

TakeMyPet



A.Accorinti, A.Di Carmine, G.Nesci

SOMMARIO

Introduzione

Caratteristiche dell'app

Casi d'usi

Entity relationship

Wireframe

Class diagram

Web App

Codice nel dettaglio

Applicazione Android

Codice nel dettaglio

Richieste Web

Sviluppi futuri





Software utilizzati:

- ECLIPSE, VISUAL STUDIO CODE, ANDROID STUDIO
 - MYSQL, HEIDISQL
 - STARUML, GIMP, BALSAMIQ
 - POWER POINT



Linguaggi utilizzati per la progettazione:

- JAVA: Servlet, JPA, Hibernate, Maven, Jetty, Jackson, JUnit, Logback, Javax
- JAVASCRIPT: JQuery, Ajax
- ANDROID: Retrofit, Fragment, SQLite
- HTML, XML, CSS
- SQL

Introduzione

Lo scopo di TakeMyPet è mettere in contatto in modo semplice ed intuitivo, i possessori di animali, con poco tempo da dedicare alla loro cura, con appassionati con del tempo libero, ma senza le risorse economiche per avere un animale. Non è un rapporto lavorativo, ma uno scambio di favori che avviene in modo gratuito(questo ci differenzia da altre app già esistenti).

Inoltre permette di creare e partecipare ad eventi, dando così l'opportunità di conoscere altre persone con la stessa passione.

È possibile iscriversi
gratuitamente al servizio,
il quale permette al pet
sitter di accudire l'animale
che preferisce tra quelli
proposti dai proprietari
vicini a lui.



Viceversa i proprietari hanno la possibilità di scegliere, tra i pet sitters che hanno dato la loro disponibilità, quello che li convince di più in modo tale da poter lasciare il proprio " in mani sicure.



La scelta è intuitiva ed avviene attraverso lo “swap” di profili proposti dal sistema in base ai parametri inseriti.

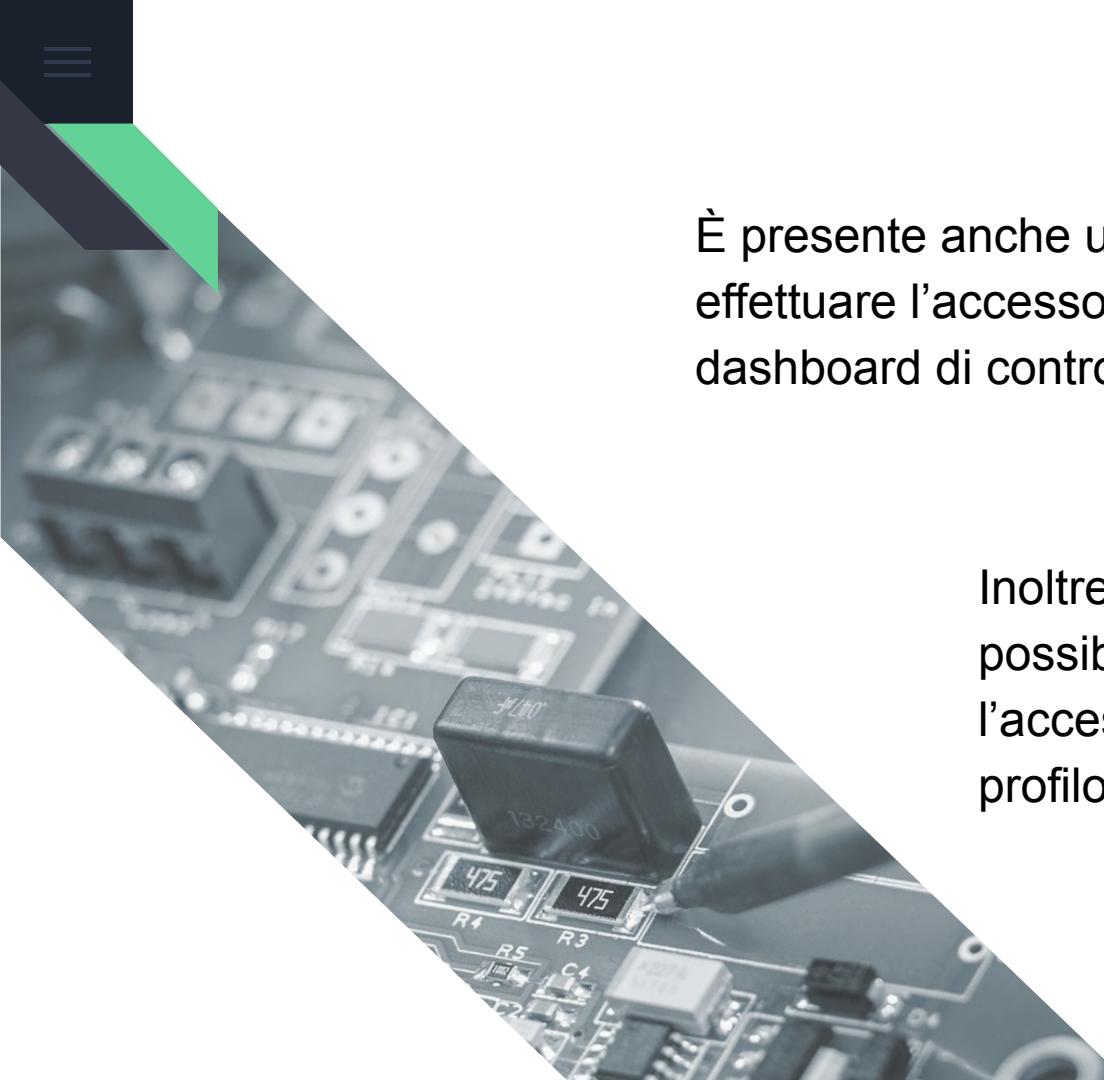
Avvenuto il match gli utenti vengono messi in contatto e possono incominciare il loro scambio di favori.





Al momento dell'iscrizione è possibile scegliere tra i due tipi di utente.

Una volta effettuato l'accesso l'app cambierà la sua interfaccia e le sue funzionalità a seconda del tipo di utente.



È presente anche un sito web, in cui l'admin può effettuare l'accesso e lavorare sulla propria dashboard di controllo dell'app.

Inoltre, per gli utenti, nel sito è possibile registrarsi, effettuare l'accesso e modificare il proprio profilo.



Funzionalità:



- ANNUNCI: crea annunci per fare accudire i tuoi animali (Proprietario).
- SWAP: scegli tra gli annunci disponibili (Pet Sitter) o tra i Pet Sitter che hanno accettato il tuo annuncio (Proprietario).

- 
- PREFERITI: gli annunci o i Pet Sitter scelti nello swap vengono messi in preferiti, in modo da rivedere le tue scelte e confermare con calma.
 - EVENTI: crea e partecipa ad eventi (ad esempio “raduno per bassotti a Villa Villetta”), per conoscere altre persone con la tua passione.
 - CHAT: trovato il match tra Proprietario e Pet Sitter i due utenti vengono messi in contatto nell'apposita chat.

Fasi di sviluppo

Individuare gli attori

Definizione dei requisiti e fattibilità

Wireframe

Analisi dei casi d'uso

ER

Progettazione del database

Javascript

View

Model

Ajax

Bootstrap

Controller

MVC

Class diagram

Servlet

Fragments

Implementazione Activities

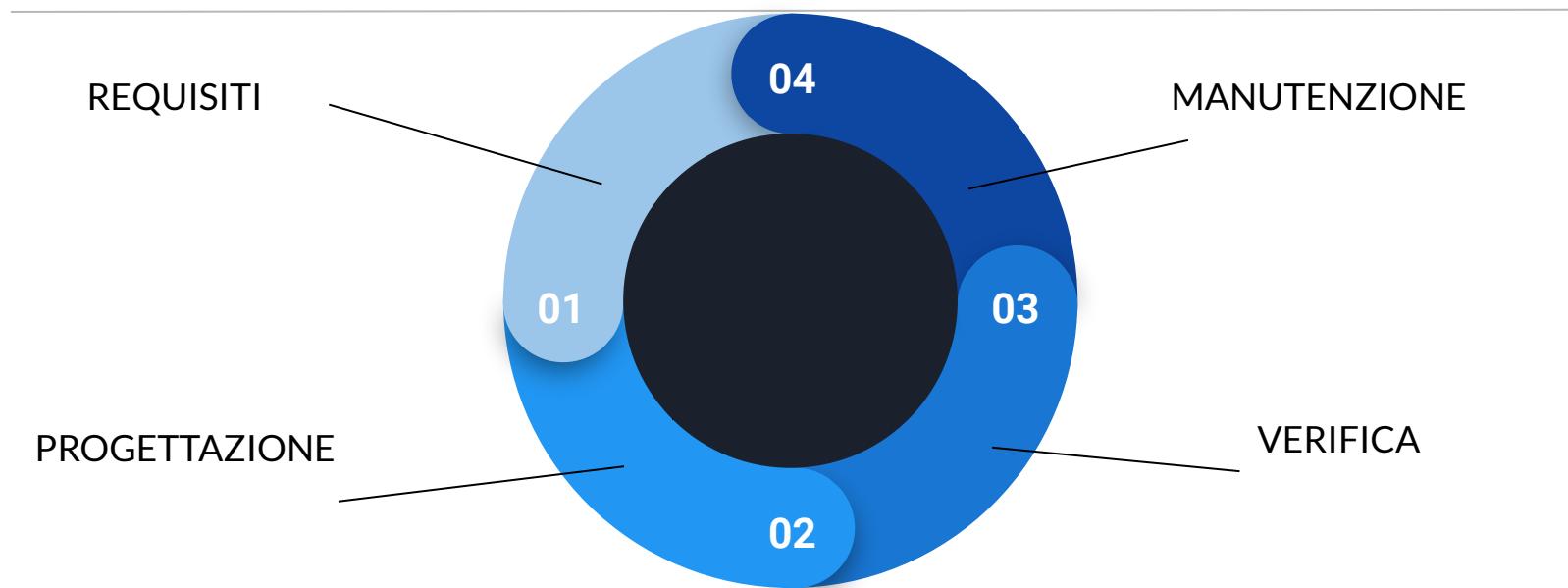
Retrofit

Implementazione applicazione Java

Implementazione applicazione Android

Software Developer Life Cycle

Per lo sviluppo del nostro software ci siamo basati sul modello a cascata, seguendo i seguenti passaggi:



