



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DIPARTIMENTO DI INFORMATICA - SCIENZA E INGEGNERIA - DISI

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

APPLICAZIONE PAWWY: RIPROGETTAZIONE ED ESTENSIONE

Relatore

Prof. Marco Patella

Presentata da

Clarissa Bovo

Sessione Marzo 2024

Anno Accademico 2022/2023

*“All’ingegnere compete rivestire di vita,
conforto e speranza lo scheletro della scienza.”*

Herbert Hoover

Abstract

Lo scopo di questa tesi è quello di riassumere il lavoro svolto di riprogettazione del documento di progetto della piattaforma Pawwy ed estensione dell'implementazione del prototipo.

In questo documento è possibile apprendere l'idea e le finalità di Pawwy, un'applicazione web sviluppata per incentivare l'adozione di animali che vivono nei rifugi e facilitare il ritrovamento di animali dispersi.

Inoltre viene effettuata una panoramica sul processo di sviluppo dall'applicazione, seguita dal confronto tra la sua progettazione iniziale e la riprogettazione avvenuta, la quale aveva come obiettivo il miglioramento del contenuto del documento e la correzione di errori e imprecisioni.

Successivamente vengono presentate le tecnologie utilizzate, le scelte implementative e mostrate delle schermate estrapolate dal prototipo aggiornato. Infine vengono riportate le conclusioni alle quali si è arrivati al termine del lavoro di tesi.

Indice

Abstract	i
1 Dominio applicativo di Pawwy	1
1.1 Contesto Applicativo	1
1.2 Requisiti di Pawwy	2
1.2.1 Requisiti generali	2
1.2.2 Requisiti dell'area adozioni	3
1.2.3 Requisiti dell'area smarrimenti	4
1.3 Scelta di titolo e slogan	5
2 Architettura di Pawwy	7
2.1 Analisi dei requisiti	8
2.1.1 Analisi del Dominio: Vocabolario	8
2.1.2 Casi d'Uso e Scenari	9
2.1.3 Analisi del Rischio	10
2.2 Analisi del problema	11
2.2.1 Analisi Documento dei Requisiti	11
2.2.2 Analisi dei Ruoli e delle Responsabilità	13
2.2.3 Scomposizione del problema	13
2.2.4 Creazione Modello del Dominio	14
2.2.5 Architettura Logica	16
2.2.6 Definizione Piano di Lavoro e Piano di Collaudo	21
2.3 Progettazione	21
2.3.1 Progettazione Architetturale	21

2.3.2	Progettazione di Dettaglio	23
2.3.3	Progettazione della Persistenza	25
2.3.4	Progettazione del Collaudo	26
2.3.5	Progettazione per il Deployment	26
3	Riprogettazione architetturale	27
3.1	Analisi dei requisiti	27
3.1.1	Casi d'Uso e Scenari	27
3.2	Analisi del problema	33
3.2.1	Creazione Modello del Dominio	34
3.2.2	Architettura Logica	35
3.3	Progettazione	39
3.3.1	Progettazione di Dettaglio	39
3.3.2	Progettazione della Persistenza	40
4	Scelte tecnologiche e implementazione	41
4.1	Scelte tecnologiche	41
4.2	Implementazione backend	42
4.3	Implementazione frontend	43
Conclusioni		47
Bibliografia		49
Ringraziamenti		51

Capitolo 1

Dominio applicativo di Pawwy

In questo capitolo viene presentato il contesto della piattaforma Pawwy, fornendo una panoramica delle sue funzionalità e introducendo requisiti, vincoli e concetti che ne stanno alla base.

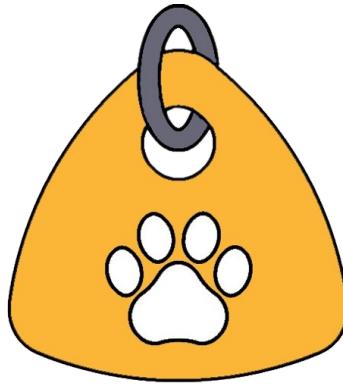


Figura 1.1: logo della piattaforma Pawwy

1.1 Contesto Applicativo

L'applicazione web Pawwy ha lo scopo di incentivare l'adozione di animali che vivono nei rifugi e facilitare il ritrovamento di animali dispersi. Il sistema dunque è suddiviso in due aree, una riguarda l'adozione, dove l'u-

tente è in grado di trovare il suo amico a quattro zampe ideale, grazie ad un breve questionario le cui risposte verranno elaborate da un algoritmo che ricerca l'animale che più gli si addice. Se l'utente è registrato può salvare sul proprio profilo il risultato del test effettuato, inoltre sia utenti registrati che non hanno la possibilità di visualizzare l'intera lista di animali adottabili presenti nelle varie strutture.

La seconda area è dedicata agli animali persi, il software è di supporto al ritrovamento di questi, attraverso le segnalazioni degli utenti, i quali possono avvertire di aver avvistato un animale apparentemente solo o segnalare di aver disperso il proprio.

1.2 Requisiti di Pawwy

Di seguito è possibile vedere la lista dei requisiti della piattaforma, presenti nel documento originale [1], permettendo così al lettore di farsi un'idea più precisa delle finalità di Pawwy.

1.2.1 Requisiti generali

1. Quando l'utente accede al sistema deve decidere se entrare nella sezione relativa alle adozioni o quella relativa agli smarrimenti
2. Un animale può essere presente nella sezione delle adozioni o in quella degli smarrimenti in maniera del tutto mutuamente esclusiva
3. Il sistema non prevede l'obbligo per l'utente di crearsi un account personale
4. L'eventuale registrazione dell'utente prevede l'inserimento di nome e cognome, uno username (email), una password almeno di 8 caratteri con lettere minuscole e maiuscole, numeri e almeno un carattere speciale, l'accettazione della privacy ed un numero di telefono facoltativo

5. Per gli utenti registrati, per accedere è necessario inserire ogni volta il proprio username e la propria password

1.2.2 Requisiti dell'area adozioni

6. Nel sistema dev'essere presente un elenco sempre aggiornato, contenente gli animali adottabili
7. Nella sezione dedicata alle adozioni un utente ha due opzioni, visionare la lista completa degli animali adottabili oppure compilare un form per scoprire qual è l'animale che più si adatta a lui e al suo stile di vita
8. Un utente può compilare il form più volte
9. Il form non vincola l'utente a procedere con l'adozione, ma questi è libero di compilarlo per pura curiosità
10. Il form dovrà mostrare una serie di domande a campo obbligatorio riguardo al tipo di animale che si vuole adottare, l'abitazione dell'utente (zona cittadina, casa singola o appartamento, presenza o meno di un giardino), la presenza in casa di bambini piccoli o anziani, preferenze in merito all'età dell'animale o alla disponibilità di adottare animali con problemi fisici
11. Il sistema riserva la possibilità per chi possiede un account personale di tener traccia delle schede degli animali preferiti
12. Il sistema non si occuperà di gestire le modalità con cui i singoli utenti si metteranno in contatto con le associazioni.
13. Un utente viene automaticamente associato ad un animale grazie alle risposte del suo test tramite un algoritmo
14. L'algoritmo restituisce una lista con n corrispondenze ordinate per affinità

15. Assieme alla descrizione dell'animale e alle sue caratteristiche ci devono essere anche i recapiti dell'associazione

1.2.3 Requisiti dell'area smarrimenti

16. Per effettuare una segnalazione è previsto che l'utente abbia un account personale e fornisca i propri recapiti
17. Il sistema è usufruibile da soli utenti privati e non prevede di allertare le autorità
18. Nella sezione dedicata alle segnalazioni un utente può scegliere se riportare una segnalazione (perdita o avvistamento) oppure consultare la lista delle segnalazioni attive
19. Prima di poter inserire una nuova segnalazione di avvistamento, l'algoritmo dovrà fare una ricerca sui dati inseriti nell'elenco delle perdite e trovare dei possibili riscontri e nel caso non ci siano allora l'utente può procedere all'inserimento della nuova segnalazione e viceversa
20. La segnalazione degli animali deve contenere i seguenti campi
 - Obbligatori: tipologia di segnalazione (ricerca o avvistamento, compilato automaticamente), tipo di animale, grandezza, colore, ha il chip (solo per ricerca), data e ora della perdita o dell'avvistamento, luogo (provincia, paese, via), coordinate (possibilità di compilazione automatica), recapiti dell'utente (non obbligatoriamente devono corrispondere a quelli inseriti alla creazione dell'account)
 - Facoltativi: foto, razza, note
21. Al ritrovamento di un animale, le segnalazioni riguardanti quest'ultimo devono essere chiuse

22. Possibilità per gli utenti che possiedono un account di visionare quelle che sono le segnalazioni che hanno effettuato, di gestirle e di ricevere una notifica nel caso in cui la sua segnalazione abbia fatto match con altre

1.3 Scelta di titolo e slogan

Il titolo scelto per la piattaforma è Pawwy, deriva dalla parola inglese “paw” ovvero zampa, un riferimento agli animali che si intende aiutare con questa applicazione web. Lo slogan è “Match your shaggy friend” che tradotto dall’inglese può significare sia trovare il tuo amico peloso, sia di abbinarti con l’amico a quattro zampe più adatto a te.

Capitolo 2

Architettura di Pawwy

Pawwy è stato progettato utilizzando UML come linguaggio di modellazione e seguendo un modello di processo ispirato a RUP (Rational Unified Process), che prevede una suddivisione del lavoro in quattro macro-fasi:

1. ANALISI dei REQUISITI

il cui scopo è quello di identificare i requisiti, gli attori ed i casi d'uso dell'applicazione

2. ANALISI del PROBLEMA

fornisce una descrizione generale della struttura dell'applicazione specificando quali debbano essere le entità deducibili dai requisiti e i loro comportamenti attesi

3. PROGETTAZIONE

fase durante la quale vengono proposte le soluzioni da adottare per ottenere i comportamenti individuati in analisi e vengono spiegati i motivi su cui si basano le decisioni prese

4. IMPLEMENTAZIONE

sviluppo di un prototipo dell'applicazione per ottenere un riscontro delle scelte progettuali fatte

In questo capitolo verrà illustrato un estratto del documento di progetto [1], frutto di tale sviluppo.

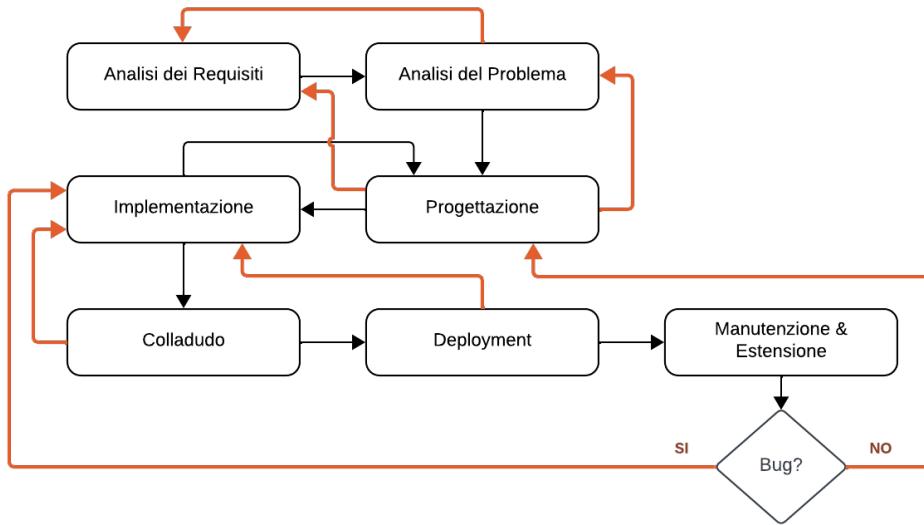


Figura 2.1: workflow del modello di progetto utilizzato [2]

2.1 Analisi dei requisiti

2.1.1 Analisi del Dominio: Vocabolario

Dopo la raccolta dei requisiti e la compilazione della tabella di questi ultimi (includendo funzionali e non funzionali), analizzando i sostantivi, è possibile formalizzare la conoscenza sul dominio applicativo, quindi costruzione di un primo modello del dominio, stilando il vocabolario del progetto, che è una lista dei termini usati nella specifica dei requisiti a cui viene data una definizione precisa [2].

Voce	Definizione	Sinonimi
Elenco	Archivio dei dati relativi agli animali adottabili o smarriti che l'utente può visionare liberamente	Lista
Smarrimento	Situazione in cui un animale viene perso e non ritorna a casa	Perdita
Utente	Persona che può accedere alla piattaforma e fare una ricerca per un'adozione o segnalare uno smarrimento	

Figura 2.2: esempio di vocabolario [1]

2.1.2 Casi d'Uso e Scenari

Invece, analizzando i verbi, è possibile individuare l'insieme delle azioni che il sistema dovrà compiere e quindi compilare il modello dei casi d'uso. I casi d'uso e i relativi scenari permettono di formalizzare i requisiti funzionali, di comprendere meglio il funzionamento del sistema e di comunicare meglio con il cliente [2].

Nella piattaforma Pawwy è stato individuato un solo attore, l'utente, che può essere autenticato all'interno della piattaforma, ma per alcune funzionalità del sistema non è necessario, ci sono tredici casi d'uso e di conseguenza tredici scenari (descrizione dettagliata di ciascun caso d'uso).

