ABESIT COLLEGE OF ENGINEERING

CRC-Training

AdoptEase

Summer Internship
Web Development using JSP, Servlet and MySQL Training

Submitted

To

CRC-Training

By

Suraj Prajapati
Roll Number: 2102900100153

Under the guidance of
Mr. Vijay Kumar / Mr. Gaurav Kansal

TABLE OF CONTENTS

ABESIT
COLLEGE OF ENGINEERING

CRC-Training

CERTIFICATE

This is to certify that the Project Report entitled **“AdoptEase”** which is submitted by **Suraj Prajapati** and in partial fulfillment of the requirement for the “**Web Development using JSP, Servlet and MySQL**” in the Department of CRC-Training of ABES Institute of Technology, is a record of the candidate own work carried out by him under my/our supervision.

Mr. Gaurav Kansal

Mr. Vijay Kumar

Date:

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the "**Web Development using JSP, Servlet and MySQL**" *undertaken during B. Tech, 3rd Year. We owe a special debt of gratitude to Mr. Vijay Kumar and Mr. Gaurav Kansal for his constant support and guidance throughout the course of our work.His constant motivation has been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.*

We also take the opportunity to acknowledge the contribution of team members of CRC-Training and HOD Name (HOD) for their full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the motivation of the Department Of Computer Science, ABES Institute of Technology to provide us the opportunity to undergo training at CRC-Training.

Signature:

Name: Suraj Prajapati

Roll No.:

2102900100153

Date:

ABSTRACT

The "**AdoptEase**" project is a comprehensive web application designed to streamline the pet adoption process and enhance the experience for both prospective pet owners and animal shelters. This project aims to bridge the gap between pet seekers and available pets through an intuitive and user-friendly online platform.

The system is built on a Model-View-Controller (MVC) architecture, utilizing Java Servlets for handling business logic, Java Server Pages (JSP) for dynamic content rendering, and JavaBeans for encapsulating data. The underlying data management is handled by a MySQL database, which stores information about pets, users, and adoption requests. Apache Tomcat serves as the web server for deploying and managing the application.

Key features of the platform include a searchable database of pets, user account management, and an adoption application process. Users can browse through a list of available pets, filter search results based on various criteria, and submit adoption requests. The system also provides a secure login for users and maintains a record of adoption activities.

Future enhancements for the project include integrating advanced technologies such as artificial intelligence for pet matching, developing mobile and progressive web applications, and expanding community engagement through educational resources and partnerships with animal welfare organizations. The project aims to improve the efficiency of pet adoption processes, support animal shelters, and ultimately contribute to better pet care and responsible pet ownership.

This abstract summarizes the project's goals, technology stack, and key features, providing an overview of its purpose and potential impact on the pet adoption community.

LIST OF TABLES

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

The "AdoptEase" project is dedicated to fostering a compassionate and responsible approach to pet ownership. With countless animals in need of loving homes, this initiative aims to connect potential pet owners with adoptable pets, while also providing comprehensive resources and guidance on pet care. By promoting adoption over purchasing, we hope to reduce the number of animals in shelters and ensure that every pet finds a safe, nurturing environment. This project emphasizes the importance of lifelong commitment to pet care, offering support for both new and experienced pet owners to create happy, healthy relationships with their furry companions. In order for readers to trust the writer, the introduction must be well written with few errors. In order to keep readers reading, the writer needs to catch the attention of the reader and write in an interesting way. The unique language enhancement feature may suggest words to strengthen the writing. Strong writing may hold readers' attention.

Problem Introduction

1.1. Problem Definition: The "AdoptEase" project addresses several critical issues in the current pet adoption process. The existing system of pet adoption, typically managed through local animal shelters or rescue organizations, faces numerous challenges that hinder the efficient placement of pets into loving homes. These challenges include limited visibility of adoptable pets, cumbersome and inconsistent adoption processes, and a lack of centralized information and resources for prospective pet owners.

