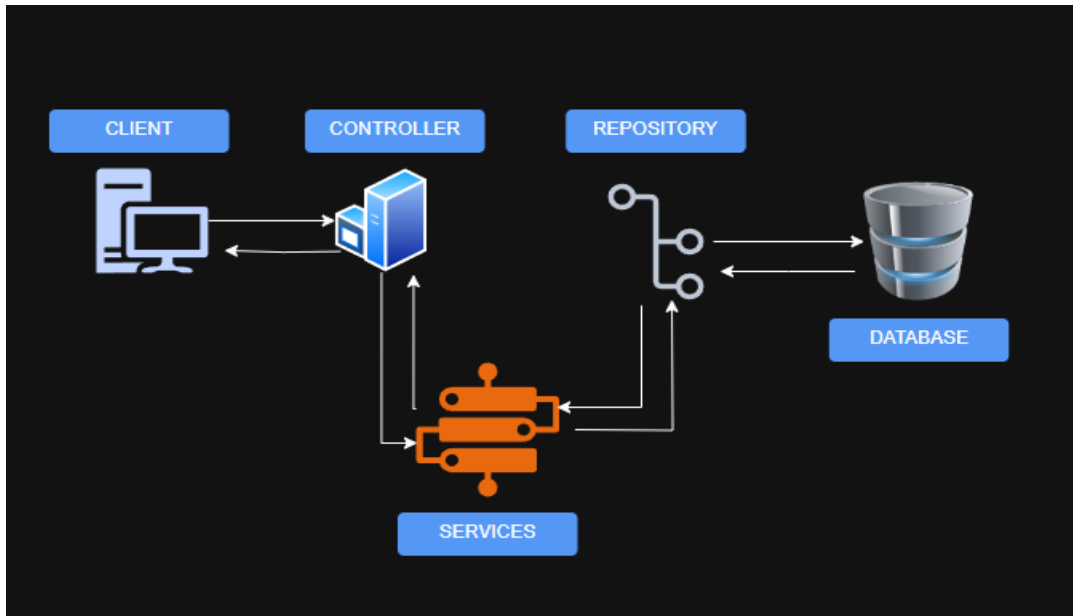


PET ADOPTION PROJECT REPORT

1) Architecture Diagram:



2) ER Diagram / Database Schema

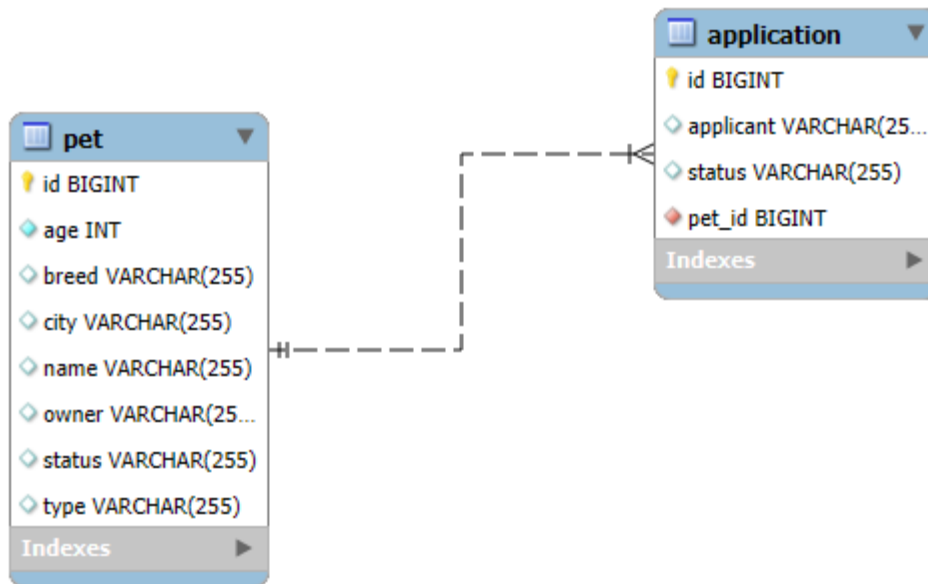


Figure 2 ER Diagram

2.1 Table Schema

Entity	Field	Type	Constraints	Notes
Pet	Id	BIGINT	PK, AUTO_INCREMENT, NOT NULL	Primary key
Pet	Age	INT	NOT NULL	Age of the pet
Pet	breed	VARCHAR(255)		Breed of the pet
Pet	City	VARCHAR(255)		Pet's city
Pet	name	VARCHAR(255)		Pet's name
Pet	owner	VARCHAR(255)		Owner's name
Pet	status	VARCHAR(255)		Availability status
Pet	Type	VARCHAR(255)		Type of pet (Dog, Cat, etc.)
Application	Id	BIGINT	PK, AUTO_INCREMENT, NOT NULL	Primary key
Application	Applicant	VARCHAR(255)	NOT NULL	Name of applicant
Application	status	VARCHAR(255)		Application status
Application	pet_id	BIGINT	NOT NULL, FK → Pet.id	References the pet

3) Functional Requirements

3.1 Controllers and Their Functions

PetController

Controller	Function Name	Input	Output	Validation / Notes
PetController	list	—	List<PetDTO>	Optional filters: type, city, status
PetController	create	PetDTO	PetDTO	Validate required fields: name, age
PetController	update	PetDTO	PetDTO	ID must exist; validate status values
PetController	view	id (PathVariable)	PetDTO	Return 404 if not found
PetController	delete	id (PathVariable)	{ success: true }	Return true/false

ApplicationController

Controller	Function Name	Input	Output	Validation / Notes
ApplicationController	list	—	List<ApplicationDTO>	Optional filters: status, pet_id
ApplicationController	create	ApplicationDTO	ApplicationDTO	Validate required fields: applicant, pet_id
ApplicationController	update	ApplicationDTO	ApplicationDTO	ID must exist; check valid status
ApplicationController	view	id (PathVariable)	ApplicationDTO	Return 404 if not found
ApplicationController	delete	id (PathVariable)	{ success: true }	Return true/false

3.2 Methods, URL, Description

Pet API

HTTP Method	URL	Purpose	Request Body (JSON)	Response (JSON)	Status Codes
GET	/api/pets	List all pets	—	Array<PetDTO>	200
POST	/api/pets	Create a pet	{ name, age, breed, ... }	PetDTO	201 / 400
PUT	/api/pets	Update a pet	{ id, name, age, ... }	PetDTO	200 / 400 / 404
GET	/api/pets/{id}	Get a pet by ID	—	PetDTO	200 / 404
DELETE	/api/pets/{id}	Delete a pet	—	{ success: true }	200 / 404

Application API

HTTP Method	URL	Purpose	Request Body (JSON)	Response (JSON)	Status Codes
GET	/api/applications	List all applications	—	Array<ApplicationDTO>	200
POST	/api/applications	Create an application	{ applicant, pet_id, status }	ApplicationDTO	201 / 400
PUT	/api/applications	Update an application	{ id, applicant, pet_id, status }	ApplicationDTO	200 / 400 / 404
GET	/api/applications/{id}	Get an application by ID	—	ApplicationDTO	200 / 404
DELETE	/api/applications/{id}	Delete an application	—	{ success: true }	200 / 404