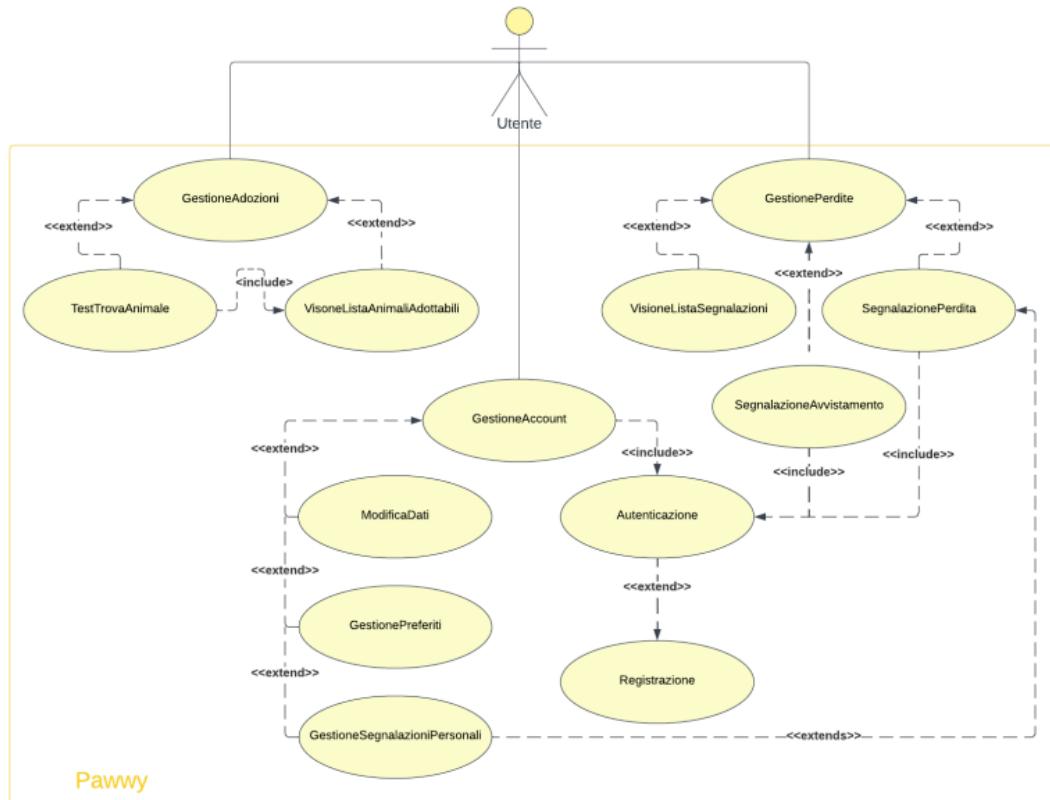


Figura 2.3: casi d'uso [1]

Titolo	SegnalazionePerdita
Descrizione	Scrivere una segnalazione per la perdita di un animale
Attori	Utente
Relazioni	GestionePerdite, CancellazionePerdita, Autenticazione
Precondizioni	<ol style="list-style-type: none"> 1. L'utente deve essere autenticato 2. L'utente ha scelto di visionare la lista delle segnalazioni dalla pagina di gestione perdite
PostCondizioni	L'utente avrà aggiunto una nuova segnalazione
Scenario Principale	<ol style="list-style-type: none"> 1. Presentazione all'utente di una pagina contenente i campi da compilare per la segnalazione 2. L'utente compila i campi (obbligatori e facoltativi) 3. L'utente conferma i dati inseriti 4. Il sistema verifica se sono presenti riscontri con segnalazioni di avvistamenti 5. Se il sistema non trova riscontri, salva la segnalazione 6. Il sistema lo riporta alla schermata di gestione perdite
Scenari alternativi	<p>Scenario a: il sistema trova riscontri con avvistamenti</p> <ol style="list-style-type: none"> 1. L'utente può verificare le segnalazioni di avvistamento proposte 2. L'utente sceglie se postare comunque la segnalazione o tornare indietro 3. Il sistema lo riporta alla schermata di gestione perdite <p>Scenario b: l'utente non compila tutti i campi obbligatori</p> <ol style="list-style-type: none"> 1. Dopo che l'utente clicca il pulsante di conferma il sistema mostra un errore per segnalare i campi mancati 2. L'utente completa la segnalazione aggiungendo ciò che manca 3. Si ritorna al punto 3 dello scenario principale
Requisiti non funzionali	R2NF, R1NF, R3NF
Punti Aperti	

Figura 2.4: esempio di uno scenario [1]

2.1.3 Analisi del Rischio

L'analisi del rischio si occupa di valutare le possibili perdite che un attacco può causare ai beni di un sistema e bilanciare queste perdite con i costi richiesti per la protezione dei beni stessi [2]. Quindi si tratta di valutare i be-

ni e analizzare minacce e controlli. Dopo si compilano i Security Use Case e i Misuse Case. I misuse case si concentrano sulle interazioni tra l'applicazione e gli attaccanti che cercano di violarla, invece il compito dei security use case è di specificare i requisiti tramite i quali l'applicazione dovrebbe essere in grado di proteggersi dalle minacce.

Durante la progettazione di Pawwy i misuse case individuati sono: DoS, furto di credenziali, MITM, SQL injection e i security use case: disponibilità sistema, controllo accesso e garantire protezione. L'unico scenario che è cambiato di conseguenza è quello dell'Autenticazione [1].

2.2 Analisi del problema

2.2.1 Analisi Documento dei Requisiti

Il documento dei requisiti evidenzia le funzionalità e i servizi che dovranno essere sviluppati (ovvero i requisiti funzionali), i vincoli di cui si deve tenere in considerazione (ovvero i requisiti non funzionali), il “mondo esterno” con cui si deve interagire e i “rischi” legati a possibili attacchi alla protezione, integrità e privacy dei dati (ovvero i requisiti di sicurezza) [2]

Per la piattaforma Pawwy sono state dedotte le funzionalità che il sistema avrebbe dovuto avere e compilate le tabelle di informazioni e flusso per ciascuna funzionalità (di seguito un esempio).

Funzionalità	Tipo	Grado Complessità	Requisiti Collegati
GestioneAdozione	Interazione con l'esterno	Semplice	R3F, R2NF
TestTrovaAnimale	Interazione con l'esterno, Manipolazione dei dati, Memorizzazione dei dati	Complesso	R3NF, R4NF, R2NF, R9F, R10F, R11F, R12F, R13F, R14F, R15F, R16F

Figura 2.5: esempio di tabella delle funzionalità [1]

Informazione	Tipo	Livello di protezione/privacy	Input/output	Vincoli
Domanda test composto da:	Composta	Basso	Output	
Tipologia Animale	Semplice	Basso	Output	
Età dell'animale	Semplice	Basso	Output	
Possibilità animali con problemi fisici	Semplice	Basso	Output	
Possibilità presenza Bambini	Semplice	Basso	Output	
Possibilità presenza Anziani	Semplice	Basso	Output	
Tipologia Abitazione dell'utente	Semplice	Basso	Output	
Risposta test composto da:	Composta	Basso	Input	
Tipologia Animale	Semplice	Basso	Input	
Età dell'animale	Semplice	Basso	Input	
Disponibilità accoglienza animali con problemi fisici	Semplice	Basso	Input	
Presenza Bambini	Semplice	Basso	Input	
Presenza Anziani	Semplice	Basso	Input	
Tipologia Abitazione dell'utente composto da:	Composta	Basso	Input	
Zona	Semplice	Basso	Input	
Presenza giardino	Semplice	Basso	Input	
Descrizione (appartamento o casa singola)	Semplice	Basso	Input	

Figura 2.6: tabella informazioni/flusso di test trova animale [1]

Inoltre nel documento [1] si possono vedere la tabella dei vincoli, quella delle maschere, quella dei sistemi esterni.

2.2.2 Analisi dei Ruoli e delle Responsabilità

Per ogni Attore individuato nei Casi d’Uso bisogna specificare le responsabilità , le informazioni a cui può accedere, le maschere che può visualizzare, il suo livello di riservatezza, la numerosità attesa.

Nel caso della piattaforma Pawwy è presente un solo attore, “Utente”, nel documento [1] si possono vedere la tabella dei ruoli e l’unica tabella ruolo/informazioni.

2.2.3 Scomposizione del problema

Per ogni funzionalità marcata come “complessa” nella Tabella delle Funzionalità occorre valutare se sia possibile operare una scomposizione [2].

Durante la stesura del documento [1] sono emerse diverse funzionalità complesse e per ciascuna di esse è stata compilata la tabella delle sotto-funzionalità.

Funzionalità	Scomposizione
TestTrovaAnimale	SottomissioneTest VisioneListaOrdinataPerAffinità VisioneSingolaScheda AggiuntaSchedaAiPrefertiti
VisioneListaAnimaliAdottabili	VisioneListaCompletaAnimaliAdottabili VisioneSingolaScheda AggiuntaSchedaAiPrefertiti
ModificaDati	VisioneInformazioniPersonalI ModificaInformazioniPersonalI
GestionePreferiti	VisioneListaPreferiti VisioneSingolaScheda EliminazioneSchedaDaiPreferiti
GestioneSegnalazioniPersonalI	VisioneListaSegnalazioniPerdita VisioneListaSegnalazioniAvvistamento VisioneSingolaSegnalazione ModificaSegnalazione CancellazioneSegnalazione ChiusuraSegnalazione VisioneRiscontriPerSegnalazioniPerdita

Figura 2.7: tabella scomposizione funzionalità [1]

2.2.4 Creazione Modello del Dominio

Creare il Modello del Dominio significa: individuare oggetti e classi rilevanti per il problema che si sta analizzando, limitarsi esclusivamente a quelle classi che fanno parte del vocabolario del dominio del problema, trovare le relazioni tra le classi e per ogni classe individuare attributi e operazioni fondamentali [2].

Per Pawwy le classi principali del Modello del Dominio progettato sono le seguenti.

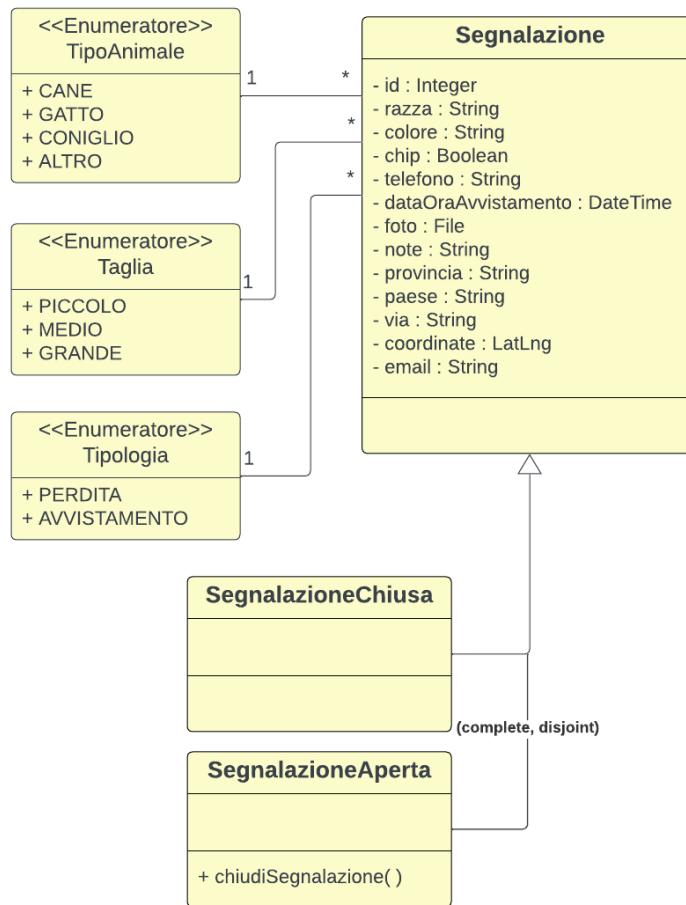


Figura 2.8: modello del dominio sezione gestione segnalazioni [1]

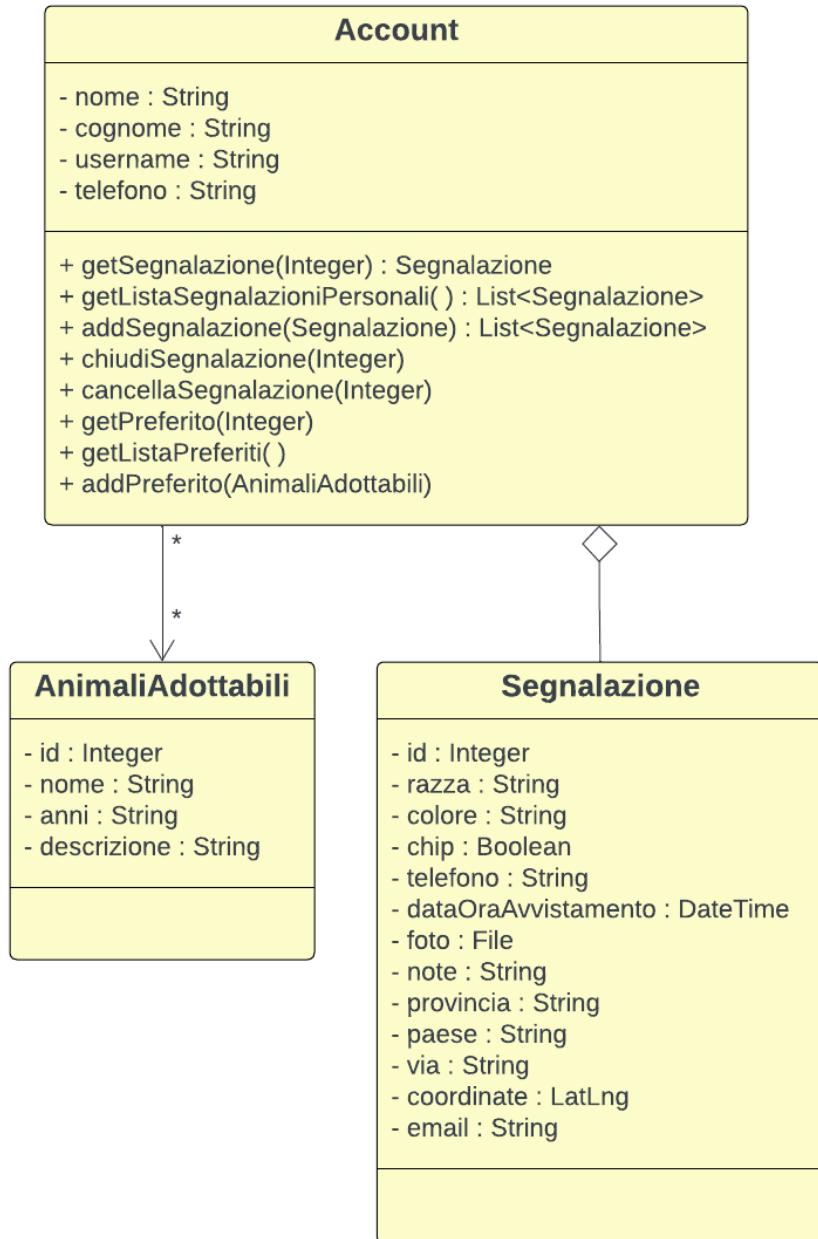


Figura 2.9: modello del dominio sezione gestione account [1]