Casi d'uso

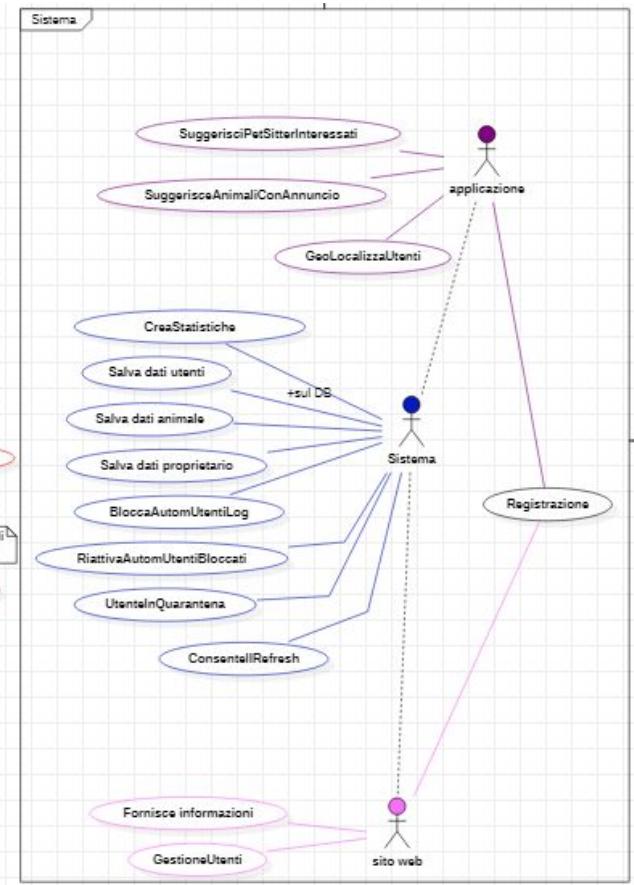
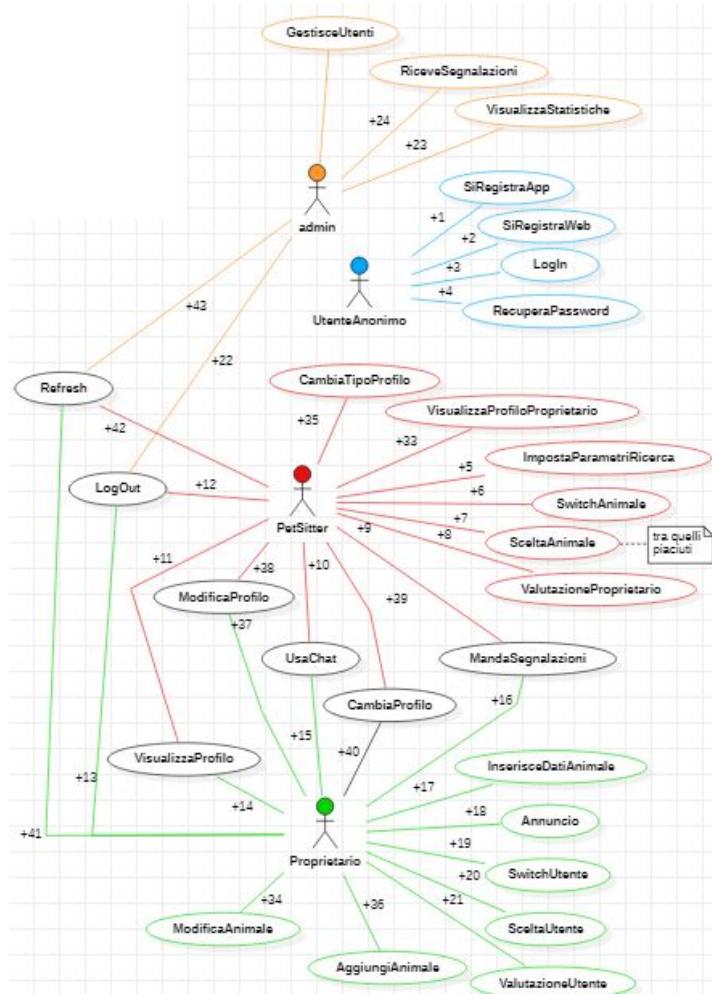


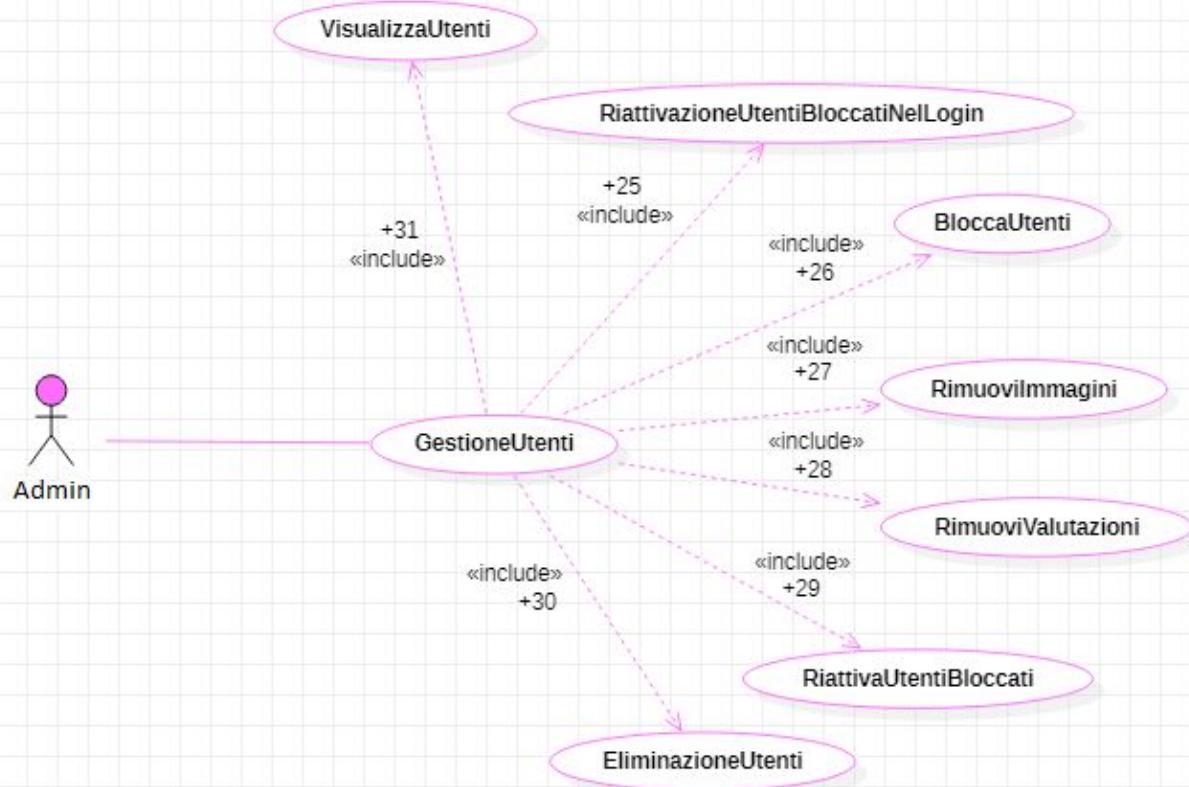
Abbiamo individuato in primo luogo gli attori del progetto, ovvero tutte le parti che hanno delle interazioni con il sistema.

Tramite gli attori abbiamo stabilito le funzioni che possono compiere, utilizzando frasi e verbi.

Attori principali

- 01 PROPRIETARIO: principalmente inserisce annunci per i propri animali, sceglie i pet sitter, crea e partecipa ad eventi, invia valutazioni e segnalazioni.
- 02 PETSITTER: principalmente sceglie gli annunci, crea e partecipa ad eventi, invia valutazioni e segnalazioni.
- 03 ADMIN: gestisce le segnalazioni, modera gli utenti e gli eventi, visualizza le statistiche dell'applicazione.





Descrizione dei casi d'uso



Descrizione dei casi d'uso

Per ogni caso d'uso abbiamo provato a descrivere il modo in cui la nostra app

dovrebbe funzionare indicando le informazioni generali, attivazione, svolgimento, flusso principale e flussi alternativi ([tutte le descrizioni sono disponibili qui](#)).

