Title of the Project: Dog Adoption Website Across the United States

Name of the students: Marissa Zhang (yzhang2587@wisc.edu), Yining Wang (ywang2739@wisc.edu), Osama Kheshaifaty (kheshaifaty@wisc.edu)

I. Introduction:

1.1 Motivation:

We chose to work on a dog adoption website because dogs play significant roles in human society, offering companionship, emotional support, and helping people connect with others. We recognized the increasing popularity of dog adoption as a compassionate and responsible way to bring a new pet into a family. However, the absence of a centralized platform for searching and adopting dogs nationwide made the process inefficient and time-consuming. To address this issue, we decided to develop a comprehensive website that contains a database of all adoptable dogs across the USA, making it easier for potential adopters to find their ideal pet.

1.2 Application Description:

Our application is a web-based dog adoption platform that provides users with a user-friendly interface to search and adopt dogs from various animal shelters and pet stores across the United States. The application allows users to search for dogs based on specific criteria such as age, size, breed, location (zipcode), and vaccination status. It also facilitates shelter owners in managing their dog adoption database by allowing them to add, delete, and modify dog records as needed.

1.3 Organization of the Report:

In this report, we will present the details of our project, including the system architecture, dataset description, final ER diagram, relational model, prototype implementation, and evaluation of the application. Each team member has contributed to different aspects of the project, and we will outline their respective contributions.

II. Our Implementation:

2.1 Description of the System Architecture:

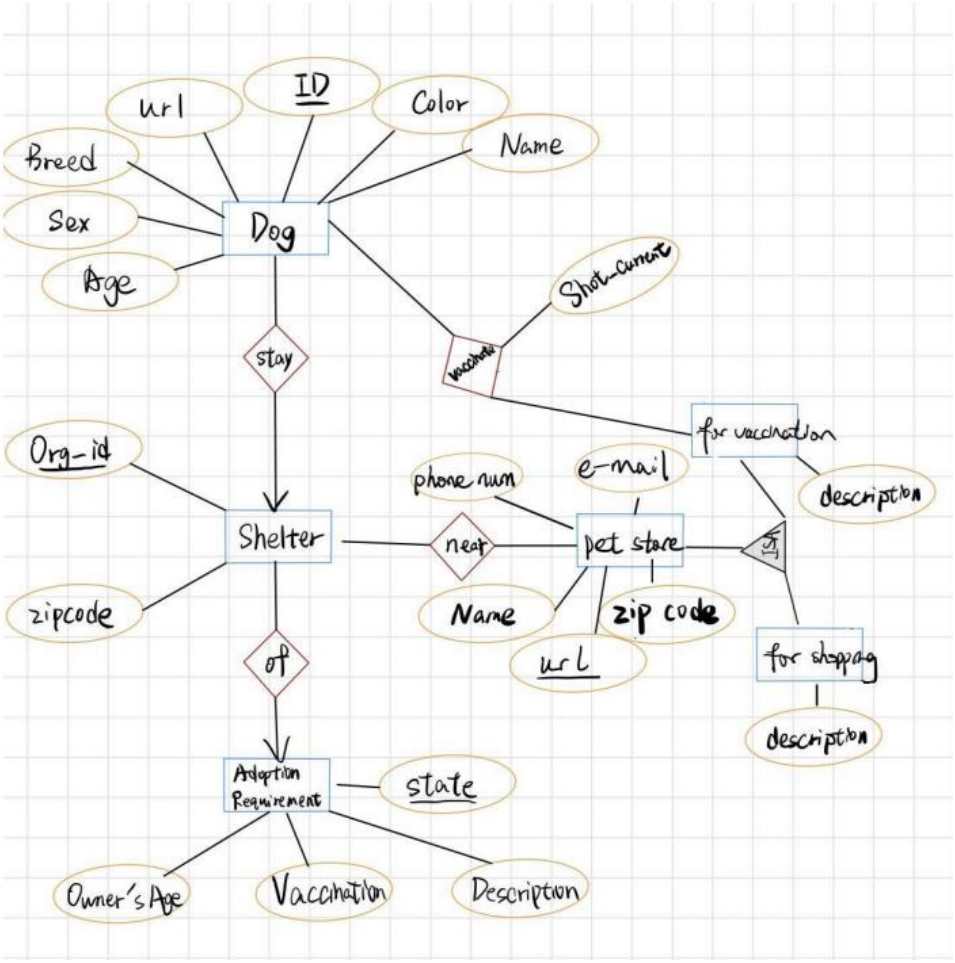
The system architecture of our web-based dog adoption website consists of a frontend and a backend. The frontend is built using HTML to create an intuitive user interface for potential adopters to search for dogs based on their preferences. The backend is developed using Python, SQLite , which handles user requests, interacts with the database, and implements the B+ tree data structure to efficiently manage the dog adoption records.