4) UI Wireframes / Mockups

4.1 List Screen (Index)

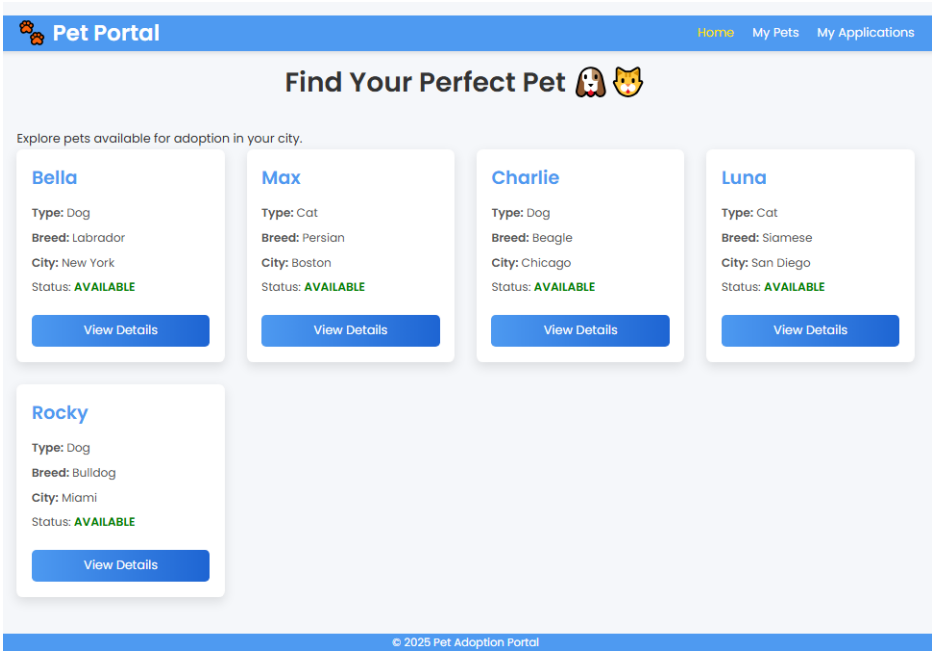


Figure 1 Pet List

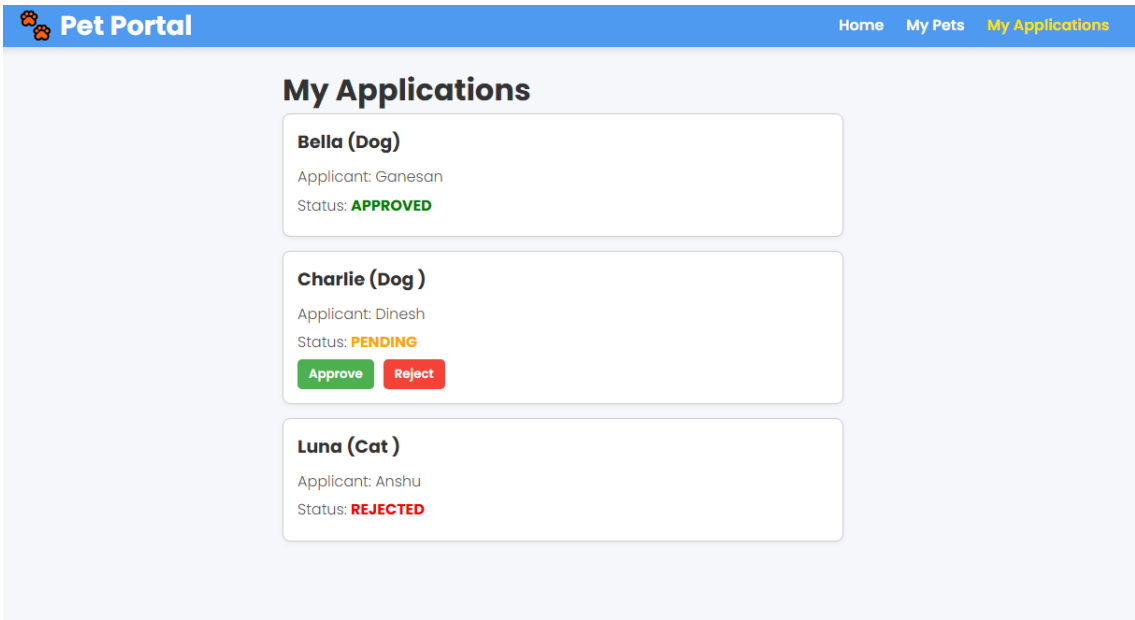



Figure 2 Application List

4.2 Create / Edit Form

 **Pet Portal**

HomeMy PetsMy Applications

Manage My Pets

Add Pet


Bella
Dog - Labrador
City: New York
Status: **ADOPTED**
Delete

Max
Cat - Persian
City: Boston
Status: **AVAILABLE**
Delete

Charlie
Dog - Beagle
City: Chicago
Status: **AVAILABLE**
Delete

Luna
Cat - Siamese
City: San Diego
Status: **AVAILABLE**
Delete

Figure 3 Add Pet

 **Pet Portal**

[Home](#) [My Pets](#) [My Applications](#)

Bella
Type: Dog
Breed: Labrador
Age: 3
City: New York
Status: AVAILABLE

Apply for Adoption

Figure 4 Application for adoption