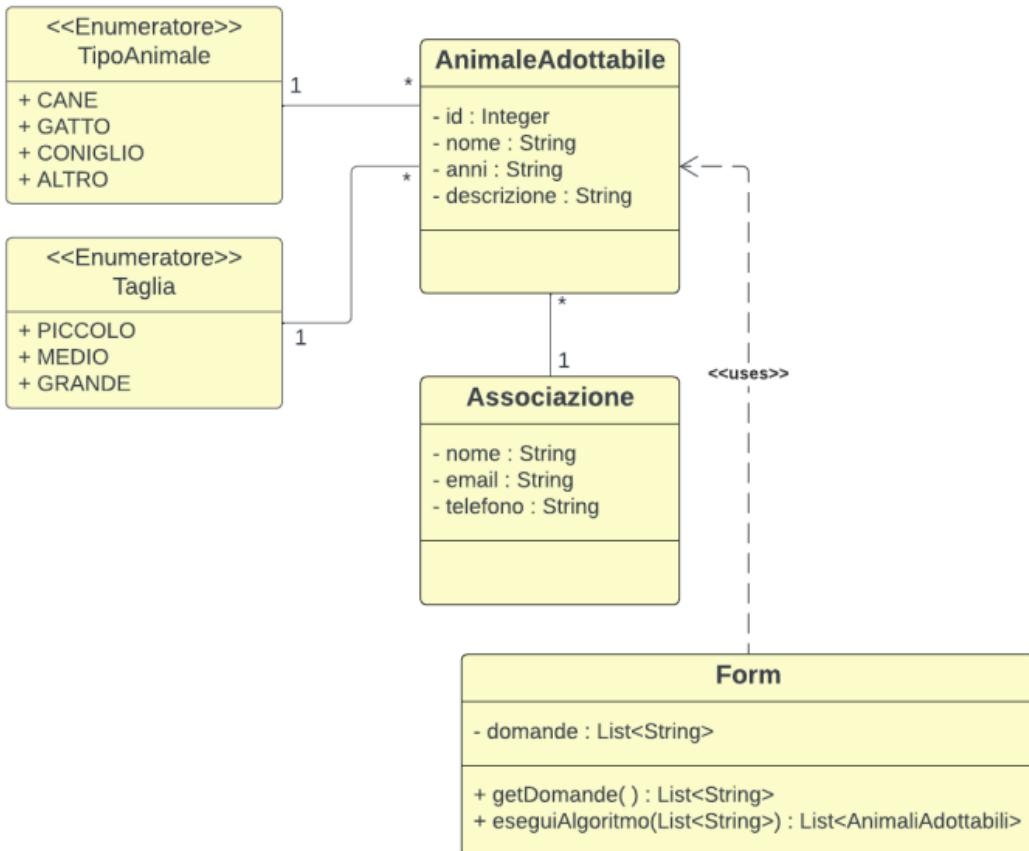


Figura 2.10: modello del dominio sezione gestione adozioni [1]

2.2.5 Architettura Logica

La parte strutturale dell'Architettura Logica è composta dal Diagramma dei Package, che fornisce una visione di alto livello dell'architettura e dal Diagramma delle Classi che fornisce una visione più dettagliata del contenuto dei singoli package.

Le interazioni dell'Architettura Logica sono composte da Diagrammi di Sequenza che evidenziano lo scambio di messaggi tra le classi della parte strutturale e gli attori, e l'ordine in cui i messaggi vengono scambiati.

Infine il comportamento dell'Architettura Logica si occupa di dettagliare funzionamenti complessi con i Diagrammi di Stato o delle Attività.

Nel progetto di Pawwy è stato usato il pattern architetturale Boundary-Control-Entity (BCE), che consiste nello partizionare sistematicamente gli use case in oggetti di tre categorie: informazione, presentazione e controllo, ottenendo una sequenza di livelli verticali (layer) che verranno mantenuti in fase di progettazione e implementazione.

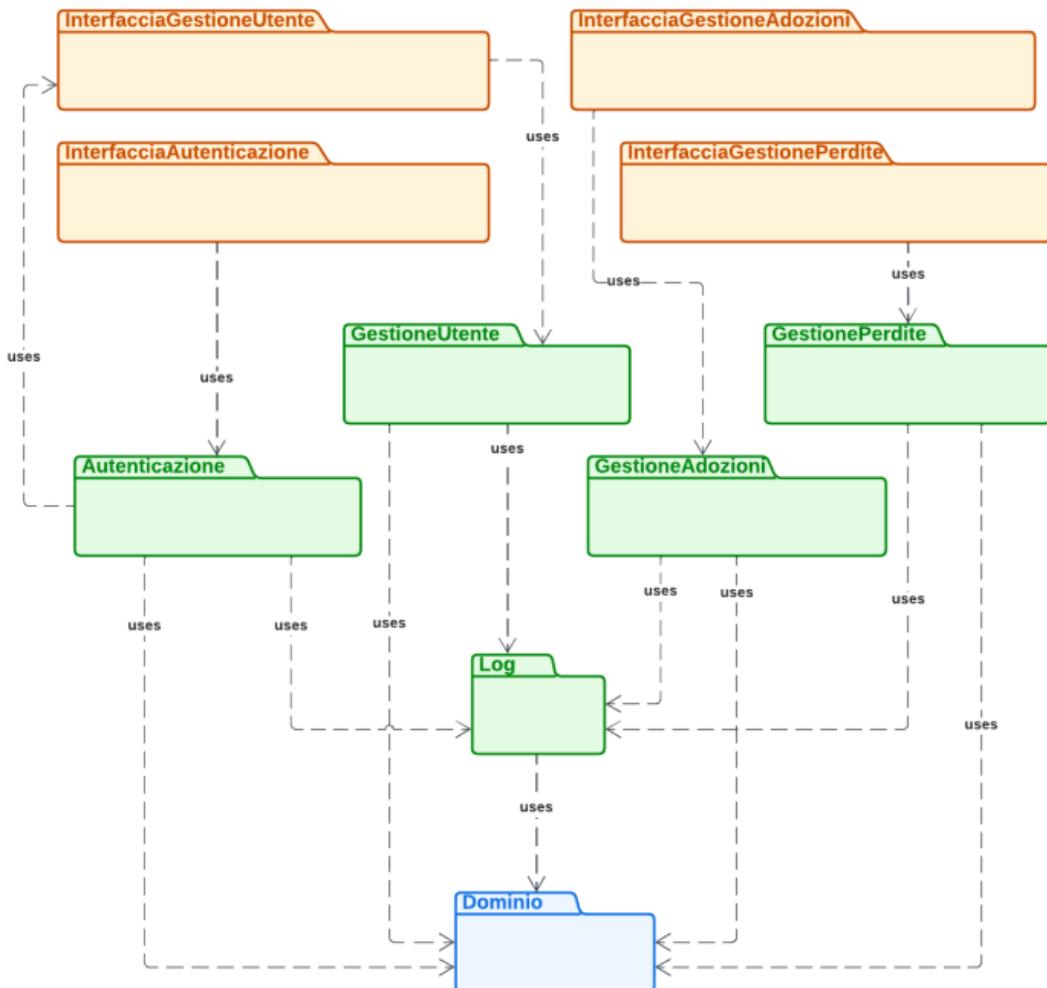


Figura 2.11: diagramma dei package [1]

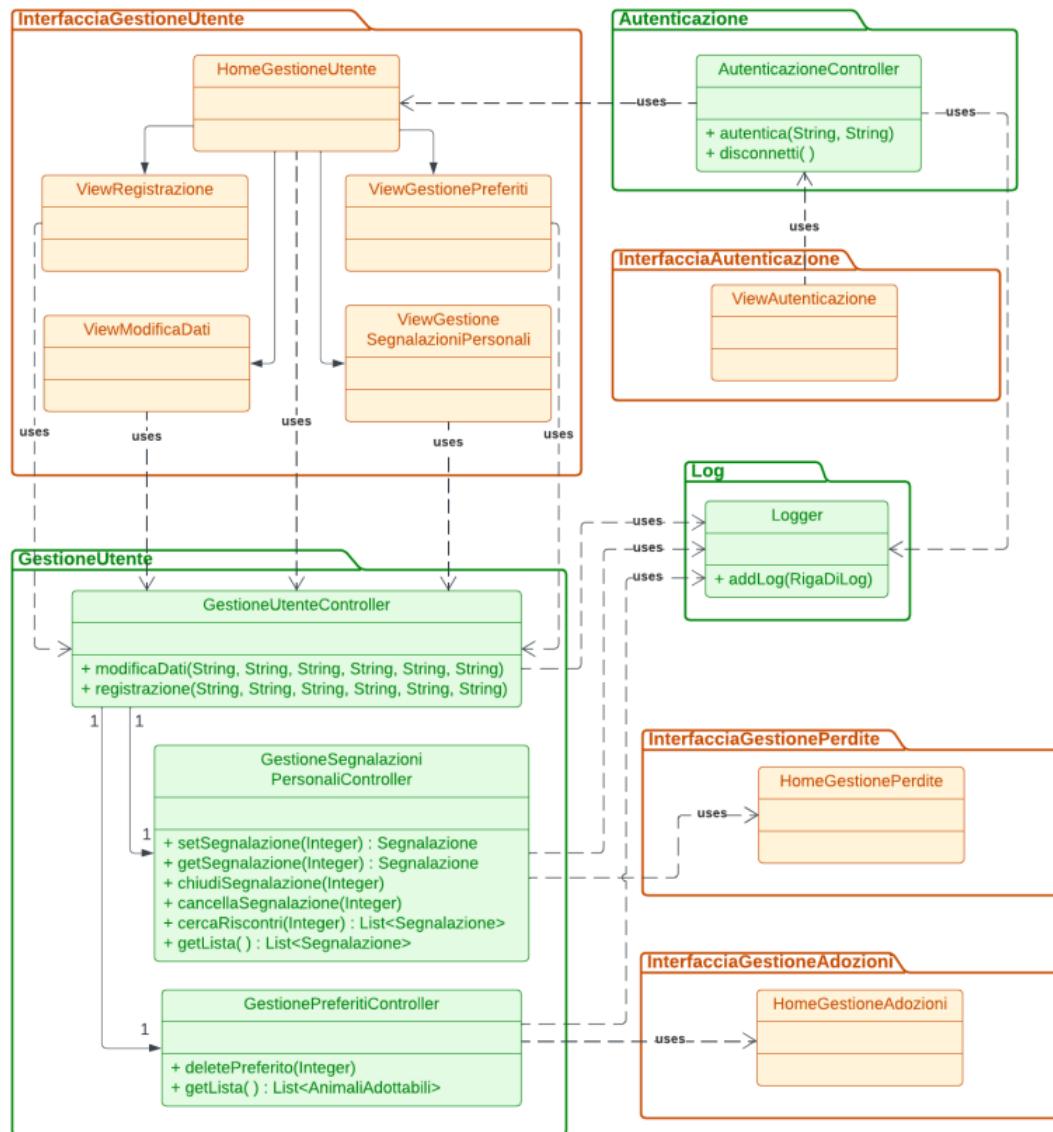


Figura 2.12: diagramma delle classi 1 [1]

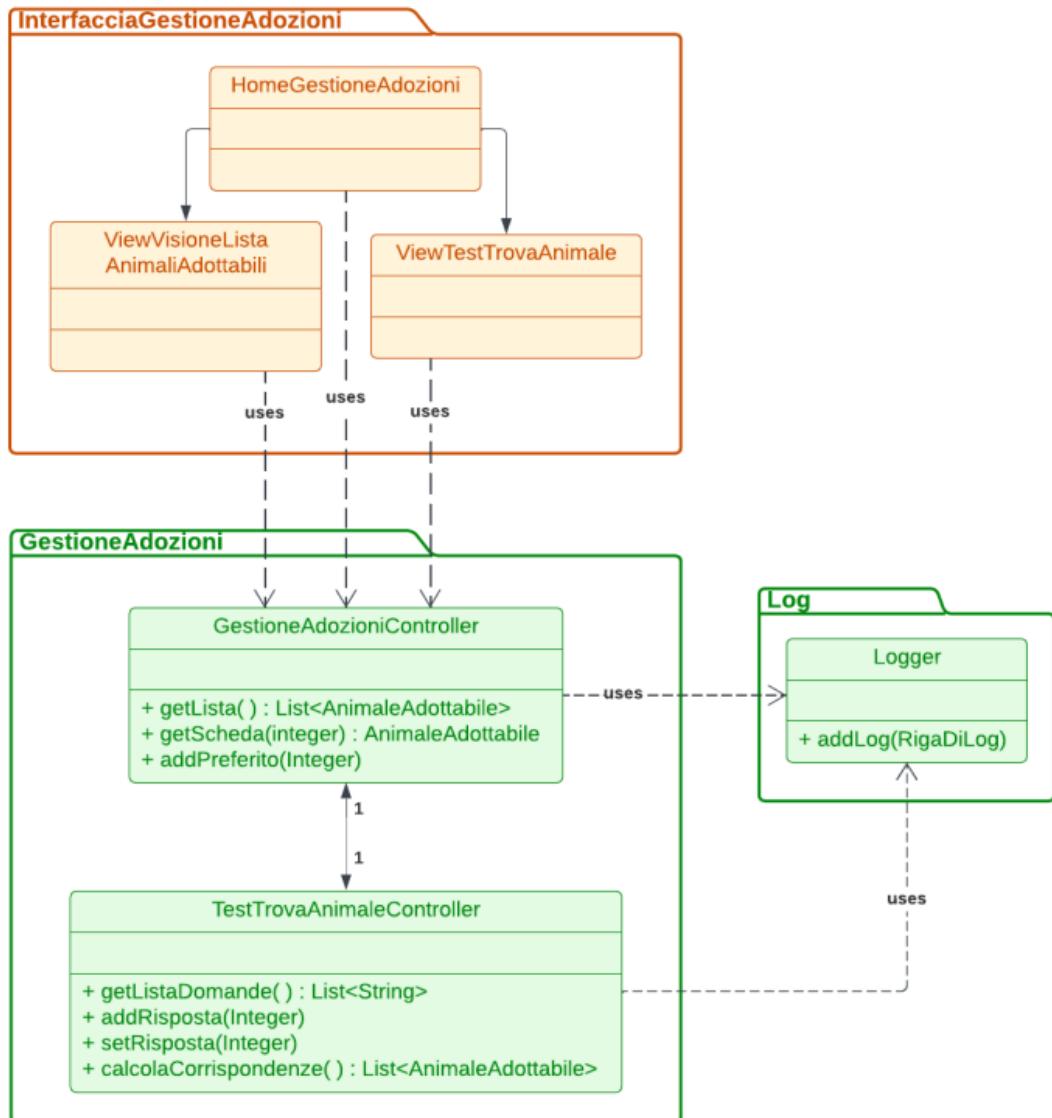


Figura 2.13: diagramma delle classi 2 [1]