- Limited Visibility and Accessibility
    - Geographical Limitations: Many prospective pet adopters are limited to visiting local shelters, which may not have the variety of pets they are looking for. This limitation reduces the chances of pets being adopted quickly and forces shelters to manage overcrowding.
    - Inefficient Communication Channels: Potential adopters often rely on phone calls or in-person visits to inquire about available pets, which can be time-consuming and inefficient. This lack of a streamlined, centralized platform to view and inquire about pets delays the adoption process.
- Cumbersome Adoption Process
    - Inconsistent Procedures: The adoption process varies significantly between shelters and rescue organizations, leading to confusion and frustration for prospective adopters. This inconsistency can include differences in application requirements, processing times, and adoption fees.
    - Manual Paperwork: Many shelters still rely on manual paperwork for processing adoption applications, which is not only time-consuming but also prone to errors and data loss. This manual process can also deter potential adopters who prefer a more streamlined, online application process.
- Lack of Resources and Support for Adopters
    - Insufficient Information: Prospective pet owners often do not have easy access to comprehensive information about the responsibilities of pet ownership, including care guidelines, training resources, and health considerations. This gap can lead to mismatched expectations and, in some cases, the return of adopted pets.

1.2. Motivation:

The motivation behind the "AdoptEase" project stems from a deep commitment to improving the lives of animals in need and the desire to enhance the overall pet adoption experience for both adopters and animal shelters. The project is driven by several key factors:

- Increasing Pet Adoption Rates

  Millions of animals end up in shelters every year, with many facing the risk of euthanasia due to overcrowding and limited resources. A significant motivation for this project is to increase adoption rates by making it easier for prospective pet owners to find and adopt pets that match their lifestyle and preferences. By providing a centralized, online platform, the project seeks to connect more people with pets in need of homes, ultimately reducing the number of animals in shelters.

- Reducing the Burden on Animal Shelters

  Animal shelters and rescue organizations are often overwhelmed by the sheer number of animals they care for, coupled with limited funding, staffing, and space. The "AdoptEase" project aims to alleviate some of this burden by streamlining the adoption process, thereby enabling shelters to focus more on the care and well-being of the animals. By automating and standardizing certain aspects of the adoption process, the project also helps shelters operate more efficiently and effectively.

- Ensuring Responsible Pet Ownership

  One of the primary motivations is to promote responsible pet ownership. Adopting a pet is a lifelong commitment, and it's crucial that adopters are fully informed about the responsibilities involved. The project aims to provide comprehensive resources, including care guides, training tips, and access to expert advice, to ensure that adopters are well-prepared to take on the role of a pet owner. This approach not only benefits the adopters but also contributes to the long-term well-being of the pets.

- Addressing Mismatched Adoptions

  Mismatched adoptions, where the pet’s needs or behavior do not align with the adopter’s expectations or lifestyle, often result in the pet being returned to the shelter. This is stressful for both the pet and the shelter and can discourage potential adopters. The motivation to reduce these mismatches is a key driver for the project, which aims to offer better matching tools and post-adoption support to ensure that more pets stay in their new homes.

- Leveraging Technology for Social Good

  In an increasingly digital world, technology can play a significant role in solving societal challenges. The "AdoptEase" project is motivated by the belief that technology can be harnessed to create a positive impact on animal welfare. By developing a user-friendly, accessible platform, the project leverages technology to bring about meaningful change in the way pet adoptions are managed and how communities engage with animal welfare.

- Fostering Community and Awareness

  Beyond the individual adoption process, the project is motivated by the desire to foster a community of responsible pet owners and advocates for animal welfare. By providing educational content, forums for discussion, and opportunities for users to connect with local shelters and other pet owners, the project seeks to raise awareness about the importance of adoption and animal care, creating a ripple effect that benefits the broader community.