Un esempio del template utilizzato





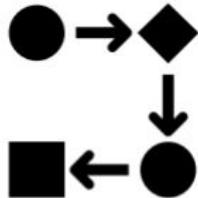
Informazioni generali

ID	3
Titolo	Login
Versione	0.0.4
Autore	Nesci, Accorinti, Di Carmine, Rossi
Data ultima revisione	2020/09/01
Autore ultima revisione	Nesci, Accorinti, Di Carmine, Rossi
Riferimenti e documenti collegati	
Note	

Attivazione



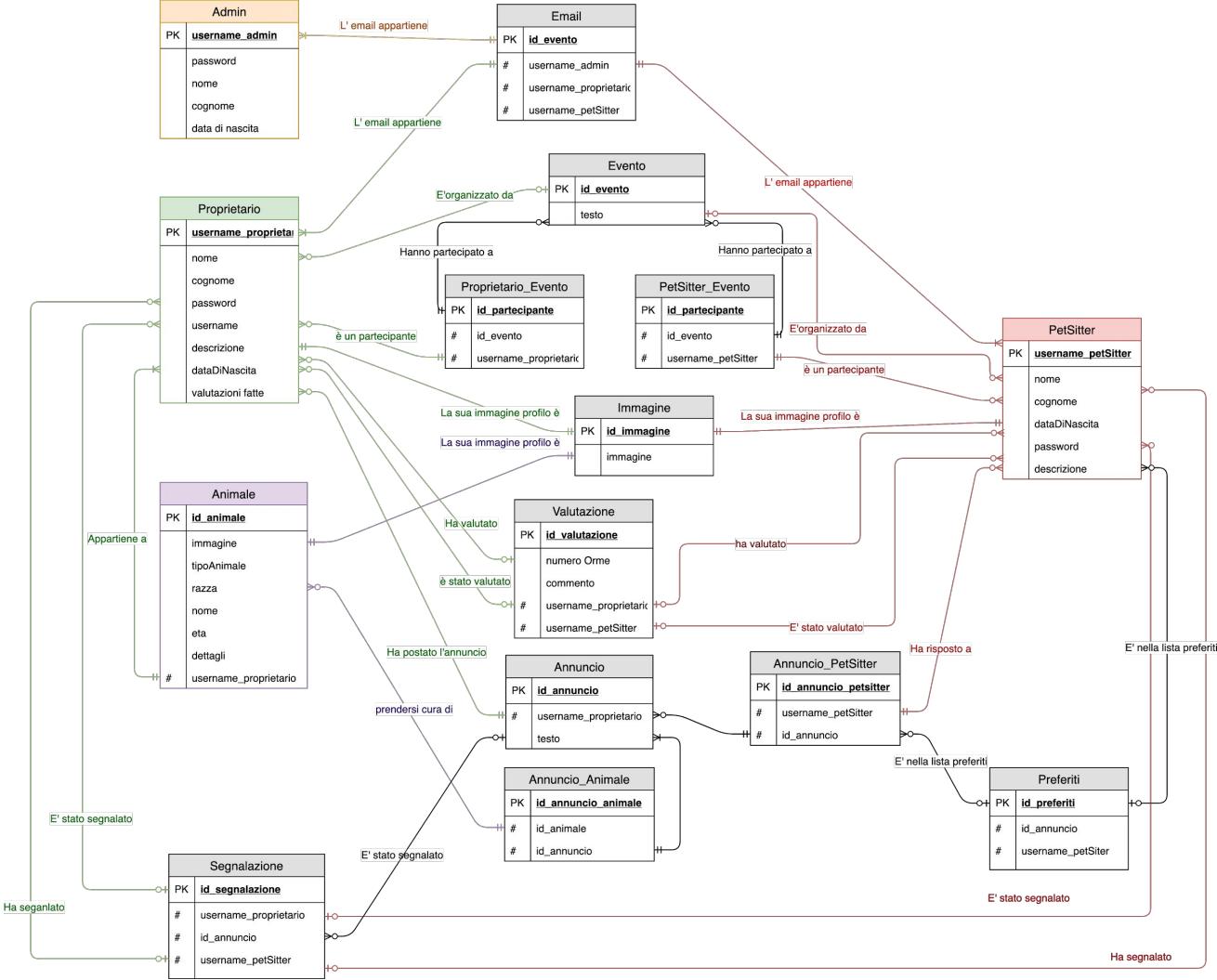
Trigger	Apertura app, click su login sul web, logout(solo per app)
Frequenza di utilizzo	Low
Benefici organizzativi	
Attore principale	UtenteAnonimo
Attori secondari	Sistema
Pre condizioni	Pagina di login
Post condizioni	Homepage web o homepage app



Svolgimento

Flusso principale	<ol style="list-style-type: none"> 1. Utente apre l'app 2. Sistema propone inserimento e-mail e password 3. L'utente conferma i dati 4. Il sistema verifica i dati sul DB, se corretti crea una sessione di lavoro per l'utente 5. Utente viene rediretto sulla home page
Flusso Alternativo 1	<p>Punto 4: l'utente inserisce dati errati Il sistema registra il numero di tentativi falliti e riporta l'utente al punto 2.</p>
Flusso eccezionale 1	<p>Punto 4: l'utente inserisce dati errati per 3 volte Il sistema blocca l'utente e invia un'e-mail con le info di sblocco.</p>
Flusso eccezionale 2	<p>Punto 4 : l'utente non ha confermato l'email Il sistema visualizza un messaggio di avviso "L'indirizzo email non è stato confermato."</p>
Flusso eccezionale 3	<p>punto 4: il sistema verifica che l'utente è stato bloccato, manda un toast "il tuo account è stato bloccato"</p>

Entity Relationship





Modello entità relazione



Partendo dalle tre entità principali (Admin, Proprietario, Petsitter), abbiamo individuato i rispettivi attributi e abbiamo separato quelli che potevano rappresentare a loro volta un'altra entità.

Successivamente, a partire dall'analisi dei casi d'uso, siamo andati a definire le entità che ci sarebbero servite per salvare i dati per il funzionamento dell'app (Preferito, Evento, Annuncio, Segnalazione, Valutazione).

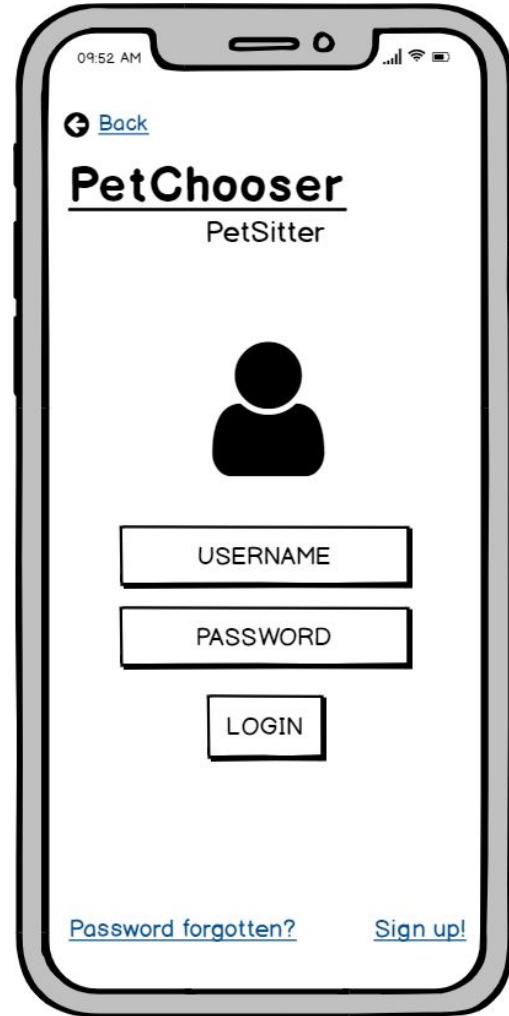
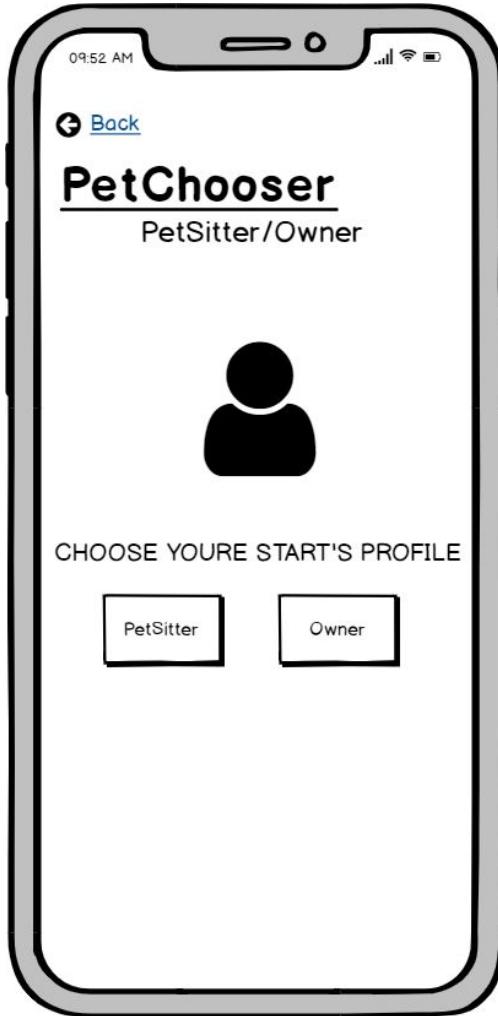
Ad ogni entità abbiamo assegnando una chiave primaria. Una volta definite le entità abbiamo individuato le relazioni che le legano, con la loro cardinalità e stabilito chiavi secondarie e tabelle ponte.