2.2 Description of the Dataset:

Our dataset is a combination of multiple sources. The main dog dataset is obtained from Kaggle, which provides a wide range of attributes, including dog information and shelter descriptions. Additionally, we filter dog clinics' information from the Yelp Open Dataset, which contains details about businesses, reviews, and user data. We also crawl dog adoption requirements from various US government dog vaccination websites. The dataset comprises more than 50,000 records, ensuring its complexity to facilitate the creation of a comprehensive ER diagram.

2.3 ER Diagram (Final Version):

Below is the latest ER diagram, showing the entities, attributes, and relationships between them:



2.4 Relational Model (Final Version):

Below is the finalized relational schema, specifying all keys and designating one as the primary key for each table:

Dogs (ID, url, Name, Age, Sex, primary_breed, Color, Org-ID,shot_current,description, contact_city)

Primary key: Dog-ID

Secondary Key: url

Foreign Key: Shelter-ID

Shelters (org_id, state, city)

Primary key: Shelter-ID

Adoption Requirements (State, requirement, age)

Primary Key: State

Secondary key: URL

Pet_care (for shopping pet accessories) (business_id, Name,postal_code, categories,state, stars)

Primary Key: business_id

Secondary Key: (Name, postal_code)

clinics (for vaccination) (business_id, Name,postal_code, categories,state, stars)

Primary Key: business_id

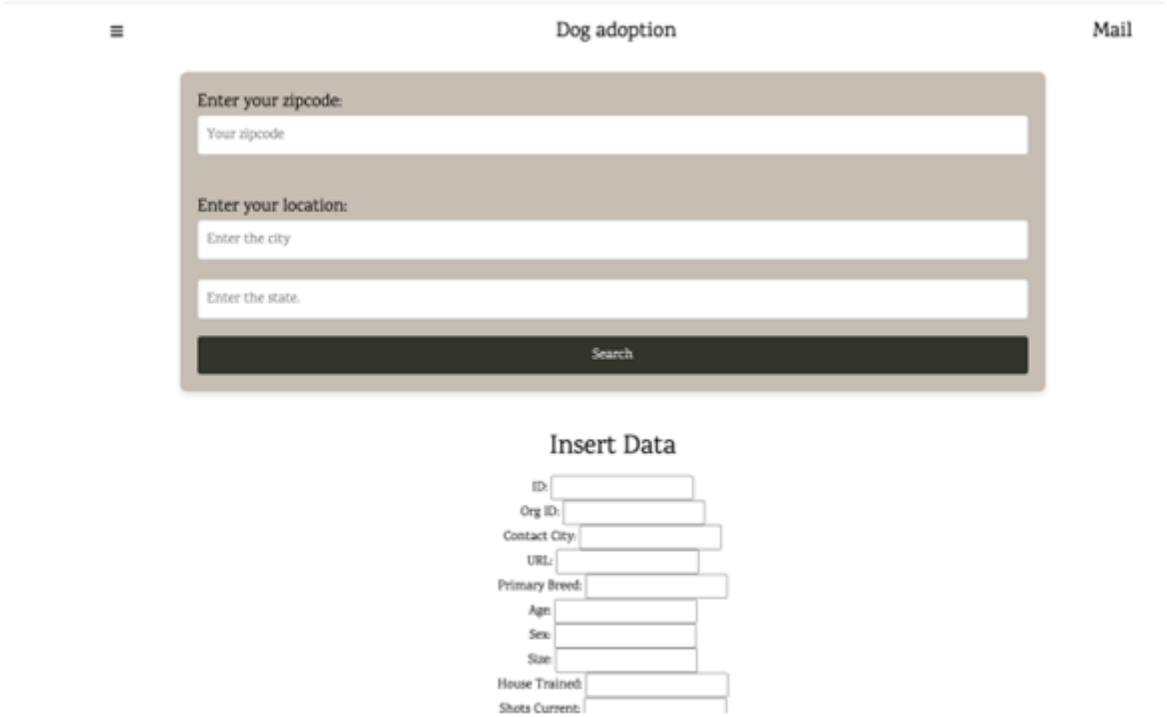
Secondary Key: (Name, postal_code)

