e non vuota	Valida
[CEpassword2]	Stringa null	Non valida
[CEpassword3]	Stringa vuota	Non valida
[CERicordami1]	Valore True	Valida
[CERicordami2]	Valore False	Valida

Tabella 37: Classi di Equivalenza del metodo savePreferences

Strategia di Testing Black-box

Per questo metodo utilizzeremo la strategia di Testing Black-box N-SECT.

TC n.	Classe	Descrizione	Oracolo
1	[CEmail1]	Stringa ‘email’ valida	Preferenze salvate
2	[CEmail2]	Stringa ‘email’ null	Nessuna azione
3	[CEmail3]	Stringa ‘email’ vuota	Nessuna azione
4	[CEpassword1]	Stringa ‘password’ valida	Preferenze salvate
5	[CEpassword2]	Stringa ‘password’ null	Nessuna azione
6	[CEpassword3]	Stringa ‘password’ vuota	Nessuna azione
7	[CERicordami1]	‘isRicordamiChecked’ è true	Preferenze salvate
8	[CERicordami2]	‘isRicordamiChecked’ è false	Preferenze rimosse

Tabella 38: Test Cases del metodo savePreferences

Codice JUnit

Per verificare la corretta annotazione dei risultati è stato necessario utilizzare il framework Mockito.

```

1  @Before
2  public void setUp() {
3      MockitoAnnotations.openMocks(this);
4      when(context.getSharedPreferences(anyString(), anyInt())).thenReturn(
5          sharedPreferences);
6      when(sharedPreferences.edit()).thenReturn(editor);
7
7      when(editor.putString(eq("email"), anyString())).thenReturn(editor);
8      when(editor.putString(eq("Password"), anyString())).thenReturn(editor);
9      when(editor.putBoolean(eq("Ricordami"), anyBoolean())).thenReturn(editor);
10     when(editor.remove(anyString())).thenReturn(editor);
11     doNothing().when(editor).apply();
12 }
```



13 }

Listing 13: setup per savePreferences

```
1 @Test
2 public void savePreferencesTestCase1() {
3     PreferencesManager.savePreferences(context, "[CEmail1]", "[CEpassword1]",
4     true);
5
6     verify(editor).putString("email", "[CEmail1]");
7     verify(editor).putString("Password", "[CEpassword1]");
8     verify(editor).putBoolean("Ricordami", true);
9     verify(editor).apply();
}
```

Listing 14: Test Case 1 per savePreferences

```
1 @Test
2 public void savePreferencesTestCase2() {
3     PreferencesManager.savePreferences(context, null, "[CEpassword2]", true);
4
5     verify(editor, never()).putString(anyString(), anyString());
6     verify(editor, never()).putBoolean(anyString(), anyBoolean());
7     verify(editor, never()).apply();
8 }
```

Listing 15: Test Case 2 per savePreferences

```
1 @Test
2 public void savePreferencesTestCase3() {
3     PreferencesManager.savePreferences(context, "[CEmail3]", "[CEpassword3]",
4     false);
5
6     verify(editor).putString("email", "[CEmail3]");
7     verify(editor).putString("Password", "[CEpassword3]");
8     verify(editor).putBoolean("Ricordami", false);
9     verify(editor).apply();
}
```

Listing 16: Test Case 3 per savePreferences

```
1 @Test
2 public void savePreferencesTestCase4() {
3     PreferencesManager.savePreferences(context, "[CEmail1]", "[CEpassword1]",
4     false);
5
6     verify(editor).putString("email", "[CEmail1]");
7     verify(editor).putString("Password", "[CEpassword1]");
8     verify(editor).putBoolean("Ricordami", false);
9     verify(editor).apply();
}
```

Listing 17: Test Case 4 per savePreferences

```
1 @Test
2 public void savePreferencesTestCase5() {
3     PreferencesManager.savePreferences(context, "[CEmail1]", "", true);
4
5     verify(editor, never()).putString(anyString(), anyString());
6     verify(editor, never()).putBoolean(anyString(), anyBoolean());
7     verify(editor, never()).apply();
8 }
```

Listing 18: Test Case 5 per savePreferences



```
1  @Test
2  public void savePreferencesTestCase6() {
3      PreferencesManager.savePreferences(context, "", "[CEpassword2]", true);
4
5      verify(editor, never()).putString(anyString(), anyString());
6      verify(editor, never()).putBoolean(anyString(), anyBoolean());
7      verify(editor, never()).apply();
8 }
```