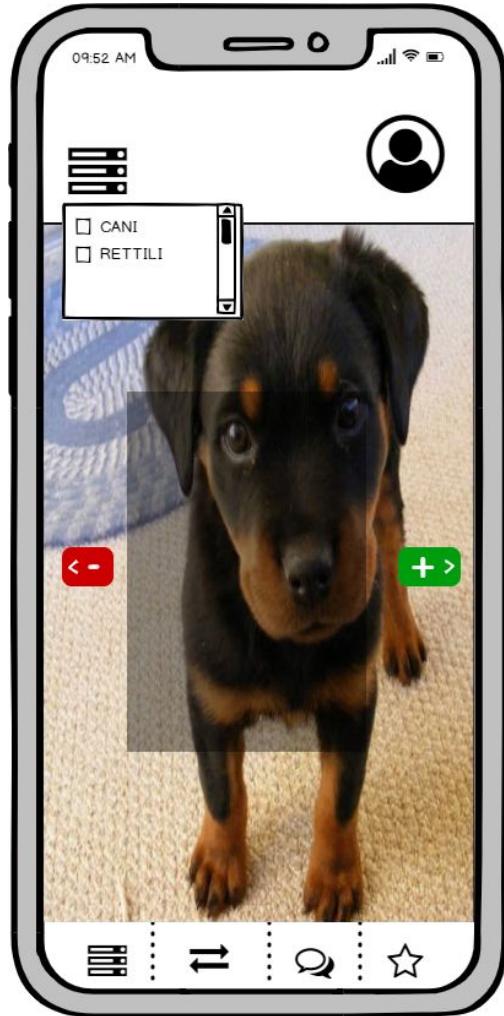
Wireframe

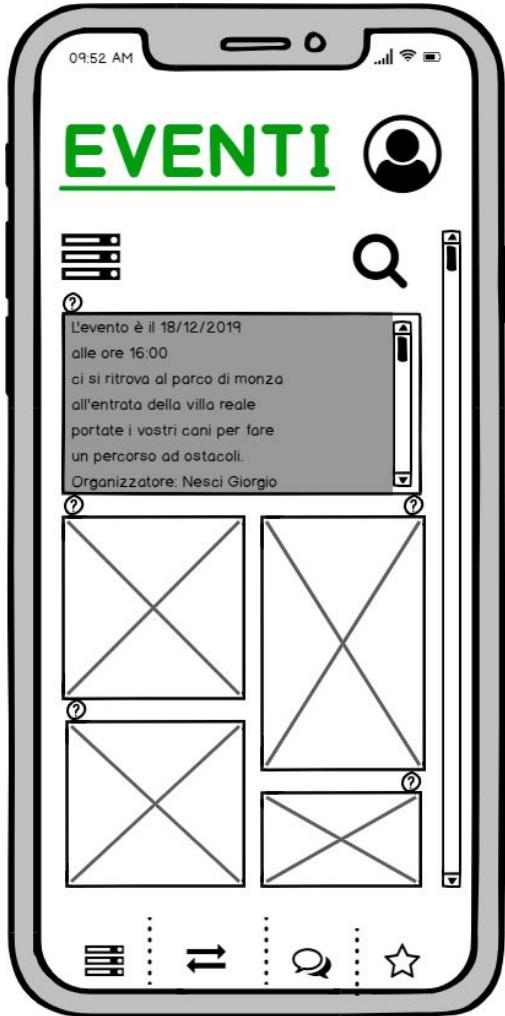
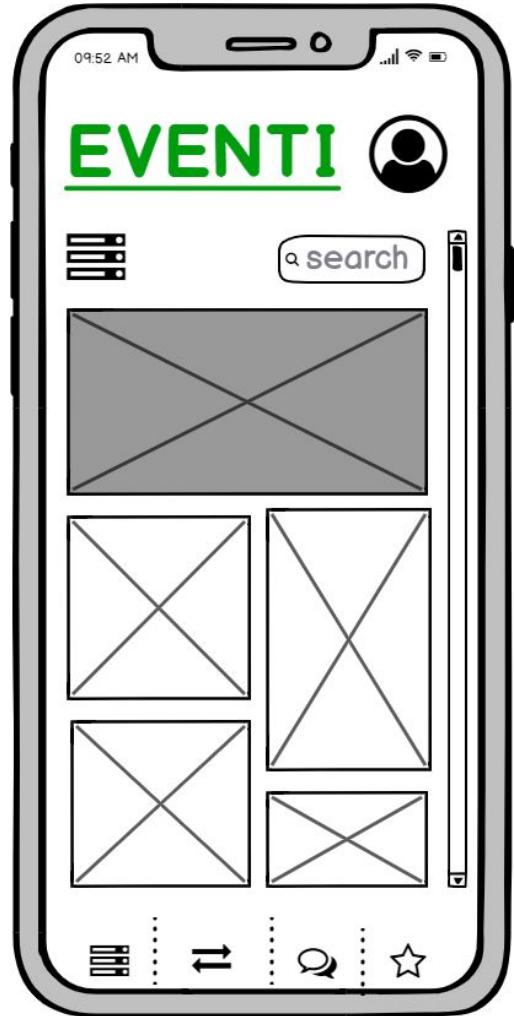


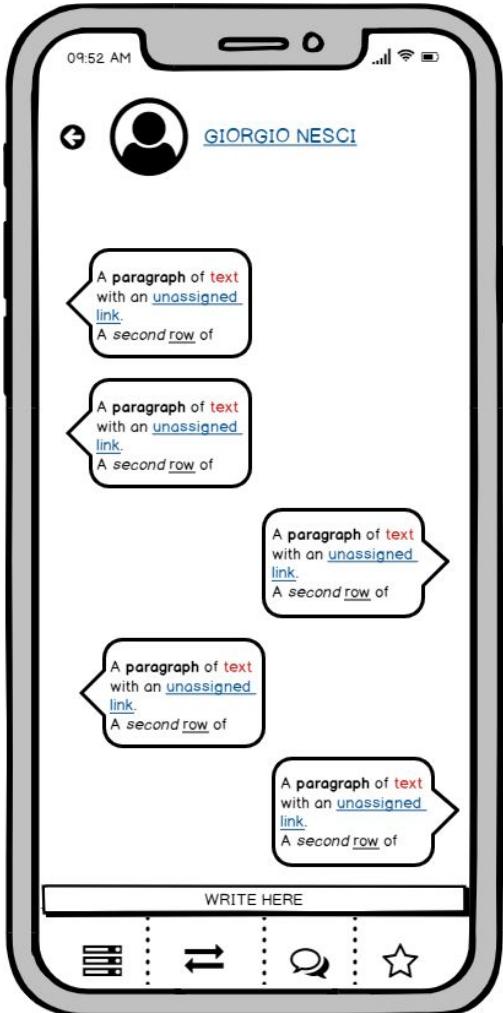
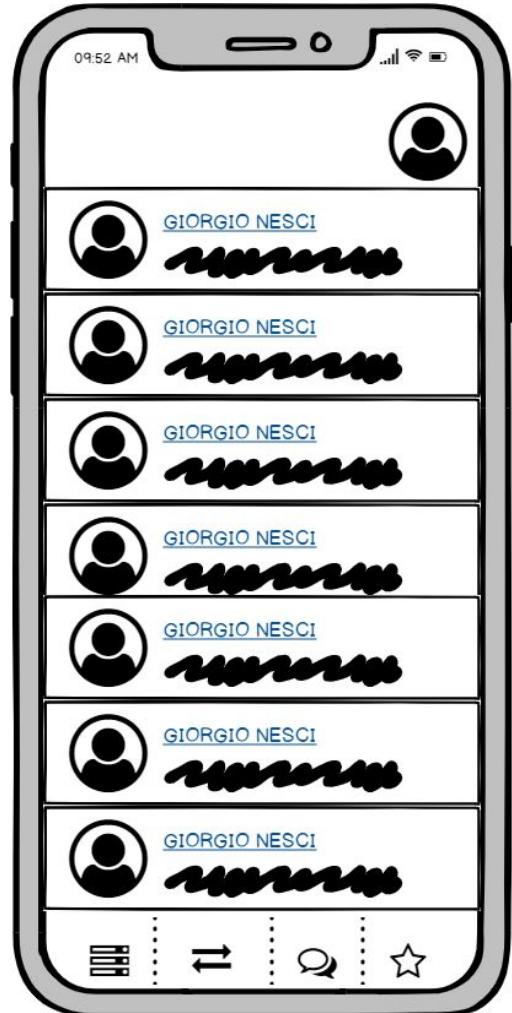
Prima di passare alla fase di progettazione, abbiamo immaginato la “User Experience” e la “User Interface” attraverso i wireframe.