1.3. Objective of the Project:

The primary objective of the "AdoptEase" project is to create a comprehensive, user-friendly web platform that facilitates the pet adoption process while supporting both prospective pet owners and animal shelters. The project aims to address key challenges in the current adoption system and promote responsible pet ownership. The specific objectives include:

- Enhancing Visibility and Accessibility of Adoptable Pets:

  To develop a centralized, searchable database where users can easily browse and find pets available for adoption from various shelters and rescue organizations. The platform aims to increase the visibility of pets, especially those that are harder to place, by providing detailed profiles and filtering options to match users with pets that fit their preferences and lifestyle.

- Streamlining the Adoption Process:

  To standardize and simplify the adoption process by creating a digital application system that reduces paperwork, minimizes errors, and shortens the time required to process adoption requests. This system will include features such as online forms, automated notifications, and a tracking mechanism for both adopters and shelters.

- Promoting Responsible Pet Ownership:

  To provide prospective adopters with access to comprehensive resources on pet care, training, and health, ensuring they are well-prepared for the responsibilities of pet ownership. The platform will offer educational materials, expert advice, and community support to help adopters make informed decisions and provide the best possible care for their pets.

- Supporting Animal Shelters and Rescue Organizations:

  To assist shelters in managing their operations more efficiently by offering tools to track adoption applications, manage pet profiles, and communicate with prospective adopters. The project aims to reduce the administrative burden on shelters, allowing them to focus more on the well-being of the animals in their care.

- Reducing Mismatches and Return Rates:

  To minimize the number of pets returned to shelters by improving the matching process between pets and adopters. The platform will include features such as detailed pet profiles, adopter questionnaires, and post-adoption support to ensure that the pets are placed in suitable homes where they are likely to thrive.

- Leveraging Technology for a Positive Impact:

  To utilize modern web technologies and a scalable architecture to create a robust, secure, and reliable platform that can be accessed by users across different devices and regions. The project aims to harness the power of technology to make the adoption process more efficient, accessible, and impactful.

- Fostering Community Engagement and Awareness:

  To build a supportive online community where pet owners, potential adopters, and animal welfare advocates can connect, share experiences, and learn from one another. The platform will also promote awareness of adoption events, volunteer opportunities, and other ways users can contribute to animal welfare.

1.4. **Scope of the Project:**

The "AdoptEase" project is designed to address various aspects of the pet adoption process, providing a comprehensive solution that benefits both prospective pet owners and animal shelters. The scope of the project encompasses several key areas:

- Development of a Centralized Platform
  - Pet Listings: The platform will feature a centralized, searchable database of adoptable pets from various shelters and rescue organizations. Each pet will have a detailed profile, including information on breed, age, health status, temperament, and special needs. Users can search and filter listings based on their preferences.
  - User Accounts: The platform will support user registration, allowing users to create accounts to save their favorite pets, track their adoption applications, and receive personalized recommendations.
  - Shelter and Rescue Organization Accounts: Animal shelters and rescue organizations will be able to create and manage their profiles, upload pet listings, track adoption applications, and communicate with prospective adopters through the platform.
- Streamlined Adoption Process
  - Online Application System: The project will implement a standardized, digital application process that allows users to submit adoption applications online. This system will include features for collecting necessary information, verifying user credentials, and automating parts of the review process.
  - Application Tracking: Prospective adopters will be able to track the status of their applications in real-time, receive updates, and communicate with shelters through the platform.
- Educational and Support Resources
  - Pet Care Resources: The platform will provide a comprehensive library of resources related to pet care, training, and health. These resources will be available to all users, helping them prepare for pet ownership and ensuring that they are equipped with the knowledge needed to care for their adopted pets.
  - Post-Adoption Support: AdoptEase will offer ongoing support for adopters, including access to expert advice, veterinary services, and community forums where they can connect with other pet owners.
- Community Engagement
  - Forums and Discussions: The platform will include community features such as forums and discussion boards where users can share experiences, ask questions, and support each other throughout the adoption process.
  - Events and Volunteering Opportunities: AdoptEase will promote adoption events, volunteer opportunities, and other community activities to encourage user engagement and support for local shelters and rescue organizations.
- Data Management and Analytics
  - Data Collection and Reporting: The platform will collect data on adoption trends, user preferences, and shelter needs. This data will be used to generate reports that can help shelters optimize their operations and identify areas for improvement.
  - Privacy and Security: The project will prioritize the security and privacy of user data, implementing robust measures to protect sensitive information and ensure compliance with relevant regulations.