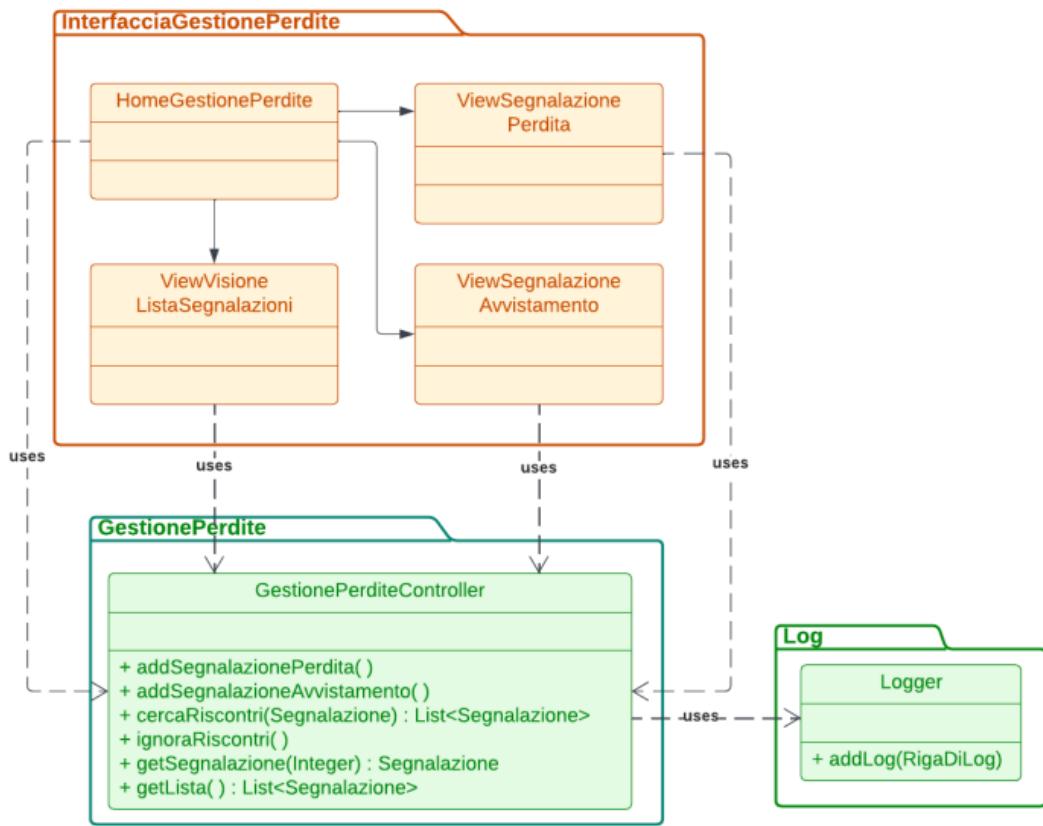


Figura 2.14: diagramma delle classi 3 [1]

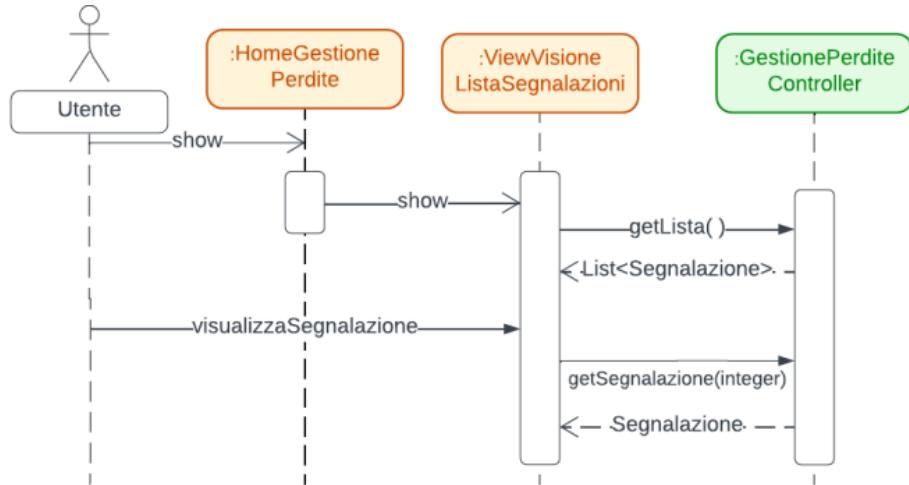


Figura 2.15: esempio di diagramma di sequenza [1]

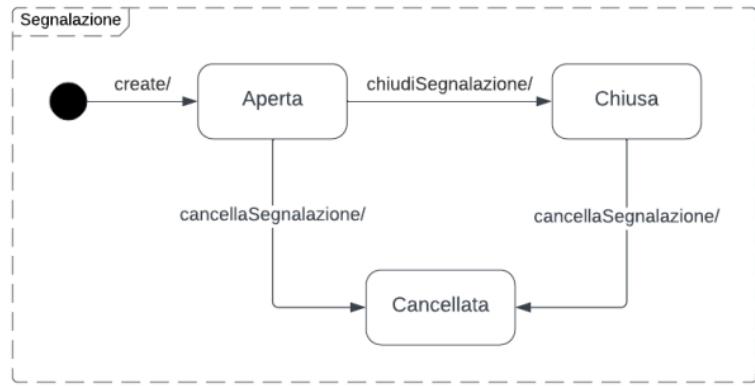


Figura 2.16: esempio di diagramma di stato [1]

2.2.6 Definizione Piano di Lavoro e Piano di Collaudo

Dopo la creazione dell’Architettura Logica è possibile iniziare a suddividere il lavoro, definendo un Piano di Lavoro.

Infine l’ultimo punto dell’Analisi del Problema è quello di definire un piano di collaudo, con lo scopo di cercare di precisare il comportamento atteso da parte di una entità prima ancora di iniziare il progetto e la realizzazione [2].

2.3 Progettazione

2.3.1 Progettazione Architetturale

L’obiettivo è definire l’Architettura del Sistema tenendo conto di tutti i vincoli e delle forze in gioco [2].

In fase di progettazione architetturale di Pawwy si è deciso di adottare il pattern MVC per suddividere il sistema in 3 parti: Model, Controller e View. Per garantire la sicurezza nella comunicazione client/server si utilizza il protocollo HTTPS basato su TLS. Inoltre è stato adottato il Pattern Broker per la gestione della sessione, ciò permette di applicare The Dependency Inversion Principle, disaccoppiando i client dal server.

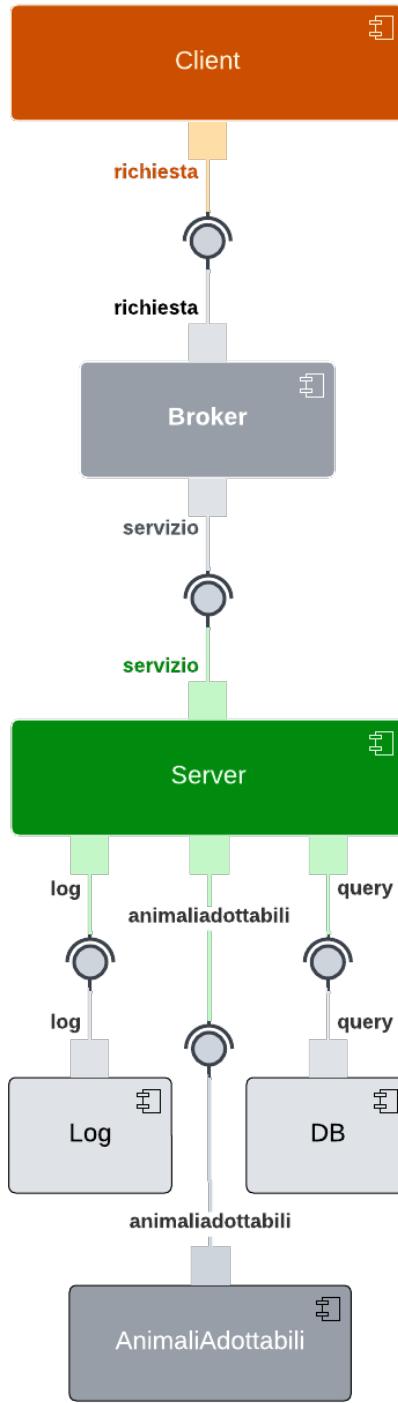


Figura 2.17: diagramma dei componenti dell'architettura del sistema [1]

2.3.2 Progettazione di Dettaglio

L'obiettivo è progettare nel dettaglio ogni aspetto del Sistema [2].

Rispetto all'Analisi del Problema nella progettazione dei diagrammi di dettaglio l'oggetto “AnimaleAdottabile” è stato eliminato, in quanto il Controller interagisce direttamente con le API del sistema esterno e non è necessario salvare i dati degli animali nel Dominio.

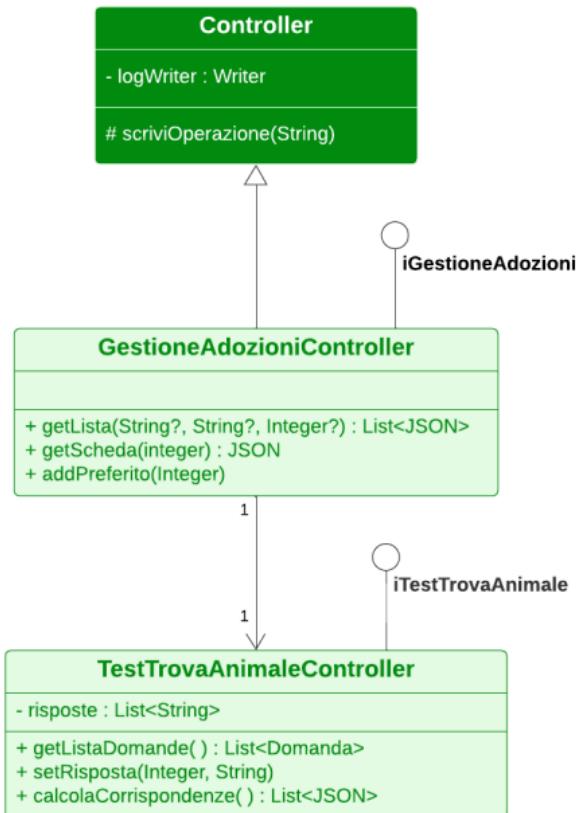


Figura 2.18: diagramma del controller della sezione adozioni [1]

Le interfacce (es. “iGestioneAdozioni” e “iTrovAnimale”) permettono di applicare The Dependency Inversion Principle.