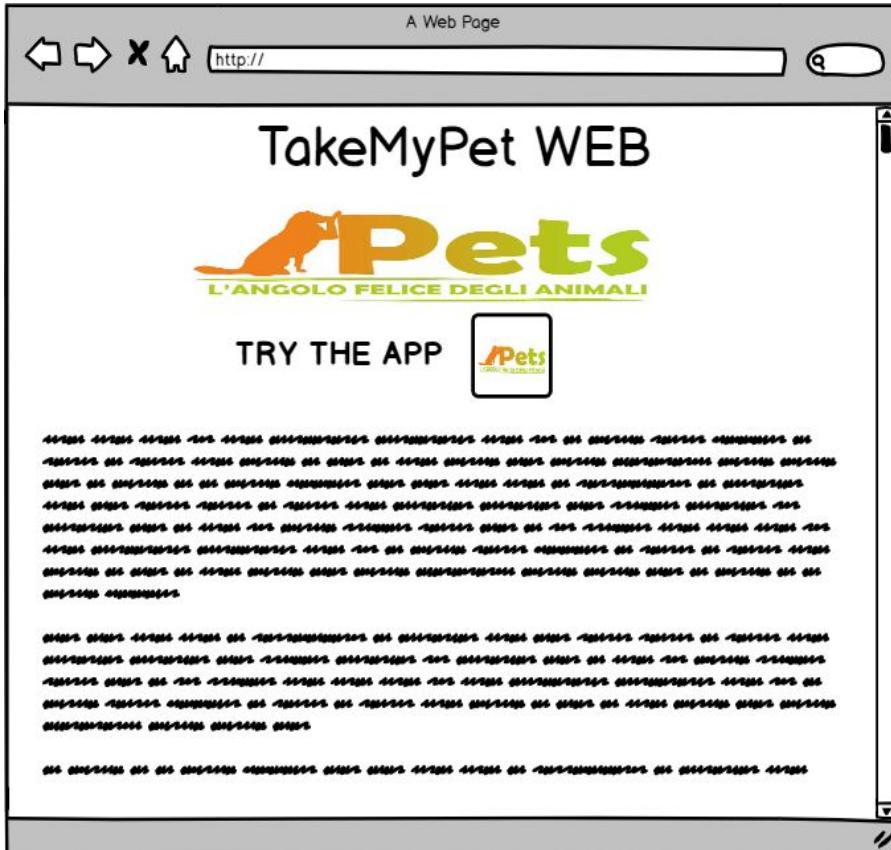
Abbiamo cercato di facilitare l'esperienza e l'utilizzo del software da parte dell'utente, utilizzando lo spazio a disposizione per rendere l'app più usabile ed intuitiva possibile.











A Web Page

<http://PetChooser/SignUp/>

[Back](#)

Insert Name: _____ *

Insert Surname: _____ *

Insert e-mail or Number: _____ *

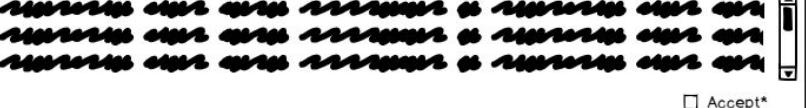
Insert password: _____ *

Insert password again: _____ *

Insert City: _____ *

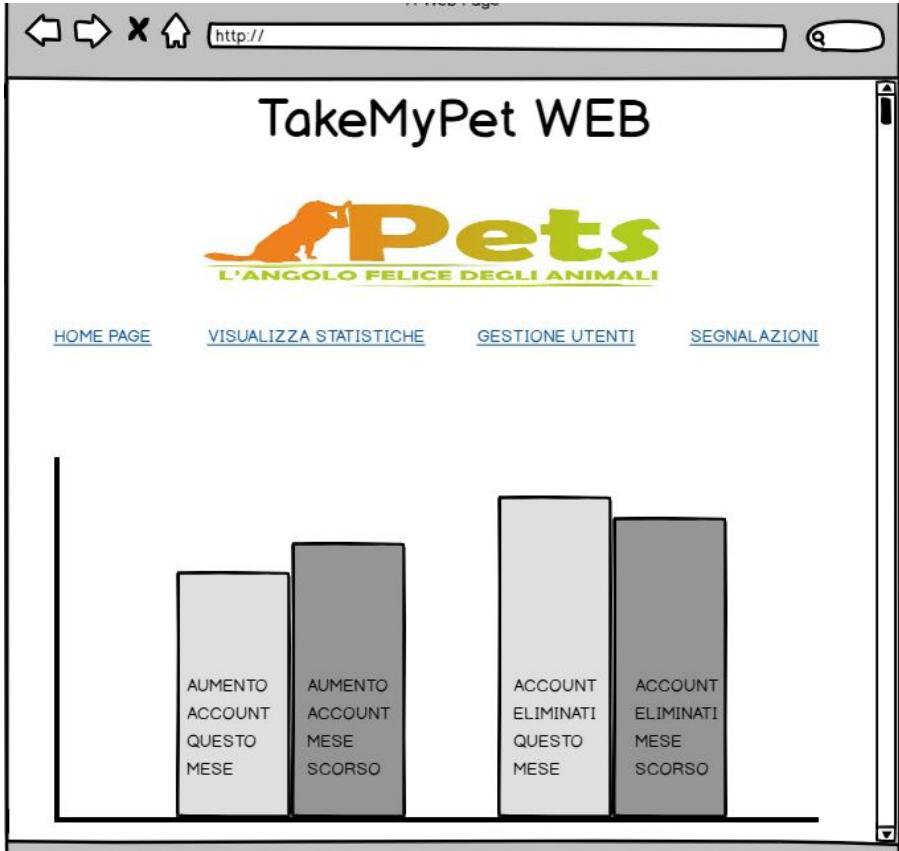
 I'm not a robot*

TERMS AND CONDITION



Accept*

SIGN UP



A Web Page

http://

TakeMyPet WEB

HOME PAGE [VISUALIZZA STATISTICHE](#) [GESTIONE UTENTI](#) [SEGNALAZIONI](#)

search

	Enrico Lo Verso	segnalazioni	blocca	elimina
	Domenico Riva	segnalazioni	blocca	elimina
	Maria Grazia Cucinotta	segnalazioni	blocca	elimina
	Anna Riva	segnalazioni	Sblocca	
	Marco Rossetti	segnalazioni	blocca	elimina
	Lorenzo	segnalazioni	blocca	elimina
	Antonio Stornaiolo	segnalazioni	Sblocca	
	Gaetano Giglio	segnalazioni	blocca	elimina
	Totò Onnis	segnalazioni	blocca	elimina
	Nino La Greca	segnalazioni	blocca	elimina
	Mingo De Pasquale	segnalazioni	blocca	elimina

This screenshot shows the 'GESTIONE UTENTI' (User Management) page of the application. It lists a table of users with their names and profile icons. To the right of each user entry are three buttons: 'segnalazioni' (Report), 'blocca' (Block), and 'elimina' (Delete). A search bar is located at the top right of the page.

A Web Page

http://

Nome Utente

segnalazione del DATA ELIMINA

A Web Page

http://

SEGNALAZIONE!

SEGNALAZIONE DEL 2020/03/01 NEI CONFRONTI DI GIORGIO NESCI

VISUALIZZA:

UTENTE:

NOME UTENTE IMMAGINE PROFILO IMMAGINE ANNUNCIO

SBLOCCA ELIMINA

Class diagram



Nella fase di progettazione abbiamo applicato i design pattern fondamentali per una buona riuscita del progetto.

Per la comunicazione abbiamo seguito il modello MVC (model, view, controller). I model sono stati realizzati attraverso l'incapsulation e la creazione di java beans.

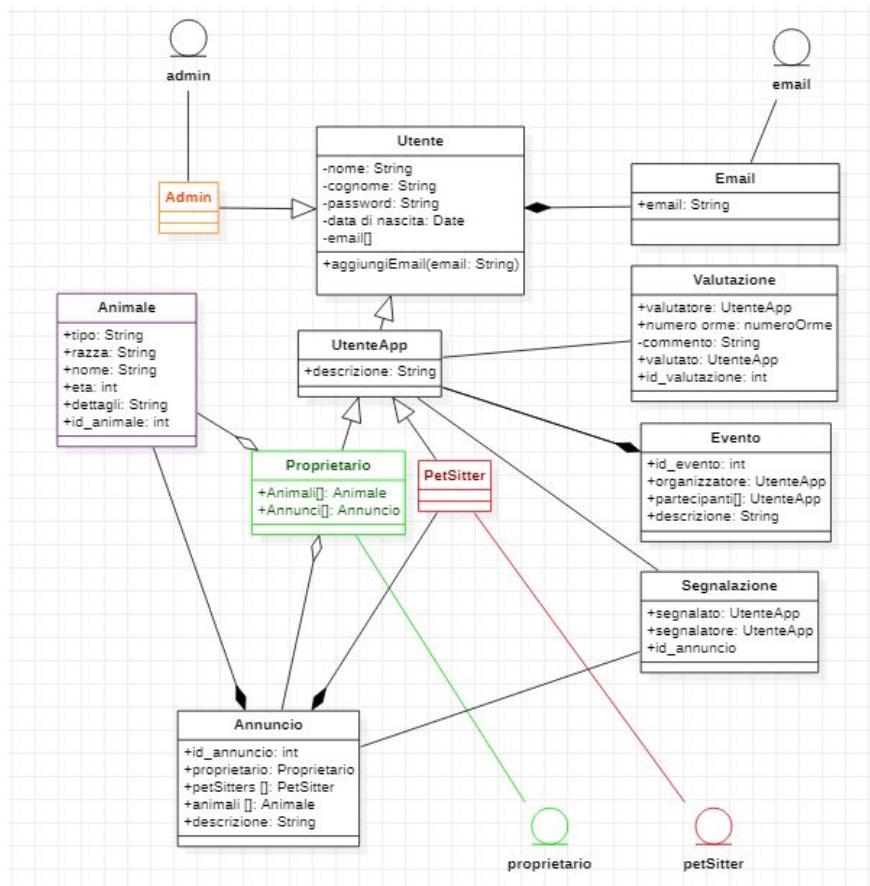
Le classi business, contenenti i metodi chiamati dai controller, sono state fatte a partire da interfacce stabilite in precedenza, creando una serie di “contratti” da rispettare (design by contract) .



