

Note da aggiungere a fine progettazione:

-nel lato: aggiungere un flag sia lato admin sia lato utente la possibilità di filtrare gli animali adottati da quelli non adottati.

-aggiungere un campo nella tabella veterinario "Disponibilità" con si o no dal momento che possiamo permettere di effettuare il login come utente amministrativo ed entrare nel gestionale con determinate limitazioni.

1. Gestione Anagrafica Animali

Obbiettivi:

Permettere all'amministratore del rifugio di:

- Inserire un nuovo animale
- Visualizzare gli animali presenti
- Aggiornare le informazioni di un animale
- Eliminare un animale
- Filtrare/ricercare gli animali

2. Struttura della tabella animali

```
CREATE TABLE animali (  
id_animale INT AUTO_INCREMENT PRIMARY KEY,  
nome VARCHAR(50) NOT NULL,  
specie VARCHAR(50) NOT NULL,  
razza VARCHAR(50),  
sesso ENUM('M', 'F') NOT NULL,  
data_nascita DATE,  
stato_salute ENUM('Ottimo', 'Buono', 'Bisogno cure', 'Critico') NOT NULL,  
data_ingresso DATE NOT NULL,  
immagine_url VARCHAR(255),  
note TEXT,  
adottato BOOLEAN DEFAULT FALSE  
);
```

****NOTE****

- L'età sarà calcolata a runtime in base a `data_nascita`
- lo stato di salute è gestito con un campo `ENUM` semplice
- Le immagini saranno salvate in locale e solo il percorso sarà memorizzato in `immagine_url`

3. Routing front-end(HTML/JS)

Percorso	Contenuto
/	Homepage vetrina
/animali	Lista animali disponibili
/animali/{id}	Scheda animale
/adozione	Form richiesta adozione
/donazioni	Info e possibilità di donare
/contatti	Informazioni rifugio

Percorso	Funzione
/login	Login amministratore
/admin/dashboard	Panoramica gestionale
/admin/animali	Lista + CRUD animali
/admin/animali/aggiungi	Inserimento nuovo animale
/admin/adozioni	Gestione adozioni
/admin/visite	Gestione visite veterinarie
/admin/donazioni	Gestione e storico donazioni

4. Suddivisione delle task in micro-task

Modellazione Database:

Codice	Task	Livello	Tempo stimato
1.1	Creare script <code>animali</code> con tutti i campi principali	2	1h
1.2	Aggiungere ENUM per <code>sex</code> e <code>stato_salute</code>	2	0.5h
1.3	(Opzionale) Aggiungere tabella <code>stato_salute</code> relazionale	3	1.5h
1.4	Verificare script e popolare con 3 animali di esempio	1-2	1h

Creazione Back-End CRUD Animali (Spring Boot)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
2.1	Creare entità <code>Animale.java</code> (annotata)	2-3	1h	Backend	Mappatura oggetto animale

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
	con JPA)				
2.2	Creare <code>AnimaleRepository</code> con Spring Data JPA	2	0.5h	Backend	Accesso ai dati
2.3	Creare <code>AnimaleController</code> con CRUD base (<code>/animali</code>)	3	2h	Backend	Esporre API principali
2.4	Aggiungere endpoint filtro per specie, età, sesso... (<code>@RequestParam</code>)	3	1h	Backend	Filtro e ricerca
2.5	(Extra) Validazione campi con <code>@Valid</code> + gestione messaggi errore	2-3	1h	Backend	Validazione input
2.6	Creare endpoint <code>GET /animali/{id}</code> per recuperare i dati di un animale	2	0.5h	Backend	Supporto al dettaglio/modifica
2.7	Creare endpoint <code>PUT /animali/{id}</code> per aggiornare i dati dell'animale	3	1h	Backend	Supporto alla modifica
2.8	Creare endpoint <code>DELETE /animali/{id}</code> per eliminare un animale	2	0.5h	Backend	Eliminazione record
	EXTRA TASK PER MIGLIORARE COMUNICAZIONE, CALCOLO ETÀ + DINAMICA E SICUREZZA BROWSER				
2.9	Aggiungere DTO <code>AnimaleDTO</code> per ottimizzare la comunicazione API (extra utile)	3	1h	Backend	Pulizia dati in ingresso/uscita
3.0	Implementare calcolo età da <code>data_nascita</code> lato backend	2	0.5h	Backend	Età calcolata dinamicamente