Listing 19: Test Case 6 per savePreferences

```
1  @Test
2  public void savePreferencesTestCase7() {
3      PreferencesManager.savePreferences(context, "", "", true);
4
5      verify(editor, never()).putString(anyString(), anyString());
6      verify(editor, never()).putBoolean(anyString(), anyBoolean());
7      verify(editor, never()).apply();
8 }
```

Listing 20: Test Case 7 per savePreferences

Strategia di Testing White-box

Il metodo in questione contiene una struttura di controllo *if-else* che crea diversi percorsi di esecuzione nel codice. La strategia di Branch Coverage è particolarmente adatta in questo caso, poiché si concentra sul garantire che ogni possibile percorso decisionale nel codice venga eseguito almeno una volta durante i test.

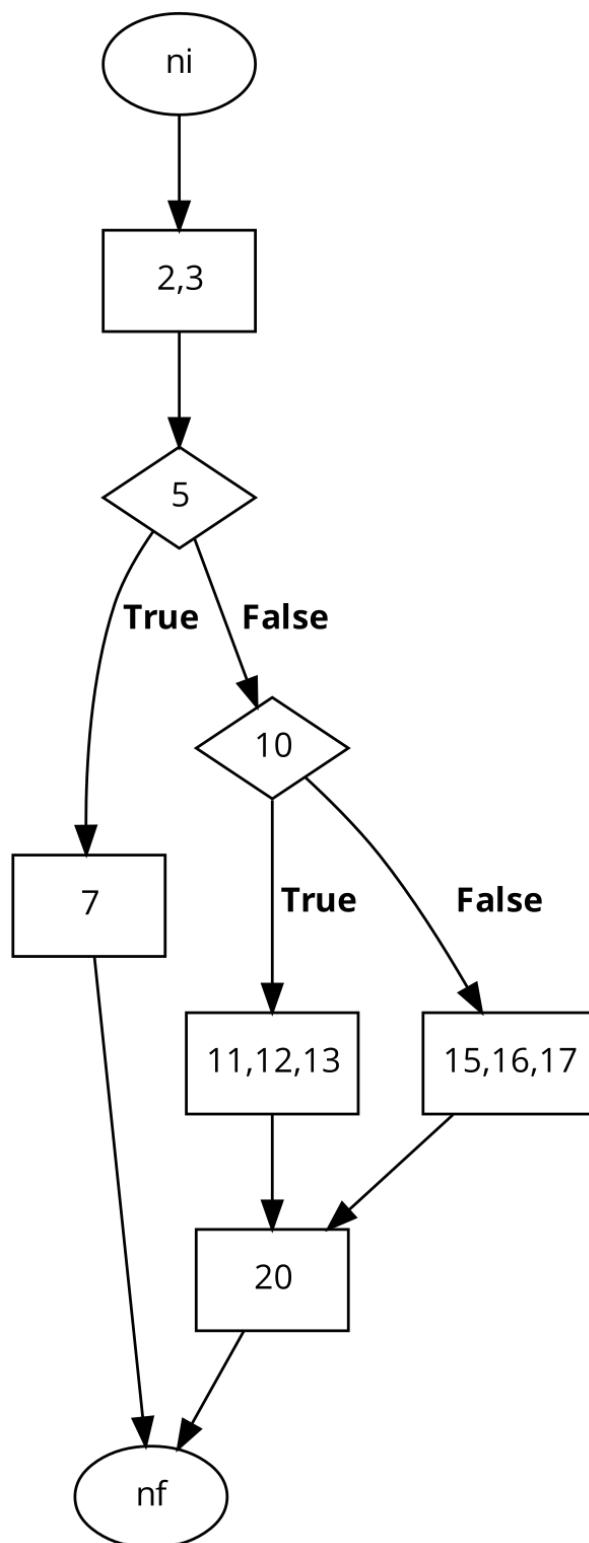


Figura 84: Control Flow Graph di savePreferences.