Abbiamo inoltre usato un singleton per l'entity manager della jpa (per evitare di occupare troppo spazio nella RAM) ed una factory per la creazione degli utenti. Sia le business, che le interfacce e i model seguono un albero di ereditarietà che è stato strutturato astraendo e trovando degli elementi comuni nelle classi.

I controller sono stati creati con le servlet, che risponderanno alle varie view, gestite da jQuery (nel caso del web) e da Android (nel caso dell'app).

In tutto il progetto sono stati tenuti sempre bene a mente i 5 principi SOLID e più in generale il principio di keep it simple.

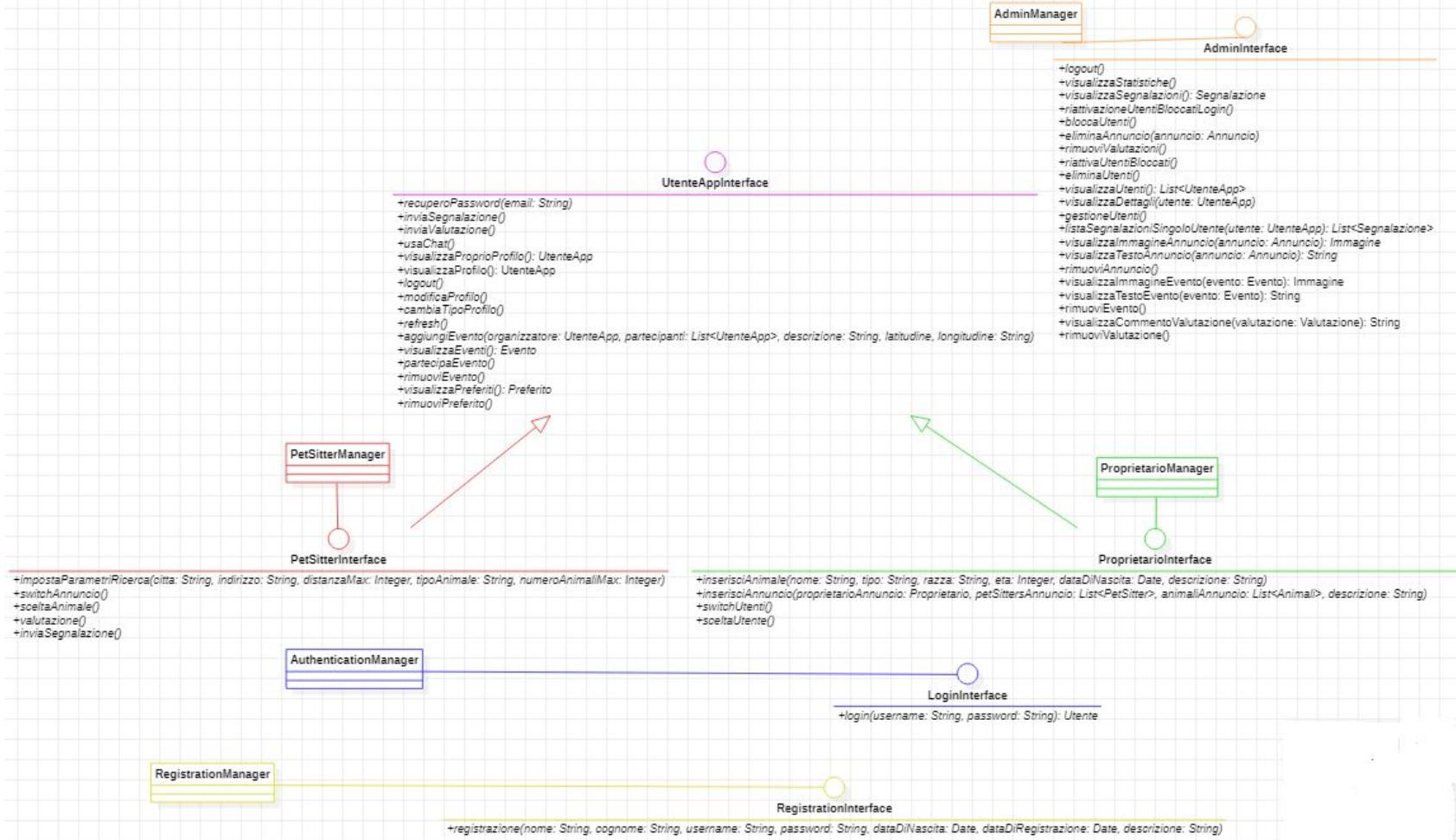




Nei model abbiamo definito una superclasse Utente che racchiude gli aspetti comuni per tutti gli utilizzatori della nostra applicazione.

Questa classe viene estesa da Admin e da UtenteApp che a sua volta viene estesa dalle classi più specifiche Proprietario e PetSitter.

I model degli utenti contengono al loro interno attributi che li relazionano (e creano chiavi secondarie nel database) ai model delle funzionalità dell'app e viceversa (ad esempio Annuncio, Animale, Segnalazione, Email).





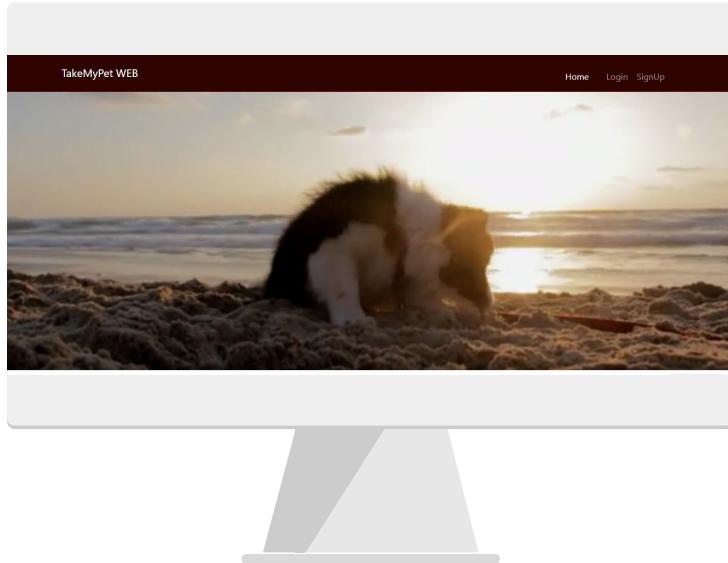
Le interfacce sono state create in base agli utenti (in quanto le funzionalità possono variare nel tempo).

UtenteApplInterface ha al suo interno tutti i metodi astratti comuni ai due tipi di utente ed è estesa da ProprietarioInterface e PetSitterInterface, che contengono i metodi più specifici.

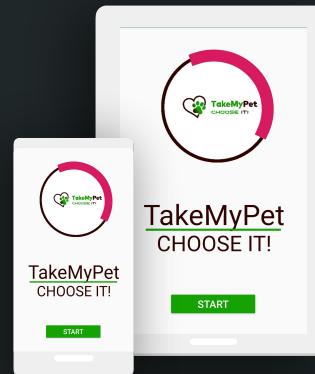
Abbiamo creato anche una interfaccia per le funzioni dell'admin, per la registrazione e per il login.

Tutte le interfacce vengono implementate dalle omonime classi business e le classi ProprietarioManager e PetSitterManager, estendono UtenteAppManager, in modo da non dover implementare gli stessi metodi comuni in due classi diverse.

Web app



Android app





Sito web

Dal sito, oltre a trovare informazioni sull'applicazione, ci si può registrare e scaricare l'app dal play store.



È possibile inoltre effettuare il login, visualizzare il proprio profilo ed effettuare modifiche.



Dynamic web page

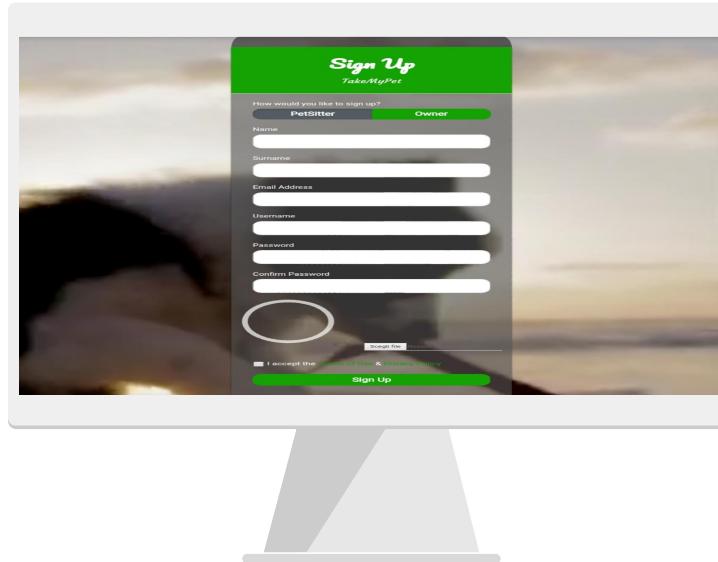
Il sito web è reso dinamico dall'utilizzo di Bootstrap 4 e JQuery.