1.5. **Need of Work:**

The need for the "AdoptEase" project arises from the challenges and inefficiencies in the current pet adoption system, which often hinder the process of finding suitable homes for animals in shelters. The project is essential for several reasons:

- Addressing Overcrowding in Animal Shelters

  Many animal shelters are overwhelmed with the number of pets in their care, leading to overcrowding and, in some cases, the unfortunate necessity of euthanasia. The "AdoptEase" platform aims to increase the visibility of adoptable pets, helping them find homes more quickly and reducing the strain on shelters. By streamlining the adoption process, the project will facilitate quicker placements and improve the overall adoption rate.

- Enhancing Accessibility and Convenience for Adopters

  The traditional pet adoption process often requires prospective adopters to visit multiple shelters, navigate inconsistent application procedures, and deal with manual paperwork. This can be time-consuming and discouraging. The "AdoptEase" platform will provide a centralized, online solution where users can easily browse available pets, complete adoption applications, and communicate with shelters, making the adoption process more accessible and user-friendly.

- Promoting Responsible Pet Ownership

  Adopting a pet is a significant commitment that requires knowledge and preparation. Many pet adoptions fail because adopters are not fully aware of the responsibilities involved, leading to high return rates. "AdoptEase" will offer educational resources, support tools, and post-adoption guidance to ensure that adopters are well-prepared and capable of providing long-term care for their pets. This proactive approach will help reduce the number of pets being returned to shelters.

- Reducing Mismatches Between Pets and Adopters

  One of the critical issues in the current adoption process is the mismatch between pets and adopters, often due to a lack of detailed information or proper guidance during the selection process. "AdoptEase" will incorporate matching tools, detailed pet profiles, and adopter questionnaires to better align the needs and characteristics of pets with the preferences and lifestyles of adopters. This will increase the likelihood of successful, long-term adoptions.

- Supporting Animal Shelters and Rescue Organizations

  Shelters and rescue organizations often operate with limited resources, making it challenging to manage the administrative tasks associated with adoptions. The "AdoptEase" platform will provide tools to automate and streamline these tasks, freeing up shelter staff to focus on the care and well-being of the animals. Additionally, the platform will offer analytics and reporting features to help shelters optimize their operations and make data-driven decisions.

- Leveraging Technology for Animal Welfare

  In an increasingly digital world, the lack of a robust online platform dedicated to pet adoption represents a missed opportunity. "AdoptEase" seeks to fill this gap by leveraging modern web technologies to create a scalable, secure, and effective solution that can adapt to the evolving needs of adopters and shelters. The project will demonstrate how technology can be harnessed to address critical social issues, such as animal welfare.

- Fostering Community Engagement

  Building a community around pet adoption and care is crucial for raising awareness and encouraging more people to adopt. "AdoptEase" will foster a supportive online community where adopters, pet owners, and animal advocates can share experiences, access resources, and participate in events. This community engagement is essential for promoting a culture of responsible pet ownership and expanding the reach of adoption efforts.

1.6. Technological Background:

1.6.1. MySQL (Database)

MySQL is an open-source relational database management system (RDBMS) that uses Structured Query Language (SQL) for managing and manipulating databases. It is one of the most popular databases in the world, widely used for web applications due to its reliability, ease of use, and support for various programming languages and platforms.