Branch Coverage

Sono stati individuati i seguenti branch di interesse:

1. 5-True e 5-False

2. 10-True e 10-False

TC n.	Oracolo	Branch Coperti
1	Coprire branch 5-False	5-False e 10
2	Coprire branch 5-True	5-True
3	Coprire branch 5-True	5-True
4	Coprire branch 5-False	5-False e 10
5	Coprire branch 5-True	5-True
6	Coprire branch 5-False	5-False
7	Coprire branch 10-True	5-False e 10-True
8	Coprire branch 10-False	5-False e 10-False

Tabella 39: Branch Coverage per il metodo savePreferences

$$TestEffectivenessRatio = \frac{\#branchcoperti}{\#branchtotali} = \frac{2}{2} = 1 = 100\%$$

15 Valutazione dell'usabilità sul campo

Per condurre uno studio di usabilità completo sull'applicazione android rilasciata, abbiamo adottato un approccio multidimensionale, utilizzando sia tecniche di studio di usabilità a priori descritte nella sezione 1.8, sia gli strumenti di logging integrati nell'applicazione Android (Firebase) per raccogliere dati sul comportamento degli utenti. Questo approccio ci ha permesso di ottenere una visione completa e dettagliata dell'esperienza utente.

Personas



Maria Rossi
Graphic Designer Freelance
"La creatività è l'intelligenza che si diverte."

Biografia
Maria, 30 anni, è una graphic designer freelance con una passione per l'arte e il design. Lavora da casa a Milano e collabora con diversi clienti internazionali. Vive da sola, appassionata di tecnologia e design moderno.

Hobby e interessi
Fotografia, viaggi, design d'interni.

Oggettivi
Trovare pezzi unici di design per arredare il suo studio casalingo.

Perché è un potenziale utente

- Ricerca di Oggetti Unici: Come designer, Maria è sempre alla ricerca di oggetti d'arte, mobili o accessori unici per il suo studio o per ispirazione. DietDeals24, con la sua varietà di aste, può offrire una gamma eclettica di oggetti che soddisfano il suo gusto per l'estetica e il design.
- Oportunità di Business: Potrebbe anche usare la piattaforma per vendere i suoi design o opere d'arte, raggiungendo un pubblico più ampio.

Figura 85: Personas n.1, Maria Rossi



Sara Conti
Proprietaria di un negozio di antiquariato
"Il passato è un prologo."

Biografia
Sara, 45 anni, gestisce un negozio di antiquariato a Firenze. Ha un forte interesse per la storia e gli oggetti vintage. Appassionata di storia dell'arte, ama restaurare mobili antichi.

Hobby e interessi
Collezionare oggetti antichi, viaggiare, leggere.

Oggettivi
Trovare oggetti rari e unici per il suo negozio.

Perché è un potenziale utente

- Acquisizione di Oggetti Rari e Antichi: La piattaforma può essere un ottimo canale per Sara per trovare e acquisire pezzi rari e antichi per il suo negozio. Le aste online sono luoghi ideali per scoprire oggetti unici che potrebbero non essere disponibili altrove.
- Espansione del Network di Vendita: Sara potrebbe utilizzare DietDeals24 per vendere alcuni dei suoi articoli, raggiungendo un pubblico più vasto e diversificato rispetto al solo negozio fisico.

Figura 86: Personas n.2, Sara Conti



Giovanni Verdi
Studente universitario in Ingegneria
"Sogna in grande e osa fallire."

Biografia
Giovanni, 22 anni, è uno studente di Ingegneria a Torino. Vive in un appartamento condiviso con altri studenti.

Hobby e interessi
Tecnologia, sport, musica

Oggettivi
Acquistare attrezzature tecniche e libri a prezzi accessibili.

Perché è un potenziale utente

- Accesso a Materiali Educativi e Tecnologici a Prezzi Accessibili: Giovanni potrebbe utilizzare la piattaforma per acquistare libri di testo, attrezzature per il suo corso di studio o persino gadget tecnologici a prezzi accessibili, adatti al suo budget da studente.
- Vendita di Libri e Materiale Usato: Inoltre, può vendere i suoi libri di testo o altri articoli non più necessari per recuperare parte dei costi e aiutare altri studenti come lui.

Figura 87: Personas n.3, Giovanni Verdi



Luca Bianchi
Dirigente in un'azienda IT
"Ogni problema è un'opportunità in maschera."

Biografia
Luca, 38 anni, lavora come dirigente IT a Roma. È sposato e ha due figli. AMA la tecnologia e si tiene sempre aggiornato sulle ultime novità.

Hobby e interessi
Gaming, elettronica, programmare.

Oggettivi
Acquistare gadget elettronici e giochi al miglior prezzo.