Bootstrap è una libreria opensource per la creazione di pagine web dinamiche che utilizza HTML, CSS, and JS.



Sito web

Nella registrazione la web app controlla che un utente con lo stesso Username non esista già, se i campi sono riempiti e se la mail inserita è nel formato giusto.

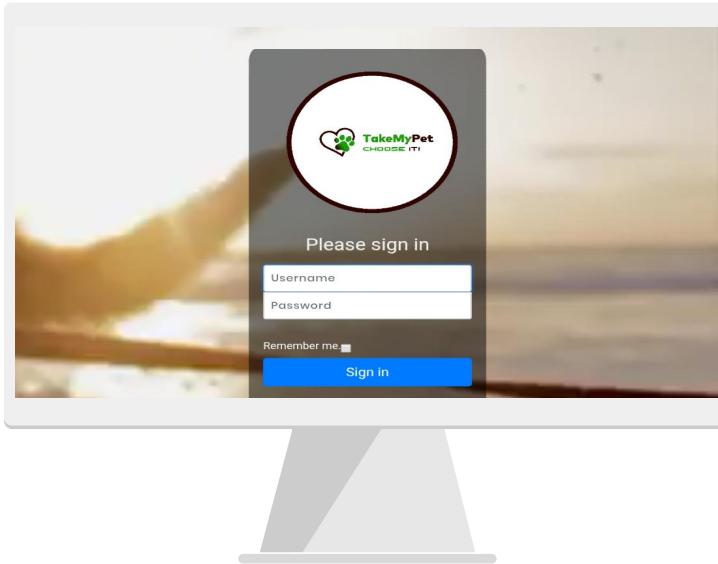


Permette anche il caricamento dell'immagine del profilo che verrà salvata lato server tramite l'invio del suo Base64. Nel DB verrà salvato solo l'url.



Sito web

Nel login il server controlla che l'utente esista nel database e carica una pagina diversa a seconda che si tratti di un utente o di un admin.

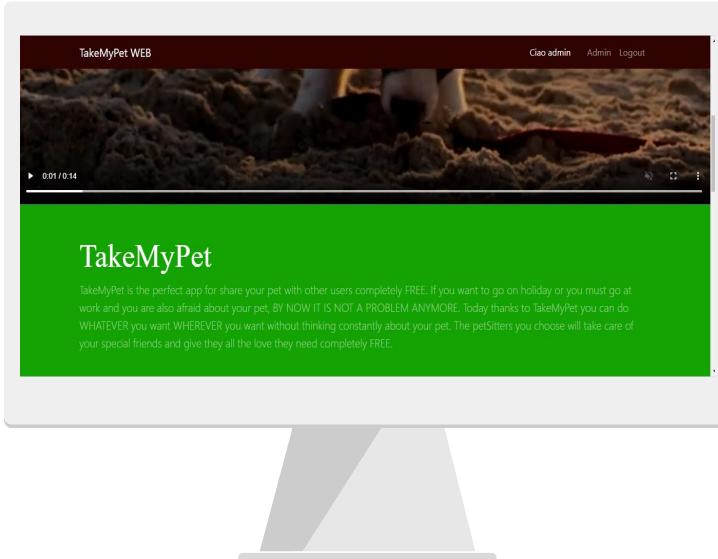


Se viene inserita per dieci volte una password sbagliata, l'account viene bloccato e viene inviata all'utente una mail contenente il link per accedere alla pagina di sblocco ed un codice alfanumerico (generato casualmente) da inserire per sbloccare l'account.



Sito web

Una volta fatto il login l'admin può accedere alla dashboard di controllo dell'app.

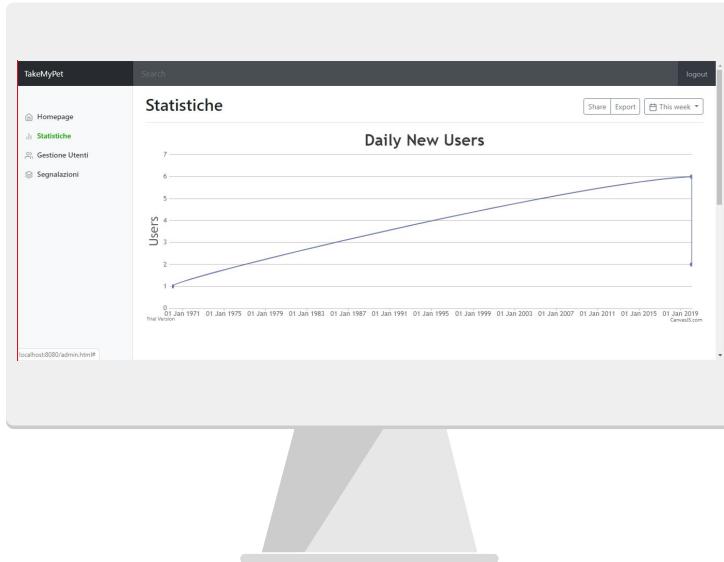


Attraverso la dashboard l'admin può visualizzare statistiche, gestire gli utenti e visualizzare le segnalazioni



Sito web

Nella home della dashboard è presente un grafico contenente le statistiche



La pagina chiama una servlet che fa una query nel database.

In base alla risposta si genera un grafico con ChartJs(una libreria per Javascript)



Sito web

Attraverso la sezione “gestione utenti” l’admin visualizza un elenco con tutti gli utenti registrati. La tabella utilizza DataTableJS.

The screenshot shows a web application interface titled "Gestione Utenti". On the left, there's a sidebar with links: "Homepage", "Statistiche", "Gestione Utenti" (which is highlighted), and "Segnalazioni". The main content area has a title "Gestione Utenti" and a search bar. Below it is a table with the following columns: Username, Nome, Cognome, Bloccato, Doppio profilo, Elimina, Blocca, and Sblocca. The table contains 9 entries. At the bottom of the table, it says "Showing 1 to 9 of 9 entries". The URL in the address bar is "localhost:8080/admin.html#".

Username	Nome	Cognome	Bloccato	Doppio profilo	Elimina	Blocca	Sblocca
ale	andrea	accorinti	false	false	Elimina	Blocca	Sblocca
alex			true	false	Elimina	Blocca	Sblocca
andrea	andrea	accorinti	false	false	Elimina	Blocca	Sblocca
i	andrea	accorinti	false	false	Elimina	Blocca	Sblocca
gio	gio	gio	false	false	Elimina	Blocca	Sblocca
l	l	l	false	false	Elimina	Blocca	Sblocca
provaling	provaling	sdfg	false	false	Elimina	Blocca	Sblocca
samuele	dsg	sdgf	false	false	Elimina	Blocca	Sblocca
sdghgreh	dsf	sdgf	false	false	Elimina	Blocca	Sblocca

In questa tabella si posson bloccare, sbloccare ed eliminare profili attraverso chiamate Ajax che comunicheranno al server di effettuare modifiche sul DB.



Applicazione dinamica

Per la comunicazione con il server abbiamo utilizzato la libreria Retrofit implementando un interfaccia per le richieste HTTP.

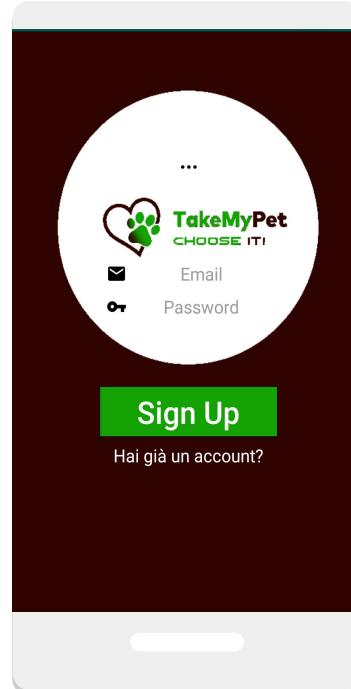
Le chiamate avvengono in modo asincrono e sul dispositivo vengono salvati i dati essenziali, per il funzionamento dell'app, utilizzando SQLite.

Le schermate dell'interfaccia utente vengono gestite dinamicamente utilizzando i fragments, che si intercambiano in un'unica activity.



Applicazione Android

L'utente può registrarsi anche attraverso l'app. Al momento della registrazione può specificare se registrarsi come PetSitter o come Proprietario.