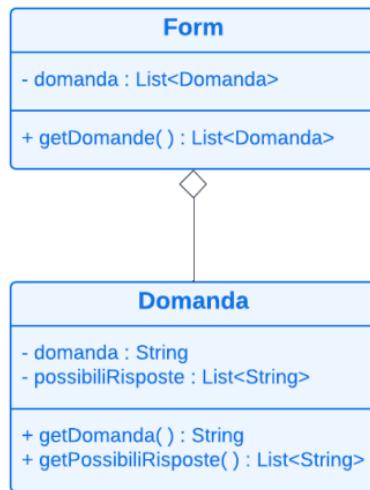


Figura 2.19: diagramma di dettaglio della sezione adozioni [1]

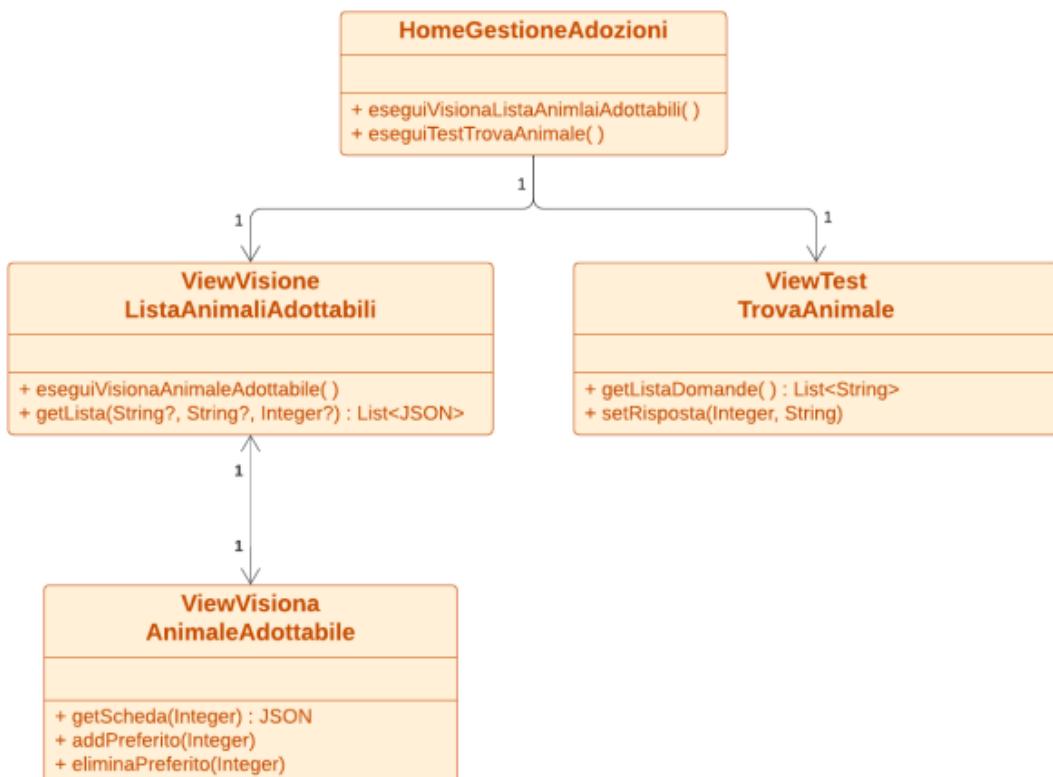


Figura 2.20: diagrammi delle home e delle view della sezione adozioni [1]

I Diagrammi di Sequenza presenti in questa fase sono una versione più specifica e incentrata maggiormente sul comportamento delle tecnologie di quelli presenti nella fase di Analisi del Problema.

2.3.3 Progettazione della Persistenza

L'obiettivo è progettare i meccanismi per la persistenza dei dati [2].

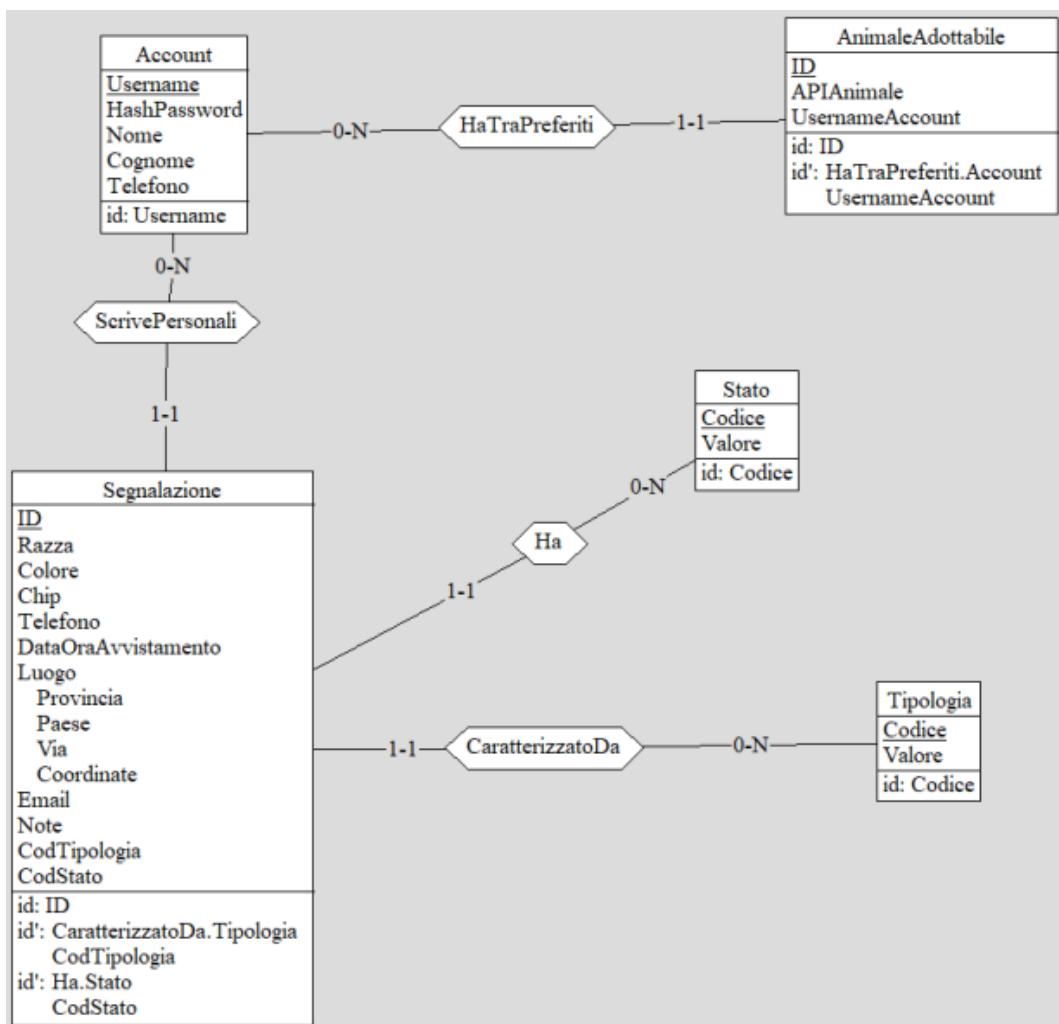


Figura 2.21: diagramma E/R della persistenza [1]

2.3.4 Progettazione del Collaudo

L'obiettivo è definire in modo chiaro e preciso come il sistema dovrà essere collaudato una volta terminata l'implementazione [2].

2.3.5 Progettazione per il Deployment

L'obiettivo è progettare il sistema in modo da rendere semplice il deployment sulle macchine e per garantire la sicurezza [2].

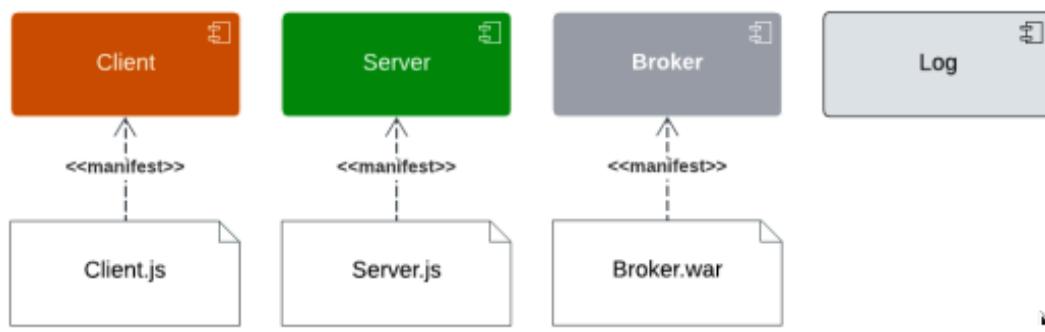


Figura 2.22: deployment del sistema [1]

Capitolo 3

Riprogettazione architetturale

In questo capitolo verranno illustrate le principali riprogettazioni e correzioni necessarie del documento originale [1] e le estensioni più rilevanti rispetto ad esso.

3.1 Analisi dei requisiti

Nell'analisi dei requisiti sono state fatte diverse migliorie rispetto al progetto iniziale di Pawwy [1], come la ridefinizione di alcune voci del vocabolario e il passaggio da requisiti funzionali a requisiti non funzionali dei seguenti:

R5NF	Mutua esclusione tra animali presenti nelle sezioni	Non Funzionale
R6NF	Test non vincolante all'adozione	Non Funzionale

Figura 3.1: requisiti modificati

3.1.1 Casi d'Uso e Scenari

È stata necessaria una riprogettazione degli scenari dei casi d'uso, in modo tale che fossero meno legati all'interfaccia grafica. In particolare, uno scenario deve descrivere l'interazione tra l'attore e il sistema e le elaborazioni necessarie per soddisfare la richiesta dell'attore.

Di seguito si possono osservare un paio d'esempi di nuovi scenari.

Scenario autenticazione prima

Nella descrizione dello scenario principale ci sono troppi riferimenti a oggetti grafici, legati all'interfaccia della piattaforma, negli scenari alternativi non è chiara la distinzione tra Scenario a e Scenario b.

Titolo	Autenticazione
Descrizione	Verificare quale utente possiede un account
Attori	Utente
Relazioni	GestioneAccount, SegnalazioniPerdita, SegnalazioniAvvistamenti
Precondizioni	L'utente deve essere correttamente registrato nella piattaforma
PostCondizioni	L'utente è autenticato
Scenario Principale	<ol style="list-style-type: none"> 1. Presentazione all'utente di una pagina dove inserire Username e Password 2. L'utente inserisce Username e Password dopodichè preme un bottone per validare i propri dati 3. Il sistema controlla che le credenziali di accesso siano corrette 4. Il sistema autentica l'utente e lo riporta alla schermata precedente alla richiesta di autenticazione
Scenari alternativi	<p>Scenario a: le credenziali inserite non corrispondono</p> <ol style="list-style-type: none"> 1. Il sistema mostra un messaggio all'utente in cui gli comunica che ha sbagliato a inserire username e password 2. Il sistema riporta l'utente al punto 1 dello scenario principale <p>Scenario b: l'utente non è ancora registrato</p> <ol style="list-style-type: none"> 1. Il sistema comunica all'utente che lo username inserito non corrisponde a nessuno di quelli già convalidati 2. Il sistema riporta l'utente il punto 1 di Registrazione
Requisiti non funzionali	R4NF, R2NF
Punti Aperti	

Figura 3.2: scenario del caso d'uso “Autenticazione” [1]

Scenario autenticazione dopo

Lo scenario principale è stato cambiato, ora la descrizione si concentra maggiormente nell'interazione tra richieste dell'attore "Utente" e le risposte del sistema.

Lo scenario alternativo a si verifica solamente quando la password è errata. Anche la descrizione dei due scenari alternativi è cambiata, eliminando i riferimenti grafici.

Titolo	Autenticazione
Descrizione	Verificare quale utente possiede un account
Attori	Utente
Relazioni	GestioneAccount, SegnalazioniPerdita, SegnalazioniAvvistamenti
Precondizioni	L'utente deve essere correttamente registrato nella piattaforma
PostCondizioni	L'utente è autenticato
Scenario Principale	<ol style="list-style-type: none"> 1. Il sistema dà la possibilità all'utente di inserire Username e Password 2. L'utente inserisce Username e Password e conferma 3. Il sistema controlla che le credenziali di accesso siano corrette 4. Il sistema autentica l'utente
Scenari alternativi	<p>Scenario a: la password è errata</p> <ol style="list-style-type: none"> 1. Il sistema comunica all'utente che ha sbagliato a inserire la password 2. Il sistema riporta l'utente al punto 1 dello scenario principale <p>Scenario b: l'utente non è ancora registrato</p> <ol style="list-style-type: none"> 1. Il sistema comunica all'utente che lo username inserito non corrisponde a nessuno di quelli già convalidati 2. Il sistema riporta l'utente il punto 1 di Registrazione
Requisiti non funzionali	R4NF, R2NF
Punti Aperti	

Figura 3.3: scenario del caso d'uso "Autenticazione"

Scenario visione lista animali adottabili prima

Come prima e come in tutti gli scenari, ci sono troppi riferimenti a oggetti grafici, legati all'interfaccia della piattaforma.

Non serve la relazione con il caso d'uso “TestTrovaAnimali”.

Nello scenario principale non è necessario il primo punto, essendo una ripetizione della precondizione e sempre in questa sezione viene dato per scontato che l'utente è autenticato (dato che può aggiungere una scheda ai preferiti), ma non c'è traccia di ciò nelle precondizioni.

Titolo	VisioneListaAnimaliAdottabili
Descrizione	Il sistema fornisce la lista di tutti gli animali che è possibile adottare
Attori	Utente
Relazioni	GestioneAdozioni, TestTrovaAnimali
Precondizioni	L'utente ha selezionata la sezione delle adozioni
PostCondizioni	Il sistema mostra una schermata con la lista di tutte le schede relative agli animali adottabili
Scenario Principale	<ol style="list-style-type: none"> 1. Nella schermata relativa alle adozioni l'utente preme il pulsante VAI ALLA LISTA 2. Il sistema restituisce la lista. Si possono applicare alla lista dei filtri (es. visualizza solo cani/gatti ecc.) 3. L'utente può selezionare ogni singola scheda 4. L'utente ha la possibilità di aggiungere una scheda tra i preferiti 5. Per ritornare dalla singola scheda alla lista completa, l'utente preme il pulsante TORNA INDIETRO
Scenari alternativi	Scenario a: l'utente non è autenticato <ol style="list-style-type: none"> 1. L'utente non può aggiungere una scheda ai preferiti 2. In caso clicchi sul bottone AGGIUNGI AI PREFERITI verrà riportato alla schermata di autenticazione
Requisiti non funzionali	R1NF, R4NF
Punti Aperti	

Figura 3.4: scenario del caso d'uso “VisioneListaAnimaliAdottabili” [1]

Scenario visione lista animali adottabili dopo

Non c'è più relazione tra “VisioneListaAnimaliAdottabili” e “TestTrovaAnimali”, sia nelle precondizioni che nelle postcondizioni sono stati sostituiti i riferimenti all'interfaccia.

Nello scenario principale non c'è più il primo punto e sono stati eliminati i riferimenti grafici dagli altri punti.

Tutto ciò che riguarda il salvataggio delle schede nei preferiti è stato eliminato, in sostituzione si vede una nuova relazione, “AggiungiAiPreferiti”, che verrà approfondita in seguito.