2.5 Implementation: Description of the Prototype:

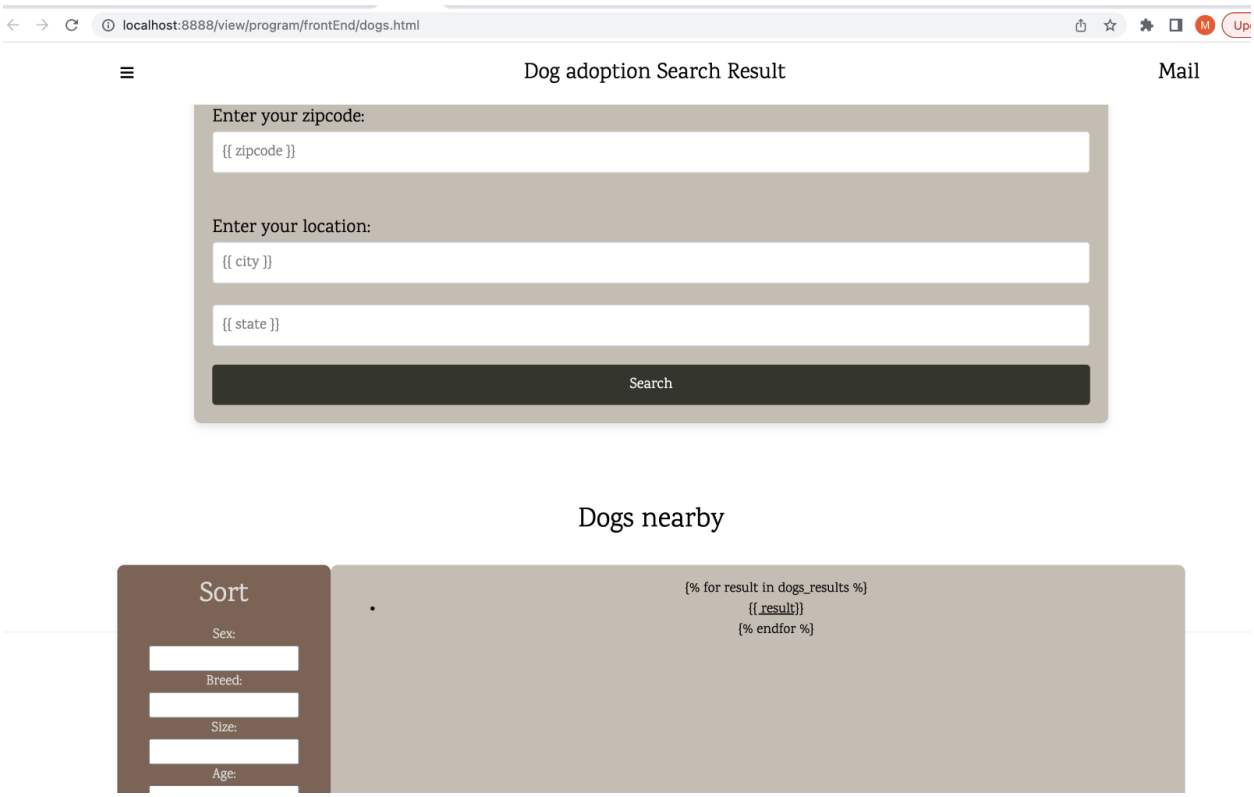
The prototype of our dog adoption website has been developed using HTML, Python, and SQLite. Python serves as the backbone of both the frontend and backend, managing the database structure and user interactions. Python is used to extract relevant information from the Yelp dataset, specifically filtering dog clinics' details. SQLite is employed as the database

management system, enabling efficient editing, deletion, and management of the adoption records.