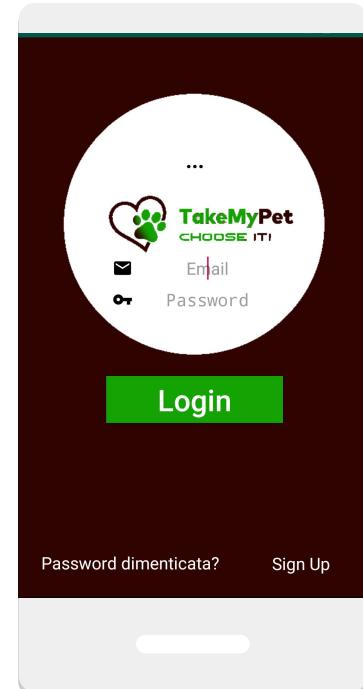
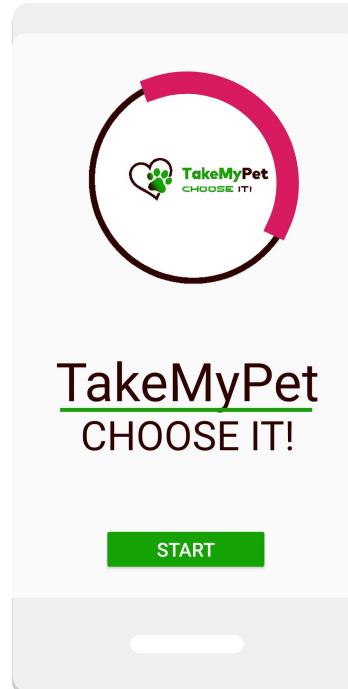




Applicazione Android

L'utente deve loggarsi per utilizzare l'applicazione.
Il funzionamento del login avviene in modo analogo al sito web, ma utilizzando Retrofit.

Il server risponderà con un Json contenente i dati dell'utente, esclusa la password.





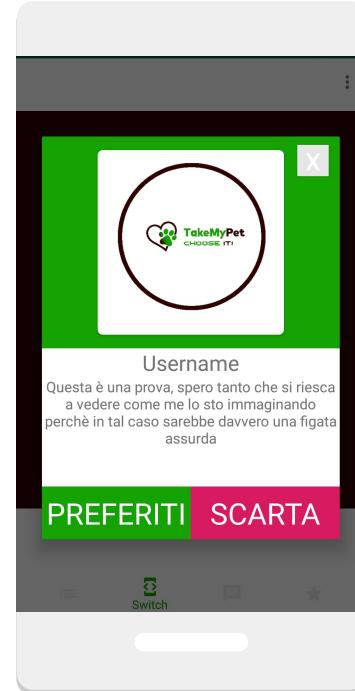
Applicazione Android

Sulla schermata eventi è possibile visualizzare quelli già esistenti ,controllare lo stato dei propri, crearne di nuovi e partecipare a quelli creati dagli altri.



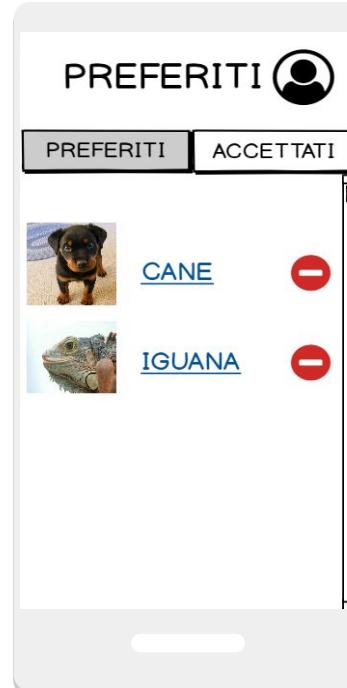
Applicazione Android

Usando lo “swap” il petsitter può trovare l’animale che preferisce e aggiungerlo alla lista dei preferiti. Il Proprietario può creare annunci per i propri animali ed usare la funzione di “swap” per trovare il petsitter più adatto, tra quelli che hanno dato la conferma.



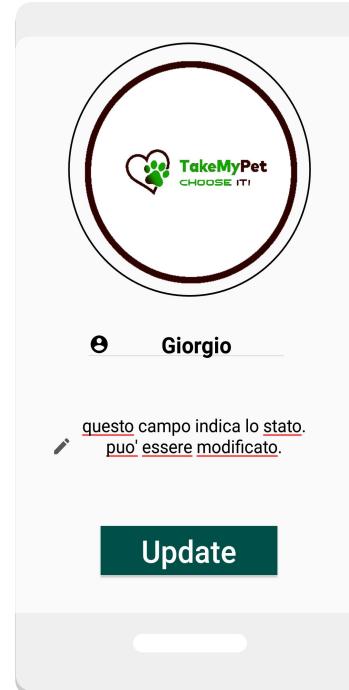
In questa schermata il Proprietario rivede e seleziona il PetSitter tra quelli salvati in preferiti tramite lo swap.

I PetSitters accettano o eliminano gli annunci salvati in preferiti.



Applicazione Android

L'utente inoltre può modificare le sue informazioni personali direttamente dall'applicazione. I dati verranno aggiornati nel database lato server e in quello locale.



Richieste Web

Metodo	Url	End Point	Parametri	Es. JSON
POST	http://takemypet.it	/login	username, password	{"username":"Admin","descrizione":"ciao"}.....}
POST	http://takemypet.it	/signUp	name, surname, username, password, dataNascita, descrizione, latitudine, longitudine, tipo, doppioProfilo	
POST	http://takemypet.it	/immagine	username, immagine	
GET	http://takemypet.it	/immagine	username	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/EventiController		[{"id_evento":1,"organizzatore":"Admin","nomeEvento":"Compleanno"}, {...}]
POST	http://takemypet.it	/EventiController	nomeEvento, descrizione, usernameOrgnizzatore	
PUT	http://takemypet.it	/EventiController	idEventoString, usernamePartecipante	
DELETE	http://takemypet.it	/EventiController	idEventoString	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/eventiUtenteController	usernameProprietario	
GET	http://takemypet.it	/PreferitiProprietarioController	usernameProprietario	[{"id":1,"preferitoDelProprietario":"Jack","petSitterPreferito":"Jim"}, {...}]
POST	http://takemypet.it	/PreferitiProprietarioController	usernameProprietario,usernamePetSitter,idAnuncioStringa	
DELETE	http://takemypet.it	/PreferitiProprietarioController	idPreferitoStringa	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/PreferitiPetSitterController	usernamePetSitter	[{"id":1,"preferitoDelPetSitter":"Frank","annuncio":"5".....}, {"id":2,"preferitoDelPetSitter":"Frank","annuncio":"8".....} ...]
POST	http://takemypet.it	/PreferitiPetSitterController	usernamePetSitter, idAnnuncioString	
DELETE	http://takemypet.it	/PreferitiPetSitterController	idPreferitoString	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/annunciProprietarioController	username	[{"id_annuncio":1,"proprietario":"Frank","nomeAnnuncio":"Ann1".....}, {"id_annuncio":5,"proprietario":"Frank","nomeAnnuncio":"Ann2".....},....]
POST	http://takemypet.it	/annunciProprietarioController	username, nomeAnnuncio, descrizione, latitudine, longitudine, dataCreazione, dataAnnuncio, listaAnimali	
DELETE	http://takemypet.it	/annunciProprietarioController	idAnnuncioString	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/annunciPetSitterController		[{"id_annuncio":1,"proprietario":"Frank","nomeAnnuncio":"Ann1"}, {"id_annuncio":2,"proprietario":"Bob","nomeAnnuncio":"Ann"}, ...]
POST	http://takemypet.it	/annunciPetSitterController	usernamePetSitter, idAnnuncioString	

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/listaUtentiAdmin		[{"id":1,"username":"Admin","descrizione":"ciao"...},{...}]
POST	http://takemypet.it	/listaUtentiAdmin	username, controllo	
GET	http://takemypet.it	/segnalazioniAdmin		[{"id_segnalazione":"3","segnalato":"Jack","segnalatore":"Jim"...},{...}]
GET	http://takemypet.it	/statisticheAdmin		[{"dataRegistrazione":"25/03/2020", "count":"258"}, {...}]

Metodo	Url	End Point	Parametri	Es. JSON
GET	http://takemypet.it	/AnimaleController	usernameProprietario	[{"id_animale":"1","tipo":"cane","razza":"pitbull","nome":"Bob","eta":"3"},{}]
POST	http://takemypet.it	/AnimaleController	usernameProprietario,dettagli,eatString,nome,razza,tip	
DELETE	http://takemypet.it	/AnimaleController	idAnimaleString	
POST	http://takemypet.it	/SbloccoController	username,codiceSblocco	{"username":"Jack","descrizione":"ciao"}.....}

Sviluppi futuri

- Aggiungere la geolocalizzazione
- Aggiungere la funzione di chat
- Aggiungere la funzione doppio profilo
- Aggiungere la funzione di recupero della password

Grazie!

TakeMyPet WEB

TakeMyPet is the perfect app for share your pet with other users completely FREE! If you want to go outside with your pet, you can't leave him alone at home and stay with your pet. NOW IT IS NOT A PROBLEM ANYMORE! Today thanks to TakeMyPet you can do WHATEVER you want WHEREVER you want without thinking about your pet. You can share your pet with other users and take care of your special friends and give they all the love they need completely FREE!

Contact Us

Address
Via Benigno Crepaldi 30
00199 Roma
3319943442
g.henrich@terzizoli.it
d.soriano@terzizoli.it
a.saccoccia@terzizoli.it
l.vaccari@terzizoli.it

What We Do

Our mission is to help those who are unable to dedicate themselves 24 hours to their pets, and to constantly give them the best care possible.

We constantly strive to improve the application so that each user has everything he needs and is able to give all the necessary care to his pets.

Call to Action

TakeMyPet CHOOSE IT!

START