Perché è un potenziale utente

- Gadget e Tecnologia a Buon Prezzo: Luca, essendo un appassionato di tecnologia e gaming, potrebbe utilizzare DietDeals24 per trovare gadget e giochi elettronici a prezzi vantaggiosi, approfittando delle aste per ottenere il miglior affare possibile.
- Vendita di Articoli Usati: Potrebbe inoltre vendere articoli tecnologici o elettronici usati, rendendo la piattaforma un'ottima scelta per riciclare prodotti in modo efficace.

Figura 88: Personas n.4, Luca Bianchi

Nelle seguenti tabelle sono riportati i compiti sotto osservazione e i risultati che ci aspettiamo vengano raggiunti:



COMPITO	COMPITI GENERALI	CLICK STIMATI	TEMPO STIMATO(s)
1	Registrazione	15-22	85-125s
2	Accesso	3-4	40s
3	Recupero password	8	80s
4	Modifica del profilo	2-17	40-120s
5	Modifica Password	5-6	60s
6	Visualizza acquisti	2	10s
7	Visualizza vendite	2	10s
8	Visualizza aste seguite	2	10s

Tabella 40: Tabella Compiti Generali

COMPITO	COMPITI ACQUIRENTE	CLICK STIMATI	TEMPO STIMATO(s)
9	Presenta offerta asta silenziosa	5	50s
10	Acquista da asta al ribasso	3	20s
11	Visualizza aste	2	10s
12	Ricerca una specifica asta	5	45s
13	Visualizza profilo venditore	2	15s

Tabella 41: Tabella Compiti Acquirente

COMPITO	COMPITI VENDITORE	CLICK STIMATI	TEMPO STIMATO(s)
14	Crea asta al ribasso	14	130s
15	Crea asta silenziosa	10	90s
16	Visualizza aste create	2	20s

Tabella 42: Tabella Compiti Venditore

Risultati

Le nostre analisi hanno prodotto i seguenti risultati:

Utente	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 1	15	15	100	1	22	22	130	1	15	15	110	1	15	15	110	1
Compito 2	4	4	35	1	4	4	40	1	4	4	40	1	4	4	37	1
Compito 3	8	8	80	1	8	8	75	1	8	8	85	1	8	8	82	1
Compito 4	15	15	110	1	2	2	120	1	15	15	115	1	15	15	110	1
Compito 5	5	5	60	1	5	5	55	1	5	5	55	1	5	5	65	1
Compito 6	2	2	10	1	2	2	10	1	2	2	10	1	2	2	12	1
Compito 7	2	2	10	1	2	2	10	1	2	2	10	1	2	2	10	1
Compito 8	2	2	10	1	2	2	10	1	2	2	10	1	2	2	10	1

Tabella 43: Calcolo Usabilità UI Prodotto finale Compiti Utente

Acquirente	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 9	5	5	40	1	5	5	42	1	5	5	42	1	5	5	35	1
Compito 10	3	3	25	1	3	3	27	1	3	3	23	1	3	3	20	1
Compito 11	2	2	10	1	2	2	12	1	2	2	10	1	2	2	10	1
Compito 12	5	5	40	1	5	5	40	1	5	5	38	1	5	5	36	1
Compito 13	2	2	20	1	2	2	25	1	2	2	22	1	2	2	20	1

Tabella 44: Calcolo Usabilità Prodotto finale Compiti Acquirente

Venditore	Maria				Sara				Giovanni				Luca			
Compito	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E	CC	CE	T	E
Compito 14	14	14	140	1	14	15	110	1	14	19	125	0.5	14	14	117	1
Compito 15	10	10	80	1	10	10	90	1	10	10	75	1	10	10	84	1
Compito 16	2	2	20	1	2	2	15	1	2	2	15	1	2	2	20	1

Tabella 45: Calcolo Usabilità UI Prodotto finale Compiti Venditore

In base ai dati raccolti, abbiamo elaborato i seguenti risultati:

	Success Rate	Time ON Task	Users Error	Efficiency	Learnability
MARIA	16	49.38	0	0.32	1
SARA	16	50.69	0.06	0.32	1
GIOVANNI	15	49.06	0.31	0.31	1
LUCA	16	48.63	0	0.33	1
ASPETTATIVE	16	47.81	2	0.33	0.90
MEDIA	15.75	49.44	0.09	0.32	1

Tabella 46: Calcolo Usabilità prodotto finito - Risultati totali

Firebase

Durante l'esecuzione dei test di usabilità, abbiamo implementato la registrazione automatica degli eventi utilizzando il framework Google Analytics for Mobile Apps. Questo ci ha fornito un'analisi dettagliata sull'utilizzo del software da parte degli utenti. Di seguito, sono riportati estratti delle statistiche di utilizzo, consultabili tramite la console di Firebase.