Titolo	VisioneListaAnimaliAdottabili
Descrizione	Il sistema fornisce la lista di tutti gli animali che è possibile adottare
Attori	Utente
Relazioni	GestioneAdozioni, AggiungiAiPreferiti
Precondizioni	L'utente ha scelto la sezione delle adozioni
PostCondizioni	Il sistema mostra la lista di tutte le schede relative agli animali adottabili
Scenario Principale	<ol style="list-style-type: none"> 1. Il sistema restituisce la lista degli animali adottabili. 2. L'utente può visualizzare per esteso ogni singola scheda 3. L'utente ha la possibilità di aggiungere una scheda tra i preferiti con AggiungiAiPreferiti 4. L'utente può ritornare dalla singola scheda alla lista completa
Scenari alternativi	
Requisiti non funzionali	R1NF, R4NF, R5NF
Punti Aperti	

Figura 3.5: scenario del caso d'uso “VisioneListaAnimaliAdottabili”

Caso d'uso aggiungi ai preferiti

Come si vede nello scenario di “VisoneListaAnimaliAdottabili” è stato aggiunto un nuovo caso d'uso, “AggiungiAiPreferiti”, che si occupa della possibilità da parte di un utente autenticato di aggiungere ai preferiti una scheda di un animale adottabile, estendendo “VisoneListaAnimaliAdottabili”, di seguito è possibile vedere la sua integrazione con gli altri casi d'uso e il suo comportamento con la tabella dello scenario.

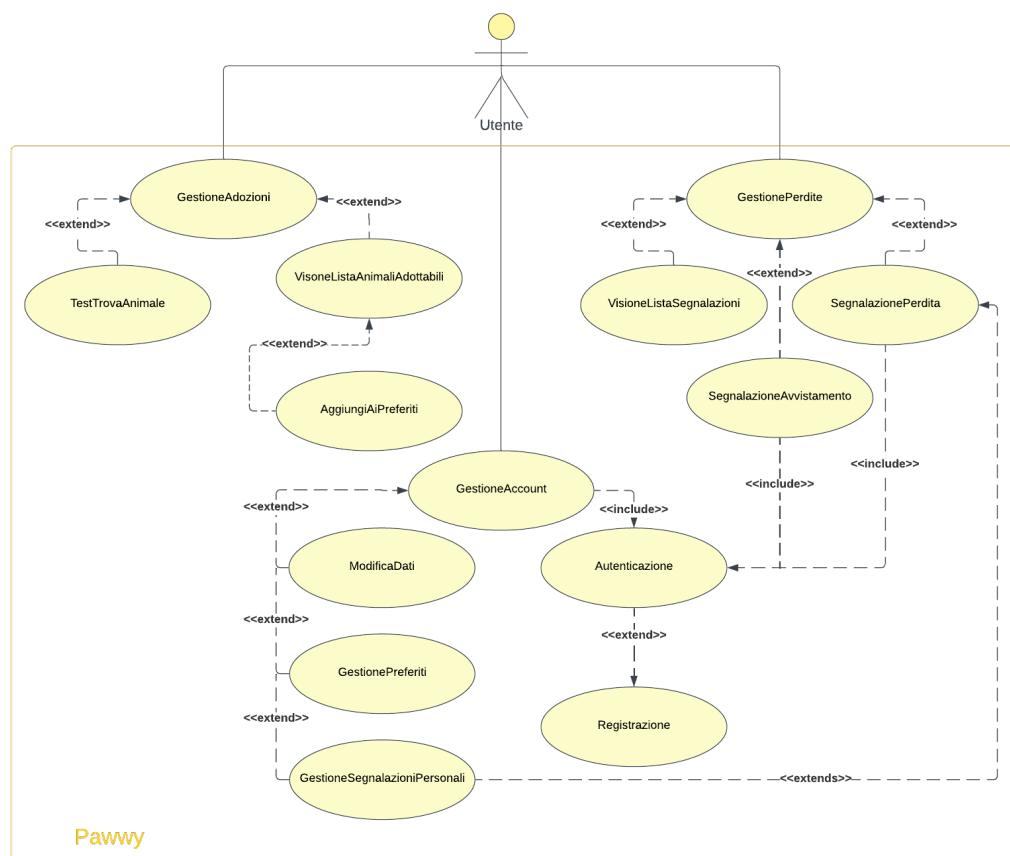


Figura 3.6: integrazione del caso d'uso “AggiungiAiPreferiti”

Titolo	AggiungiAiPreferiti
Descrizione	Il sistema offre all'utente la possibilità di aggiungere un animale adottabile ai preferiti
Attori	Utente
Relazioni	VisioneListaAnimaliAdottabili
Precondizioni	L'utente ha scelto una scheda tra quelle della lista di VisioneListaAnimaliAdottabili
PostCondizioni	L'utente ha aggiunto una scheda ai preferiti
Scenario Principale	<ol style="list-style-type: none"> 1. Il sistema mostra la scheda di un animale adottabile 2. L'utente sceglie di aggiungerla ai preferiti 3. Il sistema aggiunge la scheda ai preferiti di quell'utente
Scenari alternativi	<p>Scenario a: l'utente non è autenticato</p> <ol style="list-style-type: none"> 1. L'utente non può aggiungere una scheda ai preferiti 2. In caso provi ad aggiungere una scheda ai preferiti il sistema chiederà di autenticarsi
Requisiti non funzionali	R1NF, R3NF
Punti Aperti	

Figura 3.7: scenario del caso d'uso “AggiungiAiPreferiti”

3.2 Analisi del problema

Nell'analisi del problema è stata riprogettata parte delle funzionalità, come la ridefinizione del grado di complessità di alcune di esse, la modifica di alcune tabelle di informazioni/flusso e sono state migliorate le scomposizioni di alcune funzionalità complesse, troppo legate all'interfaccia grafica nella progettazione precedente.

Inoltre è stata stata modificata anche la tabella delle maschere, correggendo alcune View ed è cambiata la numerosità della tabella dei ruoli.

3.2.1 Creazione Modello del Dominio

La riprogettazione del Modello del Dominio ha avuto un impatto importante nella struttura della piattaforma.

Modello del Dominio Adozioni prima

Nel Modello del Dominio della sezione delle adozioni, la classe “Form” non serve, in quanto si possono gestire le domande del test e l'esecuzione dell'algoritmo per trovare un animale compatibile direttamente nel layer dei Controller.

Il precedente modello è stato mostrato alla Figura 2.10.

Modello del Dominio Adozioni dopo

Nel nuovo Modello del Dominio non troviamo più la classe “Form” .

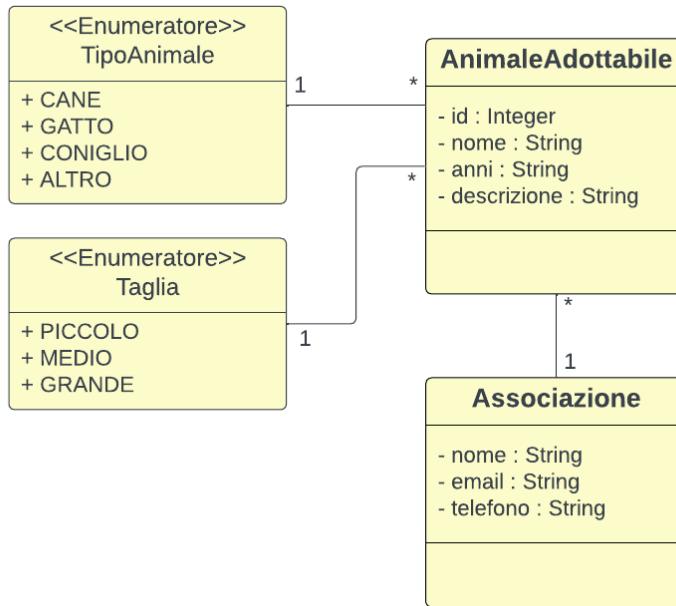


Figura 3.8: modello del domio della sezione adozioni attuale

3.2.2 Architettura Logica

Le modifiche sostanziali nella fase di Analisi del Problema sono state fatte nell'Architettura Logica, eccone degli esempi.

Diagramma dei Package

Nel Diagramma mancano le associazioni tra i Controller e le View ed è necessario il riferimento al sistema esterno.

Il precedente diagramma è stato mostrato alla Figura 2.11

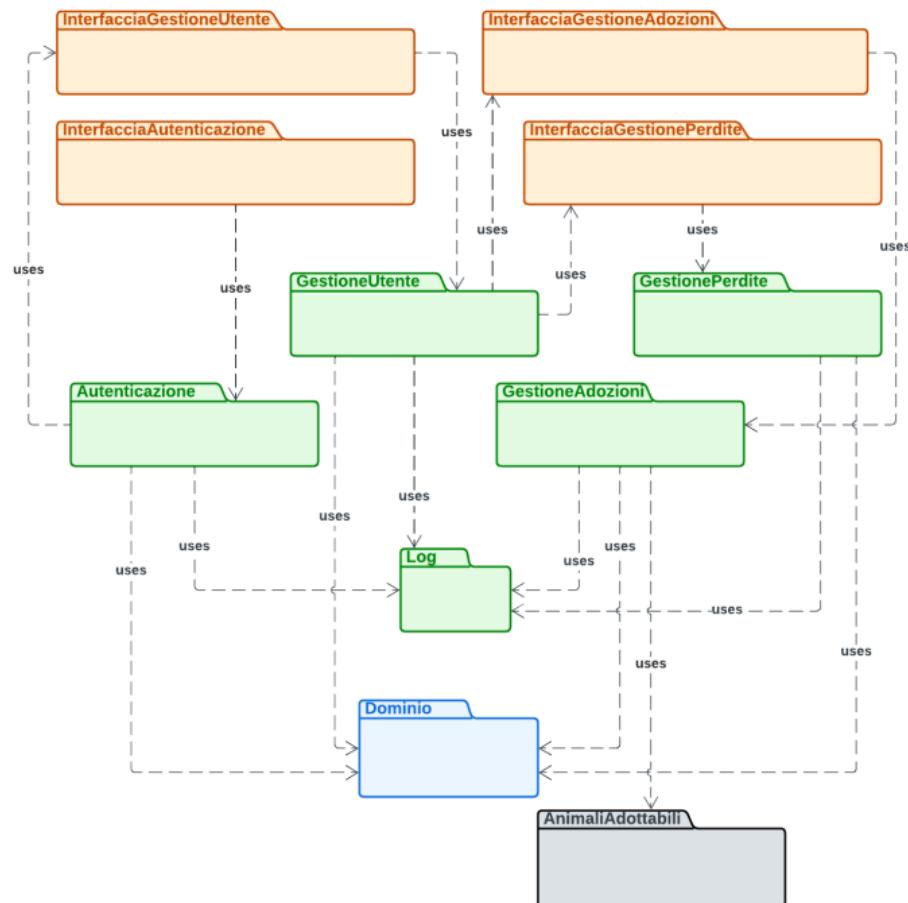


Figura 3.9: diagramma dei package attuale

Diagramma delle Classi

Nei Diagrammi delle Classi è stato modificato solo Logger che non ha più “RigaDiLog” come parametro del suo metodo addLog().

I precedenti diagrammi sono stati mostrati alle Figure 2.12, 2.13 e 2.14

Diagramma di Sequenza Registrazione

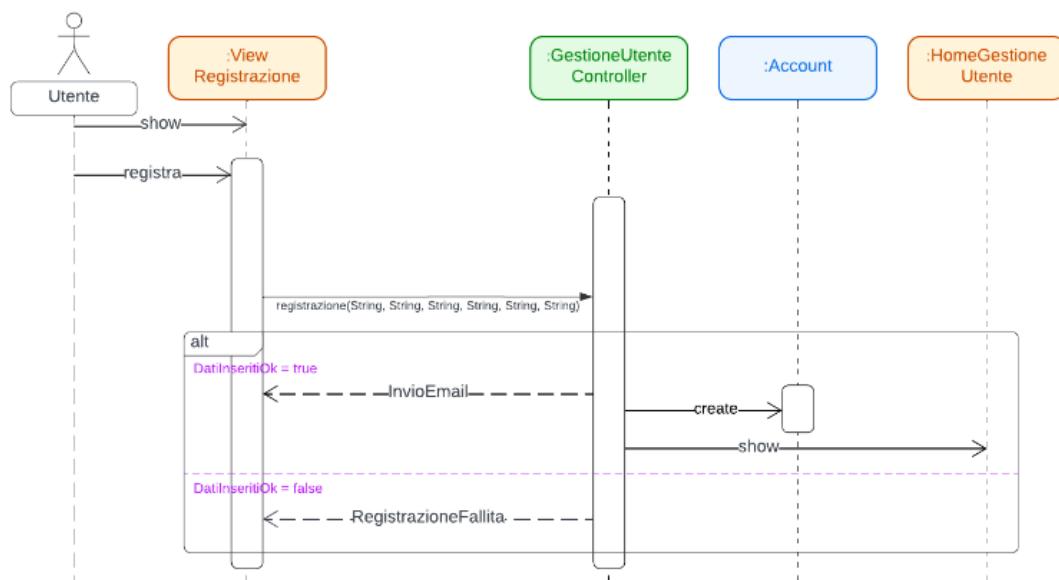


Figura 3.10: diagramma di sequenza della registrazione prima [1]

L’Utente, ovvero l’attore, non può fare la show di una View, inoltre non è molto chiara la dinamica della conferma tramite email.