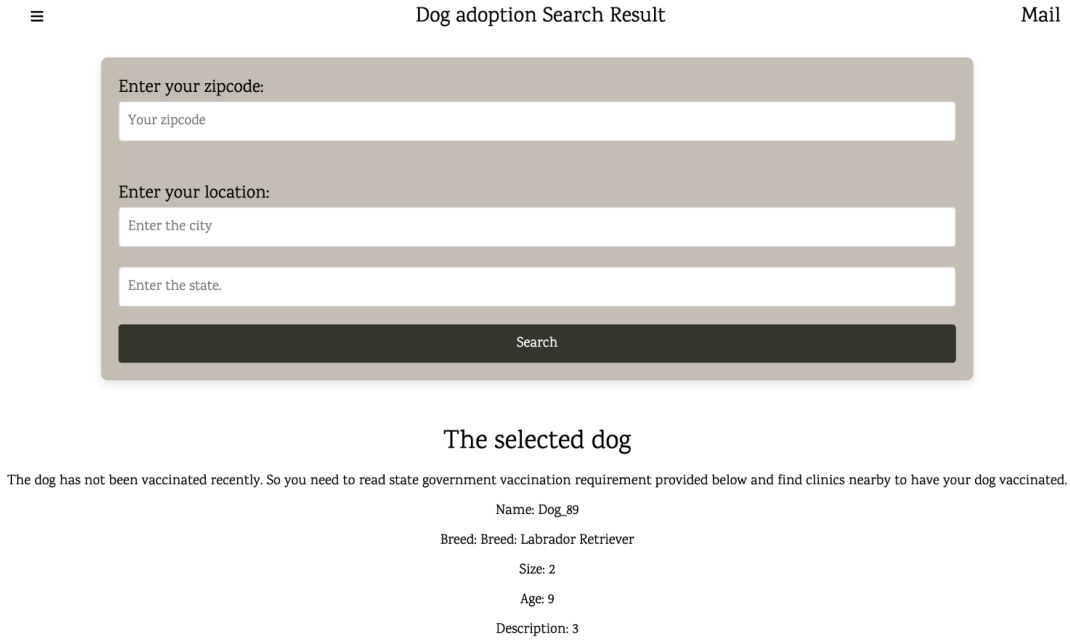
- Interface:



- The home page contains a search bar and insert place. The user can enter the zip code, city, and state information, and we can recommend the shelters and dogs nearby. Also, the user can use the form below to insert data (dog ID, Contact City, URL…..) into the database like queries 7,8,9. So the database can keep updated.



- The search result page shows the zip code, city, and state the user entered before and the dog's result. The dog nearby result shows the dogs' information by using the sqllite like query 1, and the result can be sorted by sex, breed, size, age like query 2. Each dog is clickable.



≡ Dog adoption Search Result Mail

Results from dog adoption state government vaccination requirement

Please follow the state government vaccination requirement to get a smooth dog adoption experience.