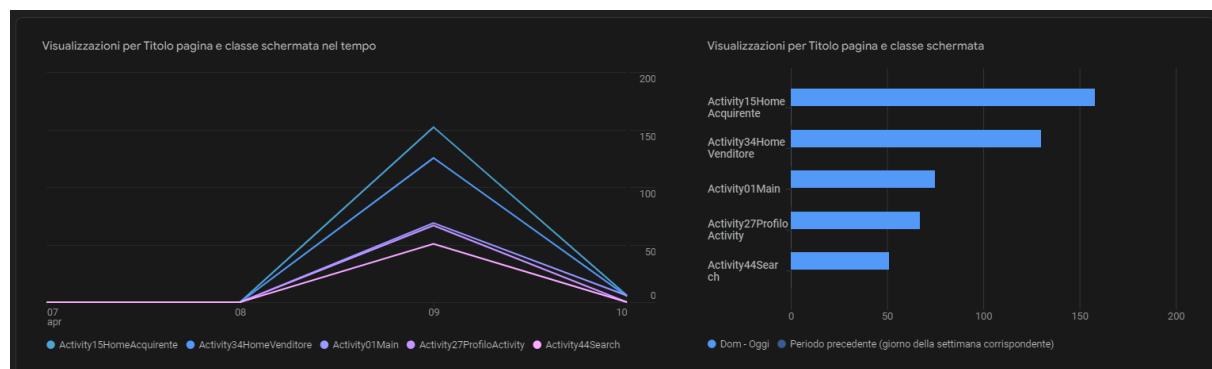


Figura 89: Firebase - Monitoraggio schermate



Indice

Parte IV

Cerca...

	Titolo pagina e...asse schermata	+	Visualizzazioni	Utenti	Visualizzazioni per utente	Durata media del coinvolgimento	Conteggio eventi	Conversioni	Entrate totali
莫斯拉互动 - PRINCIPALE									
1	Activity15HomeAcquirente		776 rispetto a 0	10 rispetto a 0	77,60 rispetto a 0,00	9 m 48 s rispetto a 0,00	1.350 rispetto a 0	9,00 rispetto a 0,00	0,00 \$ rispetto a 0,00 \$
2	Activity34HomeVenditore								
3	Activity01Main								
4	Activity27ProfiloActivity								
5	Activity44Search								
6	Activity41DettaglioASAperta								
7	Activity17AsteAcquirente								
8	Activity18AsteAcquirente								
9	Activity21DettaglioAstaSilenziosa								
10	Activity35AsteRibassoVenditore								