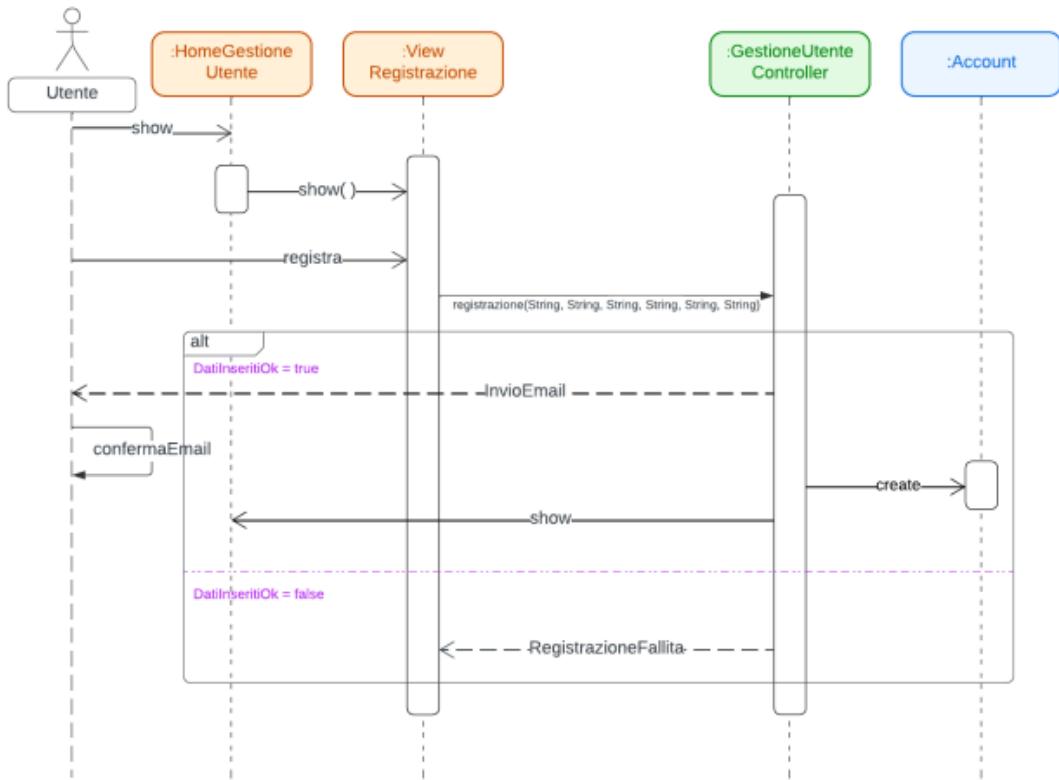


Figura 3.11: diagramma di sequenza della registrazione attuale

L’utente fa la show della Home di gestione utente, la quale a sua volta non essendo l’utente autenticato fa la show della View di registrazione, una volta completato l’inserimento dei dati e se quest’ultimi vengono approvati dal sistema, il Controller invia una mail all’utente il quale conferma la registrazione tramite l’email ricevuta, dopodichè il Controller crea un nuovo account e riporta l’utente alla Home.

Diagramma di Test Trova Animale

In “TestTrovaAnimale” è necessario eliminare la classe “Form”, non più presente nel Modello del Dominio, e aggiungere la Home “GestioneAdozioni” (un attore non può fare la show di una View)

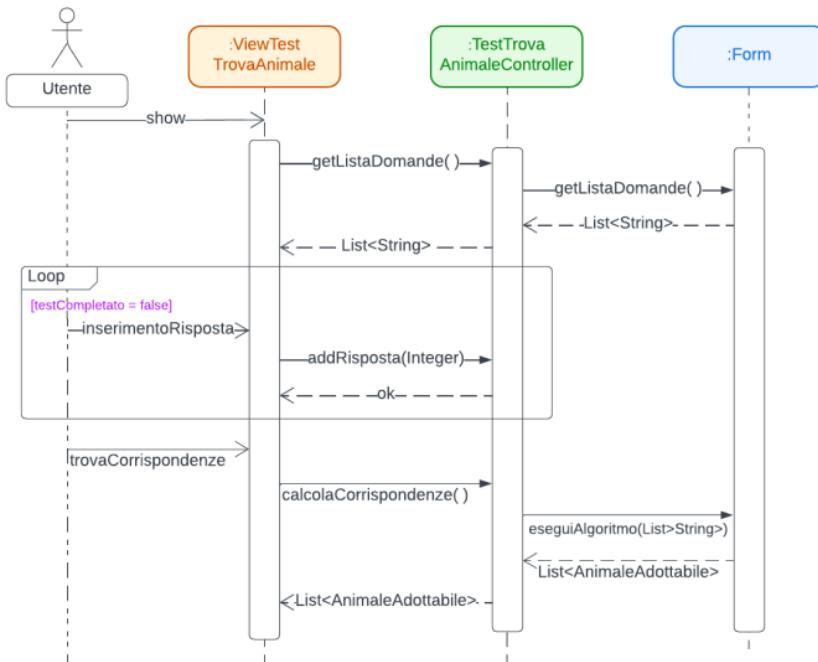


Figura 3.12: diagramma di sequenza del test trova animale prima [1]

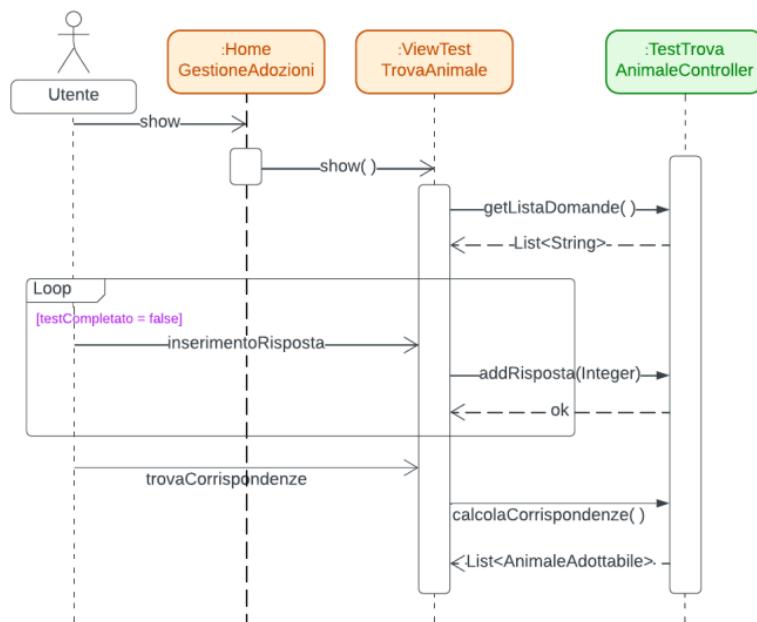


Figura 3.13: diagramma di sequenza del test trova animale attuale

3.3 Progettazione

Nella fase di Progettazione sono stati fatti cambiamenti nei Diagrammi di Dettaglio, di Sequenza e di Deployment e nella Progettazione della Persistenza. Di seguito è possibile vedere degli esempi rilevanti.

3.3.1 Progettazione di Dettaglio

Nella sezione delle adozioni, la classe “Form” e la classe “Domanda” non servono, in quanto come visto nella fase di riprogettazione ell’Analisi del Problema si possono gestire direttamente con dei Controller, di conseguenza è stato aggiunto un nuovo Controller.

Il precedente modello è stato mostrato alla Figura 3.14.

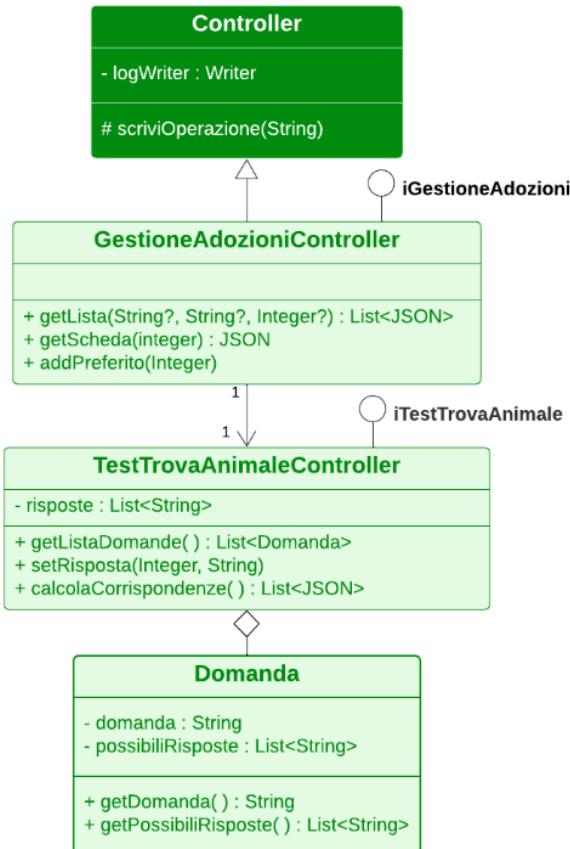


Figura 3.14: diagramma del controller della sezione adozioni [1]

3.3.2 Progettazione della Persistenza

Nella riprogettazione del Diagramma della Persistenza è stata eliminata la Tipologia della Segnalazione, in quanto si gestisce nella progettazione logica. Il precedente diagramma E/R è stato mostrato alla Figura 2.21.

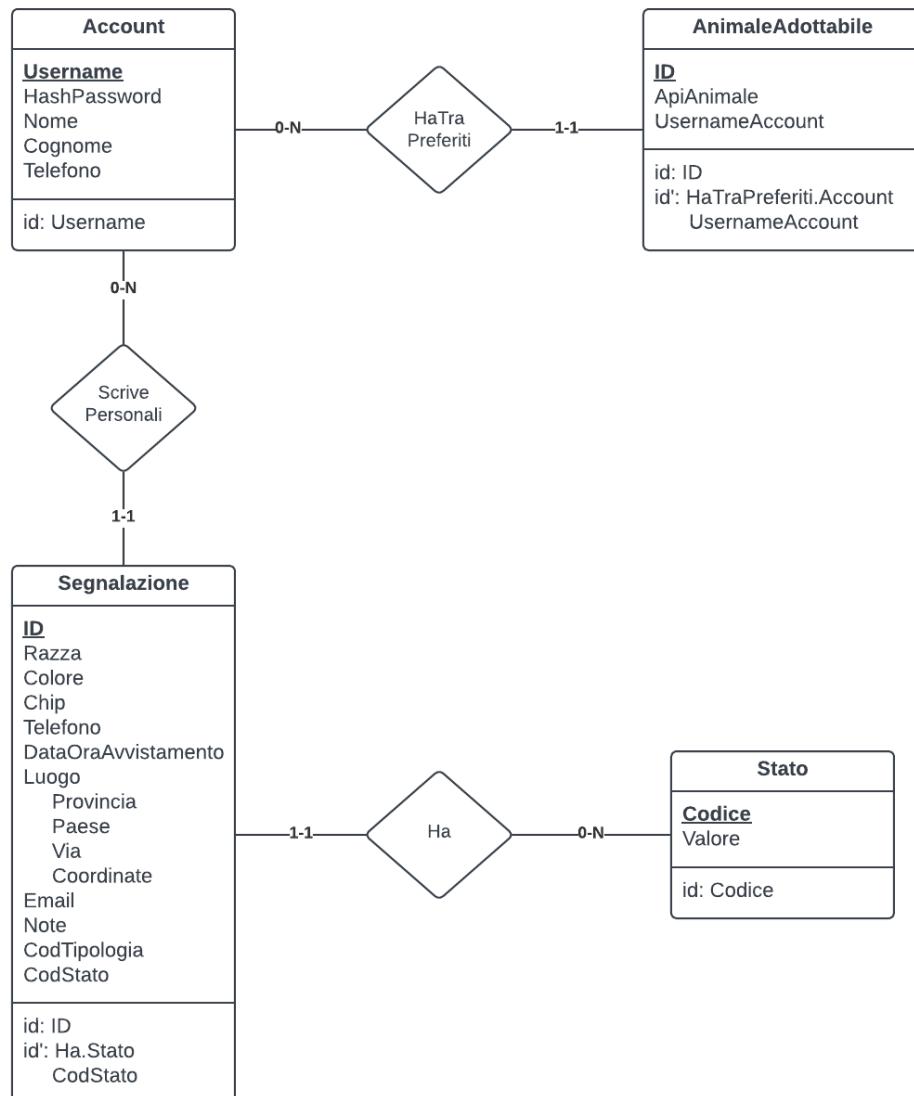


Figura 3.15: diagramma E/R della persistenza attuale

Capitolo 4

Scelte tecnologiche e implementazione

Il prototipo implementativo originale è stato espanso introducendo le funzionalità mancanti previste in fase di progettazione. Inoltre è stato migliorando il codice sorgente, rendendolo maggiormente leggibile grazie anche alla standardizzazione dei commenti.

4.1 Scelte tecnologiche

Per l'implementazione del Model è stato usato il framework Spring-boot basato su Java e con l'usilio di Hibernate. Per il Database è stato usato MySQL, che si integra con Spring-boot.

Per l'implementazione di Controller e View è stato usato il framework Next js basato sul più conosciuto React, con la caratteristica di facilitare la divisione tra View e Controller, come linguaggio è stato usato Typescript al posto di Javascript per mantenere la tipizzazione della progettazione.

4.2 Implementazione backend

Nel backend sono particolarmente rilevanti i commenti, in quanto si è cercato di rendere il codice più leggibile possibile per la possibilità di sviluppi futuri. Di seguito alcuni esempi.

```
/// getAll
///
/// Return List<Segnalazione>
///
/// Note: Questo metodo restituisce la lista di tutte le segnalazioni esistenti.
///
/// Mapping: /pawwy/segnalazione/all
///
@GetMapping("all")
public List<Segnalazione> getAll() {
    return segnalazioneRepository.findAll();
}
```

Figura 4.1: metodo che restituisce la lista delle segnalazioni

```
/// getColore
///
/// Return String
///
/// Note: Questo metodo restituisce il colore dell'animale del parametro id.
///
/// Mapping: /pawwy/segnalazione/get_colore/{id}
///
/// Parameters:
///
/// * [long] id:
///   id
@GetMapping("get_colore/{id}")
public String getColore(@PathVariable("id") long id) {
    Segnalazione segnalazione = getById(id);
    return segnalazione.getColore();
}
```

Figura 4.2: metodo che restituisce il colore di un animale di una segnalazione

```
/// setColore
///
/// Note: Questo metodo cambia il colore dell'animale del parametro id.
///
/// Mapping: /pawwy/segnalazione/set_colore/{id}/{colore}
///
/// Parameters:
///
/// * [long] id:
///   id
///
/// * [String] colore:
///   colore
@GetMapping("set_colore/{id}/{colore}")
public void setColore(@PathVariable("id") long id,
    | | | | | @PathVariable("colore") String colore) {
    Segnalazione segnalazione = getById(id);
    segnalazione.setColore(colore);
    segnalazioneRepository.save(segnalazione);
}
```

Figura 4.3: metodo che cambia il colore di un animale di una segnalazione

4.3 Implementazione frontend

I Controller si occupano della comunicazione con il Backend e con le API [3].