MySQL operates on the client-server model and provides a robust and scalable environment for data storage and retrieval. MySQL also offers features like replication, clustering, and support for ACID (Atomicity, Consistency, Isolation, Durability) properties, which are essential for ensuring data integrity and reliability.

MySQL Command Examples:

- Data Definition Language (DDL): DDL commands in MySQL are used to define and manage the structure of the database objects like tables, indexes, and views. Here are some common DDL commands:
  - Create Database: CREATE DATABASE PetAdoption;
  - Table create: CREATE TABLE Pets (PetID INT PRIMARY KEY AUTO_INCREMENT, PetName VARCHAR (50), PetType VARCHAR (50), Age INT, AdoptionStatus VARCHAR (20);
  - Drop Table: DROP TABLE Pets;
- Data Manipulation Language (DML): DML commands are used for managing data within the database tables. The most common DML commands include INSERT, UPDATE, and DELETE.
  - INSERT INTO Pets (PetName, PetType, Age, AdoptionStatus) VALUES ('Buddy', 'Dog', 3, 'Available');
  - UPDATE Pets SET AdoptionStatus = 'Adopted' WHERE PetID = 1;
  - DELETE FROM Pets WHERE PetID = 1;
- Data Query Language (DQL): DQL primarily involves the SELECT command, which is used to retrieve data from the database.
  - Select Query: SELECT * FROM Pets;
  - From Clause: SELECT PetName, PetType, Age FROM Pets;
  - Where Clause: SELECT PetName, Breed FROM Pets WHERE AdoptionStatus = 'Available';
- Join Query: Joins are used to retrieve data from multiple tables based on a related column between them. MySQL supports different types of joins, including INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN.
  - Inner Join: SELECT Pets.PetName, Adoptions.AdopterName FROM Pets INNER JOIN Adoptions ON Pets.PetID = Adoptions.PetID;
  - Left Join: SELECT Pets.PetName, Adoptions.AdopterName FROM Pets LEFT JOIN Adoptions ON Pets.PetID = Adoptions.PetID;
  - Outer Join: SELECT Pets.PetName, Adoptions.AdopterName FROM Pets RIGHT JOIN Adoptions ON Pets.PetID = Adoptions.PetID;
  - Full Join: SELECT Pets.PetName, Adoptions.AdopterName FROM Pets LEFT JOIN Adoptions ON Pets.PetID = Adoptions.PetID UNION SELECT Pets.PetName, Adoptions.AdopterName FROM Pets RIGHT JOIN Adoptions ON Pets.PetID = Adoptions.PetID;

1.6.2. Why Servlet, JSP and Bean?

- Servlets, JSP (Java Server Pages), and JavaBeans are key components of Java-based web applications. Each of these technologies serves a specific role in creating dynamic, interactive, and maintainable web applications. Here's an explanation of why they are used together and their respective roles:
    - Servlets:
        - **Role:** Servlets are Java programs that run on a server and handle client requests, typically sent via HTTP. They are the backbone of Java web applications, enabling server-side processing of data and the generation of dynamic content.
        - Why Use Servlets?
            - Efficient Request Handling: Servlets efficiently handle multiple requests from clients, managing sessions and applying business logic.
            - Flexibility: They can process form data, interact with databases, manage sessions, and more. They are suitable for tasks that involve substantial server-side processing.
            - Integration: Servlets can be integrated with other Java technologies like JSP and JavaBeans, enabling a clear separation of concerns within the application.
    - JSP (Java Server Pages):
        - Role: JSP is a technology that simplifies the creation of dynamic web pages. It allows developers to embed Java code directly into HTML, enabling the generation of dynamic content at the server side before it is sent to the client's browser.
        - Why Use JSP?
            - Separation of Presentation and Logic: JSP promotes a clear separation between the presentation layer (HTML) and the business logic (Java). This makes the web application easier to maintain and scale.
            - Ease of Use: JSP pages are easier to write and understand for developers who are familiar with HTML. They allow embedding of Java code in a way that makes web development more intuitive.
            - Tag Libraries: JSP supports custom tag libraries, which allow developers to create reusable components and reduce code duplication.
    - JavaBeans:
        - Role: JavaBeans are reusable Java components that encapsulate multiple objects into a single object (the bean). They are typically used to manage data and business logic in a web application, providing a way to pass data between the different layers of the application.
        - Why Use JavaBeans?
            - Encapsulation and Reusability: JavaBeans encapsulate the data and provide getter and setter methods to access the properties. This encapsulation makes the code more modular and reusable across different parts of the application.
            - Consistency and Clean Code: By using beans, developers can maintain consistency in how data is handled throughout the application. This leads to cleaner, more organized code.
            - Integration with JSP: JavaBeans can be easily used in JSP pages to handle form data, manage session attributes, and interact with the business logic layer. They help maintain a clear separation between the presentation and business logic.