Figura 90: Firebase - Schermate più utilizzate

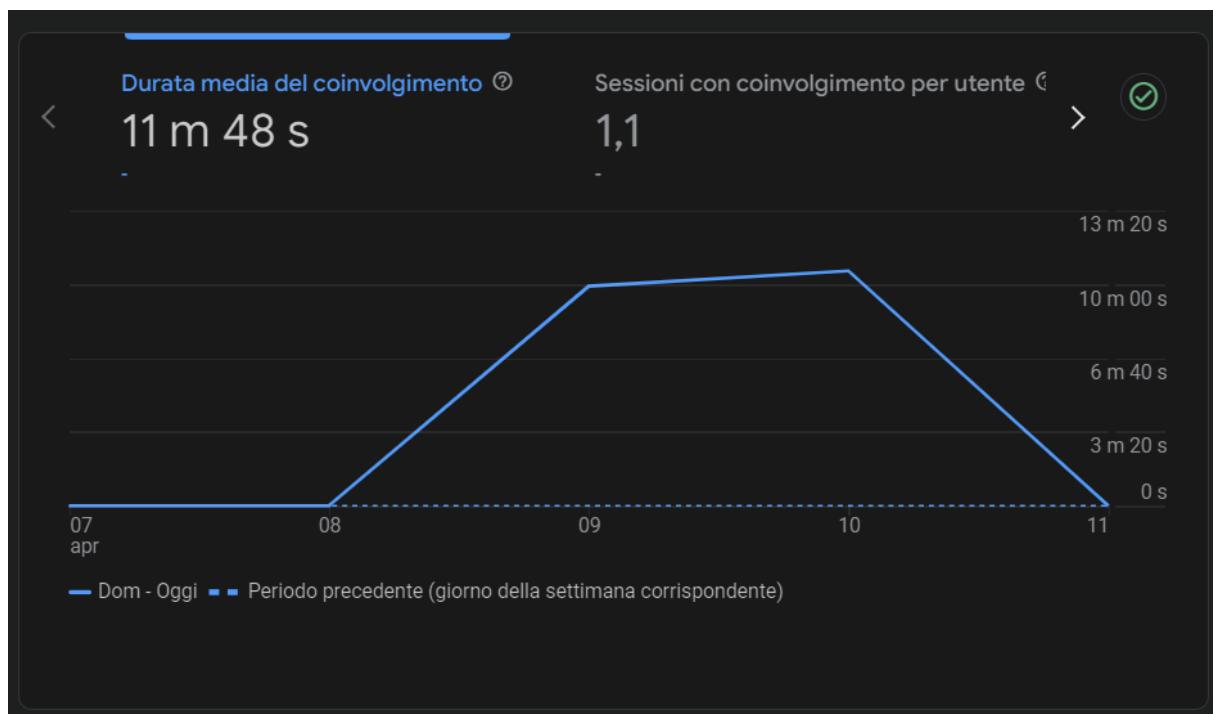


Figura 91: Firebase - Tempo medio coinvolgimento

File Log di esempio

```

1 Cliccato 'registrati' in 05welcome
2 Cliccato 'continua' in 07signup
3 Cliccato 'continua' in 08signup
4 Cliccato 'accetta terms' in 09signup
5 Cliccato 'registra' in 09signup
6 Cliccato 'accedi' in 06login
7 Cliccato 'segui/seguita' in 19dettaglio
8 Cliccato 'acquista' in 19dettaglio
9 Cliccato 'indietro' in 24dettaglio
10 Cliccato 'vedi tutte' silenziose in 15home

```



- 11 Cliccato 'Tutti' in 18aste
- 12 Cliccato 'Casa e Giardino' in 18aste
- 13 Cliccato 'CardSilenziosaApertaAcquirente' in 18aste
- 14 Cliccato 'aggiunta rapida' in 21dettaglio
- 15 Cliccato 'piu/meno' in 21dettaglio
- 16 Cliccato 'piu/meno' in 21dettaglio
- 17 Cliccato 'piu/meno' in 21dettaglio
- 18 Cliccato 'offri' in 21dettaglio
- 19 Cliccato 'segui/seguita' in 21dettaglio
- 20 Cliccato 'indietro' in 21dettaglio
- 21 Cliccato 'Casa e Giardino' in 18aste
- 22 Cliccato 'CardSilenziosaApertaAcquirente' in 18aste
- 23 Cliccato 'aste ribasso' in 18aste
- 24 Cliccato 'Casa e Giardino' in 17aste
- 25 Cliccato 'Sport e Tempo Libero' in 17aste
- 26 Cliccato 'CardRibassoApertaAcquirente' in 17aste
- 27 Cliccato per fare una ricerca in 17aste
- 28 Cliccato 'CardListenerRibassoApertaAcquirente' in 44search
- 29 Cliccato 'setupCardListenerSilenziosaApertaAcquirente' in 44search
- 30 Cliccato 'setupCardListenerSilenziosaApertaAcquirente' in 44search
- 31 Cliccato 'home' in bottom navigation in 44search
- 32 Cliccato 'vedi tutte' categorie in 15home
- 33 Cliccato 'Collezionismo' in 17aste
- 34 Cliccato 'CardRibassoApertaAcquirente' in 17aste
- 35 Cliccato 'aste silenziosa' in 17aste
- 36 Cliccato 'Collezionismo' in 18aste
- 37 Cliccato 'CardSilenziosaApertaAcquirente' in 18aste
- 38 Cliccato 'CardSilenziosaApertaAcquirente' in 18aste
- 39 Cliccato 'profilo' in bottom navigation in 18aste
- 40 Cliccato 'acquisti' in 27profilo
- 41 Cliccato 'CardSilenziosaApertaAcquirente' in 31acquisti
- 42 Cliccato 'CardRibassoChiusaAcquirente' in 31acquisti
- 43 Cliccato 'indietro' in 31acquisti
- 44 Cliccato 'negozi' in 27profilo
- 45 Cliccato 'indietro' in 32negozi
- 46 Cliccato 'seguite' in 27profilo
- 47 Cliccato 'CardRibassoChiusaAcquirente' in 33seguite
- 48 Cliccato 'CardSilenziosaApertaAcquirente' in 33seguite
- 49 Cliccato 'indietro' in 33seguite
- 50 Cliccato 'home' in bottom navigation in 27profilo
- 51 Cliccato 'vendi' in 15home
- 52 Cliccato 'creaas' in bottom navigation in 34home
- 53 Cliccato 'continua' in 39crea
- 54 Cliccato 'setupCardListenerSilenziosaApertaVenditore' in 34home
- 55 Cliccato 'indietro' in 41dettaglio
- 56 Cliccato 'setupCardListenerSilenziosaApertaVenditore' in 34home
- 57 Cliccato 'vedi tutte' in bottom navigation in 34home
- 58 Cliccato 'profilo' in bottom navigation in 35aste
- 59 Cliccato 'negozi' in 27profilo
- 60 Cliccato 'CardSilenziosaApertaVenditore' in 32negozi
- 61 Cliccato 'indietro' in 32negozi