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
3.1	Abilitare CORS su controller se necessario per fetch da frontend	2	0.5h	Backend	Comunicazione FE-BE in locale

Frontend – Form & Visualizzazione Animali

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
3.1	Creare pagina /admin/animali con struttura HTML base	1	1h	Frontend	Struttura base pagina
3.2	Creare form aggiungi animale con tutti i campi	2	1h	Frontend	Inserimento dati animale
3.3	Mostrare elenco animali da API in tabella (fetch())	2-3	1.5h	Frontend	Visualizzazione dinamica
3.4	Aggiungere pulsanti “Modifica” e “Elimina” con chiamate API	2-3	2h	Frontend	Interazione tabella
3.5	Implementare filtro lato frontend	2	1h	Frontend	Ricerca e filtro
3.6	Aggiungere colonna “Azioni” con icone ( , :, )	2	1h	Frontend	UI/UX azioni per riga
3.7	Collegare icona  a pop-up “dettaglio animale”	2	1h	Frontend	Visualizzazione dettagli
3.8	Collegare icona : a pop-up con form precompilato	3	1.5h	Frontend	Modifica dati
3.9	Collegare icona  a modale di conferma per eliminazione	2	0.5h	Frontend	Eliminazione record
4.0	Creare modale HTML per “dettaglio animale”	2	1h	Frontend	Struttura modale visualizzazione
4.1	Creare modale HTML per “modifica animale”	3	1h	Frontend	Struttura modale modifica
4.2	Creare modale HTML per confermare eliminazione	2	0.5h	Frontend	Struttura modale eliminazione

Gestione immagini (semplificata)

Codice	Task	Livello	Tempo stimato
4.1	Creare cartella <code>uploads/</code> per immagini localmente	1	0.5h
4.2	Abilitare accesso statico a immagini in Spring Boot (<code>resources/static</code> o configurazione)	3	1h
4.3	Creare campo <code>immagine</code> nel form (tipo file)	2	1h
4.4	Implementare endpoint per upload immagine e salvataggio path	3	2h
4.5	Visualizzare immagine in tabella e in scheda animale	2	1h

Routing e Login Amministratore

Codice	Task	Livello	Tempo stimato
5.1	Creare tabella <code>admin</code> nel database	2	0.5h
5.2	Creare form di login <code>/login</code> (HTML + JS)	2	1h
5.3	Implementare login lato backend con controllo credenziali	3	2h
5.4	Reindirizzare a <code>/admin/dashboard</code> dopo login	2	1h
5.5	(Extra) Sessione locale semplificata con JS o cookie	3	1.5h

Gestione Adozioni

Tabella Adozioni

```
CREATE TABLE adozioni (  
  id_adozione INT AUTO_INCREMENT PRIMARY KEY,  
  id_animale INT NOT NULL,  
  nome_adottante VARCHAR(50) NOT NULL,  
  cognome_adottante VARCHAR(50) NOT NULL,  
  email VARCHAR(100),  
  telefono VARCHAR(20),  
  indirizzo TEXT,  
  data_adozione DATE NOT NULL,  
  note TEXT,
```

```
FOREIGN KEY (id_animale) REFERENCES animali(id_animale)
ON DELETE CASCADE
ON UPDATE CASCADE
);
```