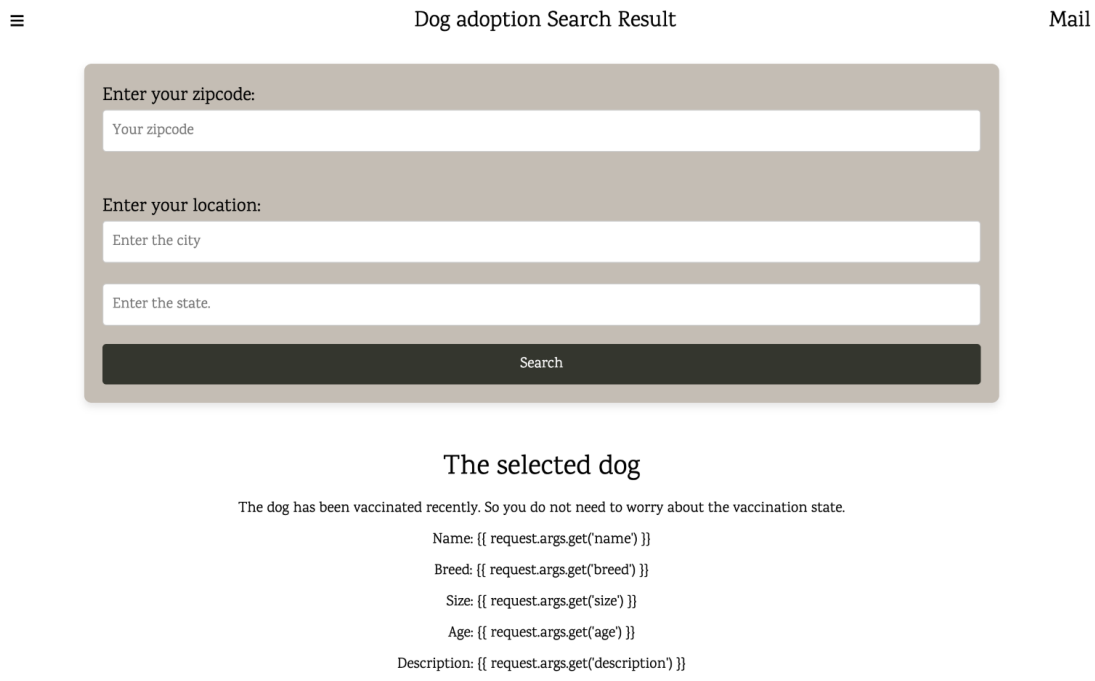
Clinics nearby

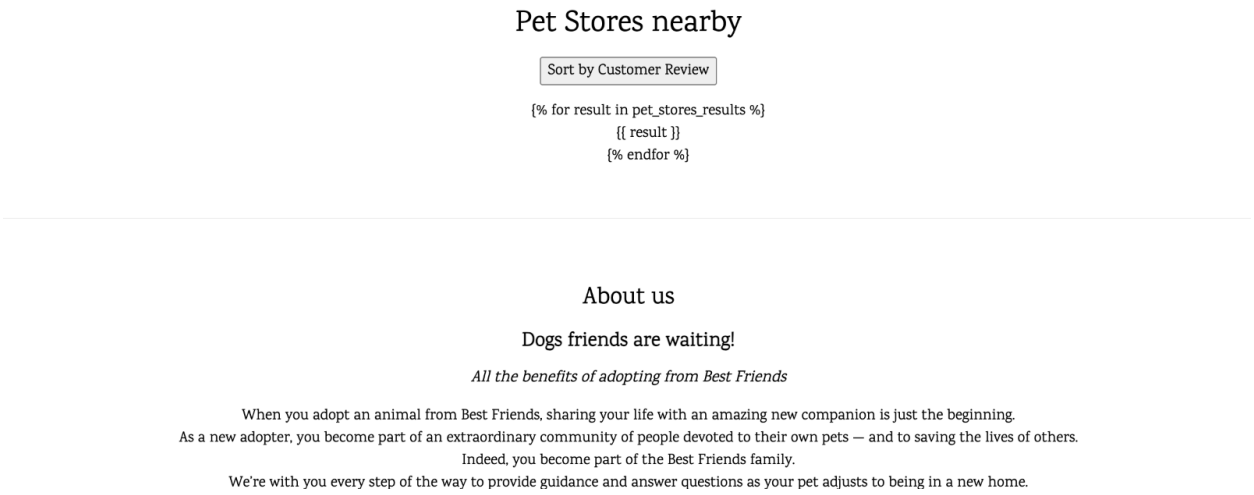
Sort by Customer Review

Pet Stores nearby

Sort by Customer Review

- If the selected dog is not shot vaccination recently, we will show the vaccination requirement for the state, clinics, and pet stores nearby. The data is collected like query 4,5,6. The details of the dog information will show again. By clicking the button “sort by customer review”, the result can be shown from the default order to the rating order.





- If the selected dog is shot vaccination recently, we will only show the details of the dog information and pet stores nearby like query 6. By clicking the button “sort by customer review”, the result can be shown from the default order to the rating order.

2.6 Evaluation: Description of Testing Approach:

To test our application's functionality and accuracy, we have designed multiple testing scenarios and SQL queries that represent common user interactions and adoption use cases. We run these scenarios through the application, examining the returned results to ensure that the information retrieved from the database aligns with the user's search criteria. We have conducted several rounds of testing to identify and fix any potential bugs or discrepancies in the application's performance.

1. test storing data

- test adding new dog or shelter to the table
- test if a dog is added to the shelter, but the org_id is new (it should automatically add the corresponding org_id to shelter table)
- test if a duplicated dog id is added, error message
- test if a wrong format of entry is added, error message
- test adding a new state requirement
- test if a dog info could be removed from the table(if adopted)

2. test front end

- For the frond end, test if a non-vaccinated dog is selected, it will return the nearby dog clinics
- test if a query could be sent correctly to the backend(if the data entered by user could be compact correctly)
- test if a wrong format, or value of data is entered
- test if the displayed string is correct
- test if no desired entry is found

3. test back end

- based on the user input, if a valid query could be established
- test if the returned result is successfully sorted,
- test if the string passed to front end is correct,
- test if the stored data can be selected
- test error input

III. Conclusion:

What We Learned from the Project:

Throughout this project, we have gained valuable insights into the importance of developing efficient database systems, especially when handling large datasets. Working with complex data and designing a user-friendly application has allowed us to apply our database knowledge and practical skills to real-world scenarios.

Relevance of Database Knowledge:

The concepts and techniques we learned in the database course have been instrumental in designing and implementing the database schema and B+ tree structure. Understanding ER modeling, normalization, and functional dependencies have guided us in maintaining data integrity and ensuring efficient querying and management.

Database Relevant Issues Encountered:

During the project, we encountered challenges related to database indexing, data modeling, and query optimization. Addressing these issues required careful consideration of database design choices, such as selecting appropriate primary and secondary keys and implementing efficient data retrieval strategies.

IV. References:

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press. Chapter 18: B-Trees.

2. Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). Database Systems: The Complete Book (2nd ed.). Pearson. Chapter 10: B-Trees and B+ Trees.

3. Elmasri, R., & Navathe, S. B. (2015). Fundamentals of Database Systems (7th ed.). Pearson. Chapter 4: Entity-Relationship (ER) Modeling.