- Why Use Them Together?
  - MVC Architecture: Using Servlets, JSP, and JavaBeans together supports the Model-View-Controller (MVC) architecture, which is a widely adopted design pattern in web development.
  - Servlets act as Controllers, managing the flow of the application by handling requests and responses, processing data, and interacting with the business logic.
  - JSP pages serve as the View, responsible for displaying the data to the user in a formatted manner.
  - JavaBeans represent the Model, encapsulating the application's data and business logic.

1.6.3. Servlet/JSP together

- Working Together:
  - Request Handling (Servlets):
    - Servlets act as the controller in a Model-View-Controller (MVC) architecture. They handle HTTP requests from clients, process them (often involving interaction with a database or other business logic), and then determine the appropriate view (usually a JSP page) to render the response.
    - For example, a servlet might receive a form submission, validate the data, update the database, and then forward the request to a JSP page to display the result or confirmation to the user.
  - Dynamic Content Generation (JSP):
    - JSPs are used to generate the HTML (or other types of content) that is sent back to the client. They are primarily responsible for the view layer, displaying data and providing the user interface.
    - JSP pages can easily access JavaBeans, which might have been populated by the servlet, to display dynamic data. JSPs can also use Java code snippets, but it is considered best practice to keep JSPs as free from Java logic as possible, focusing them on presentation.
- Data Exchange (JavaBeans):
    - JavaBeans are often used to pass data between servlets and JSPs. A servlet might instantiate a bean, set its properties based on business logic or database results, and then store the bean in a request or session scope.
    - The JSP can then access this bean to display the data to the user, ensuring that the JSP remains focused on presentation rather than business logic.
- Example Workflow:
  - User Request:
    - A user submits a form on a web page (e.g., a login form).
  - Servlet Processing:
    - The form submission is sent to a servlet. The servlet processes the request, such as verifying the user's credentials by checking against a database.
    - If the credentials are valid, the servlet might create a UserBean to store user details and set this bean as a request attribute.

    - Forwarding to JSP:
        - The servlet forwards the request (and the associated UserBean) to a JSP page using RequestDispatcher.forward().
- Advantages of Using Servlet and JSP Together:
    - Separation of Concerns:
        - Servlets manage business logic, while JSPs focus on the presentation layer. This separation makes the application easier to maintain and develop.
    - Reusability and Modularity:
        - By using JavaBeans and keeping logic in servlets, components become more reusable and modular. JSPs can be easily modified for different views without affecting the underlying logic.
    - Maintainability:
        - Changes in the business logic or presentation layer can be made independently, making the codebase easier to maintain.
    - Scalability**:**
        - The clear separation of responsibilities and the use of Java's efficient threading model in servlets ensure that applications are scalable and can handle multiple simultaneous requests efficiently.
- MVC Pattern:
    - The combination of servlets and JSPs naturally fits into the MVC (Model-View-Controller) pattern, which is a well-known architectural pattern for building scalable and maintainable web applications.
    - **Conclusion:** By using Servlets and JSPs together, developers can create robust web applications that are well-organized, scalable, and easier to maintain. Servlets handle the backend processing and business logic, while JSPs are used for creating the user interface, leading to a clear separation of concerns and a more maintainable codebase.