Tabella prenotazioni_adozione

```
CREATE TABLE prenotazioni_adozione (
```

```
id_prenotazione INT AUTO_INCREMENT PRIMARY KEY,
id_animale INT NOT NULL,
nome VARCHAR(50) NOT NULL,
cognome VARCHAR(50) NOT NULL,
telefono VARCHAR(20),
email VARCHAR(100) NOT NULL,
data_appuntamento DATE NOT NULL,
orario_appuntamento TIME NOT NULL,
note TEXT,
stato ENUM('In attesa', 'Approvata', 'Rifiutata') DEFAULT 'In attesa',
archiviato BOOLEAN DEFAULT FALSE,
FOREIGN KEY (id_animale) REFERENCES animali(id_animale)
ON DELETE CASCADE
ON UPDATE CASCADE
);
```

☒ FRONTEND – Utente visitatore (modulo prenotazione)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
PV1	Aggiungere pulsante "Richiedi visita" nella pagina animale pubblico (/animali/{id})	2	0.5h	Frontend	Inizio processo di prenotazione
PV2	Creare form HTML per prenotazione (nome, cognome, tel, email, data, ora, note)	2	1h	Frontend	Inserimento dati prenotazione
PV3	Validare input lato client	2	0.5h	Frontend	Controlli su email, orari, campi obbl.

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
PV4	Inviare prenotazione tramite <code>fetch()</code> a <code>/prenotazioni</code> (POST)	2	0.5h	Frontend	Comunicazione con backend
PV5	Mostrare messaggio di conferma invio	1	0.5h	Frontend	UX feedback

✅ FRONTEND – Amministratore (gestione prenotazioni)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
PV6	Creare pagina <code>/admin/prenotazioni</code> con tabella delle richieste	2	1h	Frontend	Vista e gestione lato admin
PV7	Mostrare dati: nome utente, animale, data, ora, stato, contatti	2	1h	Frontend	Interfaccia informativa
PV8	Aggiungere filtro per <code>stato</code> (In attesa, Approvata, Rifiutata)	2	0.5h	Frontend	Ordinamento e usabilità
PV9	Aggiungere colonna "Azione" con icona 🗑 per archiviare prenotazione	2	0.5h	Frontend	Pulizia visiva
PV10	Inviare <code>PUT</code> per cambiare stato (opzionale: dropdown o pulsanti)	3	1h	Frontend	Gestione stato prenotazione

✅ BACKEND – Prenotazioni

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
PV11	Creare entità <code>PrenotazioneVisita.java</code> (JPA)	2-3	1h	Backend	Mappatura richiesta
PV12	Creare repository <code>PrenotazioneVisitaRepository</code>	2	0.5h	Backend	Accesso ai dati
PV13	Creare controller <code>PrenotazioneVisitaController</code> con <code>POST /prenotazioni</code>	3	1h	Backend	Ricezione prenotazione

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
PV14	Aggiungere endpoint GET /admin/prenotazioni con filtro per stato e archiviato	3	1h	Backend	Vista amministrativa
PV15	Aggiungere endpoint PUT /admin/prenotazioni/{id} per aggiornare stato	3	0.5h	Backend	Gestione stato lato admin
PV16	Aggiungere campo archiviato BOOLEAN DEFAULT FALSE e endpoint PUT per archiviarlo	2	0.5h	Backend	Archiviazione logica

Gestione Donazioni



1. BACKEND – CRUD Donazioni + Statistiche

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
D1	Creare entità Donazione.java (annotata con JPA)	2-3	1h	Backend	Mappatura tabella donazioni
D2	Creare repository DonazioneRepository con metodi custom SUM(...)	2	0.5h	Backend	Accesso e aggregazione dati
D3	Creare controller DonazioneController	3	1h	Backend	Endpoint POST e GET
D4	Endpoint POST /admin/donazioni per registrazione donazione	3	1h	Backend	Inserimento da admin
D5	Endpoint GET /admin/donazioni per visualizzare elenco donazioni	2	0.5h	Backend	Tabella lato admin
D6	Endpoint GET /donazioni/statistiche per restituire i contatori aggregati	2-3	0.5h	Backend	Supporto contatori pubblici
D7	Validare dinamicamente i campi in base al tipo (Monetaria vs Prodotti)	3	1h	Backend	Controlli logici e consistenza