Considerazioni finali

In conclusione, desideriamo sottolineare l'importanza e il valore del progetto su cui abbiamo lavorato. Questo progetto ha rappresentato un'opportunità significativa per applicare e approfondire le nostre competenze tecniche e di migliorare ancora di più le nostre capacità di co-working.

Nel corso dello sviluppo di questo progetto di ingegneria del software, abbiamo affrontato una serie di sfide tecniche e concettuali. Tra queste, una delle più significative è stata l'implementazione di interfacce utente moderne e intuitive, progettate per soddisfare le esigenze di un'ampia gamma di utenti. Questo ha richiesto una comprensione approfondita dei principi di design dell'interfaccia utente e una capacità di tradurre queste conoscenze in un'implementazione efficace.

Abbiamo anche avuto l'opportunità di sviluppare una serie di funzionalità che sono comuni nelle applicazioni che utilizziamo quotidianamente. Queste includono l'invio di notifiche e email, la funzione di ricerca di aste all'interno del sistema, e lo sviluppo di un sistema di recupero password con invio di pin di recupero via mail. Quest'ultimo è stato particolarmente interessante, in quanto simula una sorta di autenticazione a due fattori, un elemento chiave della sicurezza informatica moderna.

Inoltre, siamo riusciti a padroneggiare una serie di strumenti e tecnologie nuovi. Questi includono il framework Springboot, che abbiamo utilizzato per lo sviluppo del back-end dell'applicazione, il sistema di cloud AWS EC2 e AWS S3, che abbiamo utilizzato per l'hosting e la gestione delle immagini, e Docker, che abbiamo utilizzato per la creazione e la gestione dei container dell'applicazione.

Il risultato finale è un'applicazione che non solo è tecnicamente solida, ma che è anche intuitiva e facile da usare per l'utente finale.

Siamo soddisfatti del lavoro svolto non solo per i risultati ottenuti, ma anche per il processo che ci ha portato a questi risultati, che ha richiesto un lavoro di squadra efficace.

Le competenze acquisite saranno di grande valore per i nostri futuri impegni professionali.

N86003605 , Biagio Scotto di Covella
N86003546 , Erasmo Prosciutto





Bibliografia

- [1] Italiani pazzi per le aste online, appassionati sempre più giovani. <https://www.tag24.it/281705-notizie-economia/italiani-pazzi-per-le-aste-online-appassionati-sempre-piu-giovani/>. Accessed: 2023-11-21.
- [2] Online auction market size, share trends to 2027. <https://www.technavio.com/report/online-auction-market-industry-analysis>. Accessed: 2023-11-21.
- [3] Il Sole 24 ORE. Il sole 24 ore website.
- [4] Research and Markets. Title of the article. *Global Online Auction Market (by Product Type, Region): Insights and Forecast with Potential Impact of COVID-19 (2022-2027)*, 2022.
- [5] style.1000lands.com. style.1000lands.com website.



Disclaimer

Tutte le informazioni fornite riflettono lo stato del progetto DietiDeals al momento della stesura e possono essere soggette a modifiche.

Copyright Notice

© 2024