Per la grafica delle view è stato usato Tailwind css, di seguito è possibile vedere alcuni screenshot delle schermate più rilevanti.



Figura 4.4: screenshot della schermata iniziale

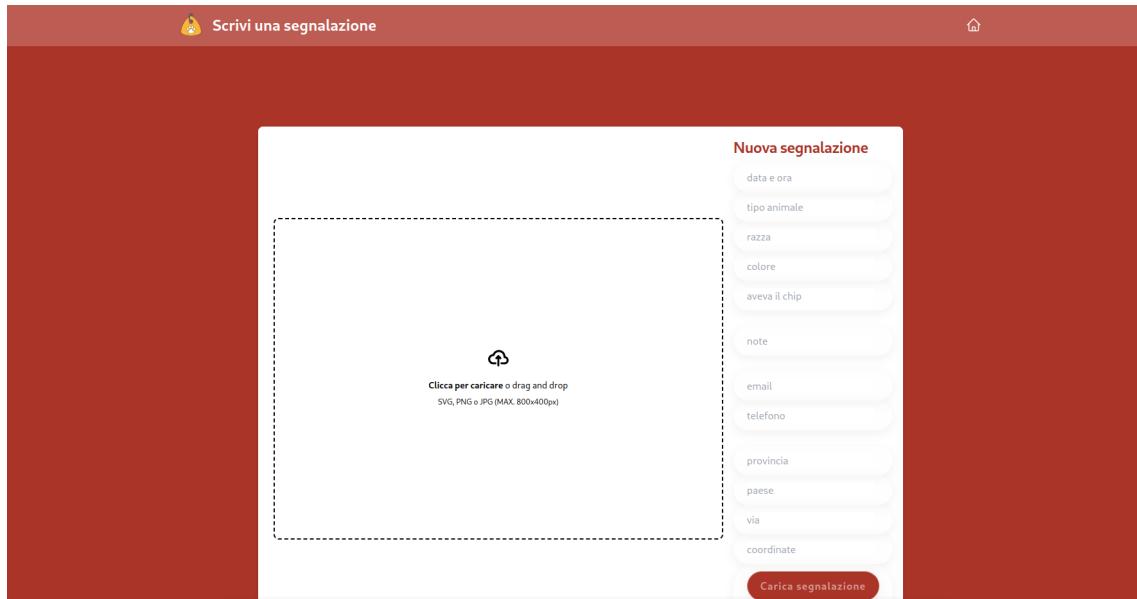


Figura 4.5: screenshot della schermata per aggiungere una segnalazione

Domanda 1 di 6 Hai qualche nano malefico in casa? <input checked="" type="radio"/> Sì <input type="radio"/> No	Domanda 2 di 6 Come preferisci che sia? <input checked="" type="radio"/> Cucciolino, bisognoso di tante attenzioni e coccole <input type="radio"/> Giovane e pieno di energie <input type="radio"/> Vecchietto e nulla facente	Domanda 3 di 6 Ti va bene che abbia qualche disabilità o non ne vuoi sapere mezza? <input checked="" type="radio"/> Prima <input type="radio"/> Seconda
Domanda 4 di 6 Che animale vuoi? <input checked="" type="radio"/> Un amico fedele <input type="radio"/> Uno autonomo e menefreghista <input type="radio"/> Uno carino da guardare <input type="radio"/> Voglio sfidare la sorte	Domanda 5 di 6 Hai qualche rugoso rompiscatole in casa? <input checked="" type="radio"/> Sì <input type="radio"/> No	Domanda 6 di 6 Dove abiti? <input checked="" type="radio"/> Una reggia degna di un imperatore <input type="radio"/> Una casa in mezzo al niente <input type="radio"/> Un appartamento decente <input type="radio"/> Un buco <input type="radio"/> In carcere

Figura 4.6: screenshot della schermata del test

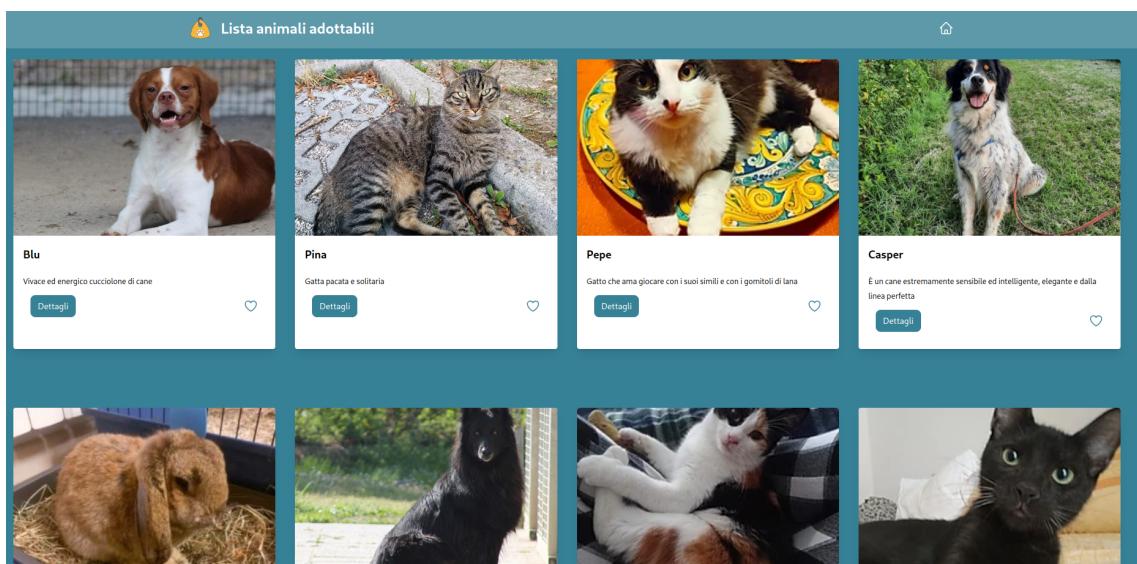


Figura 4.7: screenshot della schermata della lista di animali adottabili

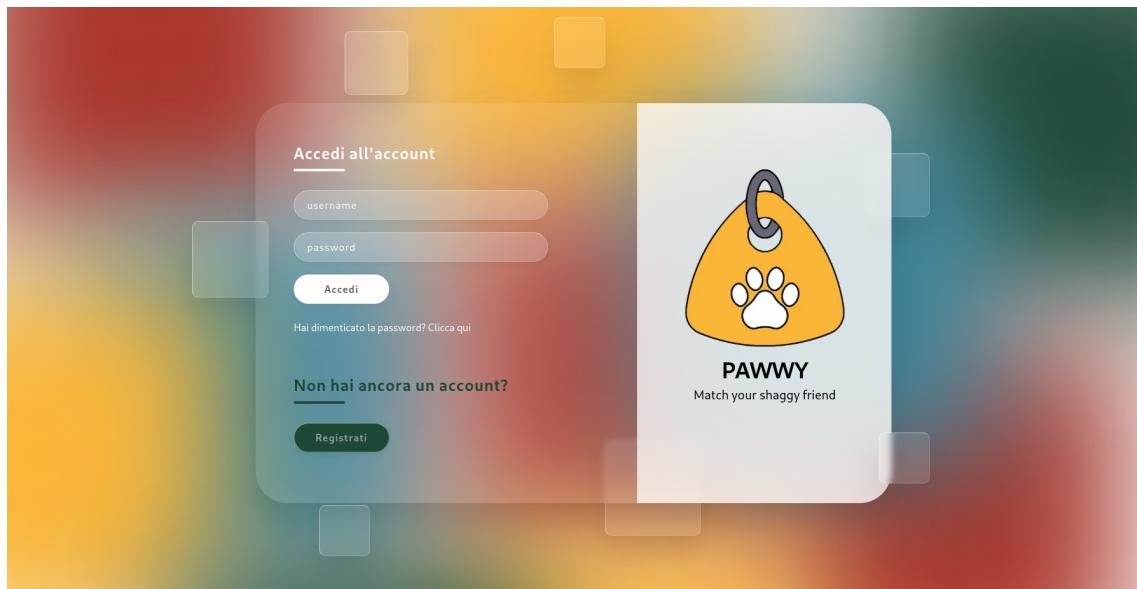


Figura 4.8: screenshot della schermata di autenticazione

Conclusioni

In questo lavoro di tesi è stato affrontato il processo di riprogettazione, correzione e miglioramento del documento di progetto [1] dell'applicazione web Pawwy, inoltre è stato espanso il prototipo implementativo, migliorandone il codice sorgente e introducendo le funzionalità mancanti nella prima versione.

In primis è stato analizzato il documento di progetto preesistente individuando le criticità, successivamente si è cercato di risolvere al meglio questi problemi senza compromettere la struttura della piattaforma. La riprogettazione ha apportato diversi miglioramenti tra i quali:

- modifiche alla Tabella requisitie al Vocabolario
- correzione degli Scenari, originariamente troppo legati all'interfaccia
- aggiunta di un Caso d'Uso “AggiungiAPreferiti”
- modifiche alla Tabella Valutazione dei Beni e ai Requisiti di Sicurezza
- variazioni del grado di complessità delle Funzionalità
- correzioni delle View nella Tabella delle Maschere
- eliminazione della classe Form nel Modello del Dominio
- aggiunte nel Diagramma dei Package
- correzioni e modifiche nei Diagrammi di Dettaglio e di Sequenza
- eliminazione della classe Tipologia dalla Persistenza

In secondo luogo è stato aggiornato il prototipo, le tecnologie utilizzate per l'implementazione sono rimaste quelle precedenti, per il Backend il framework Spring-boot e per il Frontend Next js, è stata migliorata la leggibilità del codice del Backend con l'aggiunta di commenti e sono state aggiunte le schermate del Frontend mancanti.

Eventuali sviluppi futuri potranno includere la possibilità per gli utenti di comunicare tra loro tramite una chat pubblica sotto le segnalazioni, per ottenere maggiori dettagli sugli animali scomparsi ed un metodo per permettere ad un utente di contattare un'agenzia di adozioni direttamente dalla piattaforma web.

Bibliografia

- [1] Documento di Progetto Pawwy di Clarissa Bovo, Annamaria Martelli, Antonio Zara.
- [2] <https://virtuale.unibo.it/> Ingegneria del Software T di Marco Patella
- [3] <https://rescuegroups.org/services/adoptable-pet-data-api/>

Ringraziamenti

Concludo questo elaborato facendo i dovuti ringraziamenti.

In primis ci tengo a ringraziare il mio relatore, il professor Marco Patella, per i suoi consigli e il suo tempo che mi hanno portata alla stesura di questa tesi.

In secondo luogo vorrei ringraziare la mia famiglia, i miei genitori per essermi stati accanto, in particolare ringrazio mia mamma per avermi sempre spronata a fare del mio meglio e per avermi trasmesso il suo bisogno di fare quindicimila cose contemporaneamente (no non è vero, per questo non la ringrazio). E ringrazio mio papà per aver convinto mia mamma all'epoca a farmi studiare informatica e per tenermi allenata con costanti quesiti informatici... come farei senza di lui...

Poi ci tengo a ringraziare le mie fantastiche sorelle, Lucrezia e Rachele, grazie a loro ho sviluppato l'arte della pazienza infinita, che mi è stata molto utile in questi anni di università. Nonostante tutto sono le mie migliori amiche e non le cambierei per nulla al mondo.

Voglio ringraziare il mio fidanzato, Alvise, che per questi anni ha ripetuto in loop che ce l'avrei fatta a passare quel esame che mi sembrava insuperabile, che sarei riuscita a conciliare lavoro e studio nonostante le difficoltà e lo stress, e che un giorno il mio impegno sarebbe stato ripagato.

Non l'ho mai ascoltato.

Non posso non ringraziare e scusarmi con il mio coinquilino, Mattia, che ha reso questi ultimi tre anni decisamente migliori di quanto lo sarebbero stati senza di lui, e che ha dovuto sopportare la versione peggiore di me durante le sessioni d'esami (una menzione particolare va a quella in cui abbiamo avuto il covid, una quarantena tosta).

Inoltre ringrazio tutte le persone che hanno migliorato la mia vita in questi anni. Tra cui i miei nonni a cui voglio un mondo di bene. I genitori di Alvise che sono una seconda famiglia per me. I miei zii che mi hanno sostenuta in questo percorso universitario. I miei vecchi amici che ci sono sempre per me, in particolare il mio finto coinquilino Paolo, Rickie il delfino, Laura la mia amica d'infanzia. E tutti i nuovi amici che ho incontrato all'università, in particolare Antonio con cui ho fatto il progetto iniziale e che si è pazientemente lasciato bullizzare in questi anni, Alice che mi ha reso una persona più socievole, Francesca la mia bolognesissima guida turistica, Margherita, Fabio, Dario, Alessio e Victoria. Ringrazio anche Rambo, la Kuroneko e Zilla.

Infine voglio ringraziare il mio gatto Lucifero, per essere la cosa più carina che io abbia mai visto e grazie a lui (un trovatello che mi ha cambiato la vita) ho avuto l'idea per questo progetto.