2. FRONTEND – Pannello Donazioni (Admin)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
D8	Creare pagina <code>/admin/donazioni</code> con tabella delle donazioni	2	1h	Frontend	Visualizzazione lato amministratore
D9	Aggiungere pulsante “Registra Donazione”	1	0.5h	Frontend	Avvio popup di inserimento
D10	Creare form/modale con campi condizionali (tipo: Monetaria/Prodotti)	3	1.5h	Frontend	Inserimento dettagliato
D11	Validare dinamicamente i campi (visibilità e obbligatorietà condizionata)	2-3	1h	Frontend	UX semplificata
D12	Inviare i dati del form a backend con <code>fetch()</code> (POST)	2	0.5h	Frontend	Connessione form ↔ API
D13	Aggiornare la tabella dinamicamente dopo l'invio	2	0.5h	Frontend	Ricaricamento senza refresh completo

3 FRONTEND PUBBLICO – Contatori Donazioni

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
D14	Creare sezione <code>#contatori-donazioni</code> nella home o in <code>/donazioni</code>	1	1h	Frontend	Mostrare dati in forma chiara
D15	Fetch contatori da <code>/donazioni/statistiche</code> e aggiornare DOM	2	0.5h	Frontend	Dinamismo lato utente pubblico
D16	Aggiungere styling CSS semplice e comunicativo ( e )	1	0.5h	Frontend	Accessibilità e design
D17	(Opz.) Testare contatori con dati falsi in locale	1	0.5h	QA	Verifica funzionamento


GESTIONE VISITE VETERINARIE

1. GESTIONE VETERINARI (CRUD ADMIN)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
V1	Creare tabella <code>veterinari</code> (con campo <code>tipo_collaborazione</code>)	2	0.5h	DB	Struttura anagrafica veterinari
V2	Creare entità <code>Veterinario.java</code> (JPA)	2-3	1h	Backend	Mappatura classe veterinario
V3	Creare repository <code>VeterinarioRepository</code>	2	0.5h	Backend	Accesso ai dati
V4	Creare controller <code>VeterinarioController</code> con CRUD (GET/POST/PUT/DELETE)	3	1.5h	Backend	API per gestione veterinari
V5	Creare pagina <code>/admin/veterinari</code> con tabella e form aggiunta/modifica	2	1.5h	Frontend	Interfaccia CRUD veterinari
V6	Validare campi (email unica, telefono unico) nel form	2	0.5h	Frontend	Sicurezza e pulizia input

2. REGISTRAZIONE VISITA (DALL'ANAGRAFICA ANIMALE)

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
V7	Creare tabella <code>visite</code> (già esistente e approvata)	2	0.5h	DB	Collegamento animale ↔ veterinario
V8	Creare entità <code>Visita.java</code> (JPA)	2-3	1h	Backend	Mappatura visita
V9	Creare repository <code>VisitaRepository</code>	2	0.5h	Backend	Accesso ai dati
V10	Creare controller <code>VisitaController</code> con POST/GET/PUT	3	1.5h	Backend	Inserimento e modifica visita

Codice	Descrizione (Task)	Livello	Tempo stimato	Categoria	Funzione principale
V11	Nella tabella animali, aggiungere icona “  visita” nella colonna “Azioni”	2	0.5h	Frontend	Collegamento rapido visita ↔ animale
V12	Mostrare popup con selezione veterinario + data, ora, tipo, urgenza, note	2-3	1h	Frontend	Form dettagliato
V13	Bloccare la creazione della visita se non ci sono veterinari registrati	3	0.5h	Frontend	Controllo logico

3. GESTIONE VISITE – FILTRI, STATO E MONITORAGGIO