1.6.4. Bean/JSP together

In Java web development, the combination of JavaBeans and JavaServer Pages (JSP) is a powerful pattern for separating concerns between data management and presentation. This approach is essential in projects like "Adopt a Pet and Care," where maintaining a clean, modular structure is critical for scalability and ease of maintenance.

- Role of JavaBeans
    - JavaBeans are reusable software components that follow specific conventions:
    - Properties: JavaBeans have private attributes with public getter and setter methods, allowing other parts of the application, such as JSPs, to interact with them.
    - No-Argument Constructor: JavaBeans have a no-argument constructor, enabling them to be easily instantiated.
    - Serializable: Beans are serializable, meaning they can be converted to a byte stream, stored, and restored later.
- In the context of the "Adopt a Pet and Care" project, JavaBeans are used to encapsulate the data and business logic related to different entities like Pet, User, and Adoption. For example:
    - PetBean: Manages attributes like PetID, Name, Breed, Age, and Description, along with methods for accessing and manipulating this data.

  - UserBean: Manages user-related data such as Username, Password, Email, and handles user authentication logic.
  - AdoptionBean: Tracks the status of adoption requests, linking users with the pets they are interested in adopting.

- Role of JSP
  - JSPs are primarily used for displaying data and interacting with users. They provide a dynamic interface by allowing HTML, CSS, and Java code to be combined in one file. JSPs are designed to be more readable and maintainable by separating business logic from the presentation layer.
  - In the "AdoptEase" project, JSPs handle tasks such as:
  - Displaying Pet Information: JSP pages render lists of pets available for adoption, displaying data retrieved from PetBean.
  - User Interaction: Forms for user registration, login, and adoption requests are created in JSPs, which then process this input using JavaBeans.
  - Dynamic Content Rendering: JSPs dynamically generate HTML content based on the data stored in JavaBeans, such as displaying personalized greetings or user-specific adoption status updates.

CHAPTER-2

Related Previous Work/Market Survey

In the field of pet adoption and care, there has been considerable research and development, leading to the creation of various web portals, applications, and studies aimed at addressing the challenges of pet overpopulation, adoption, and responsible pet ownership.

1. Research on the Problem:

    1.1. Pet Overpopulation and Shelter Dynamics: Numerous studies have highlighted the issue of pet overpopulation, particularly in urban areas, leading to overcrowded shelters and high euthanasia rates. Research has shown that increasing public awareness about adoption and providing resources for pet owners can significantly reduce these rates.

    1.2. Behavioural Studies: Research on pet behaviour, especially concerning abandoned or shelter pets, has provided insights into the importance of post-adoption support. Studies emphasize the need for educational resources for new pet owners to manage and train their pets effectively, reducing the likelihood of pets being returned to shelters.

2. Existing Web Portals:

    2.1. Petfinder: One of the most well-known platforms, Petfinder connects potential adopters with animals in shelters across the U.S. and Canada. It provides detailed profiles of pets and supports local shelters by expanding their reach.

    2.2. Adopt-a-Pet: This portal is another significant platform that partners with thousands of shelters and rescue organizations. It offers tools for potential adopters to search for pets based on various criteria and provides resources on pet care.

    2.3. ASPCA Adoption Portal: The American Society for the Prevention of Cruelty to Animals (ASPCA) offers an adoption portal that lists adoptable pets from its own facilities and partner shelters. It also provides extensive educational materials for new pet owners.

3. Summary of Results:

    3.1. Adoption Rates Improvement: Portals like Petfinder and Adopt-a-Pet have contributed to increased adoption rates by making it easier for people to find adoptable pets and learn about the adoption process.

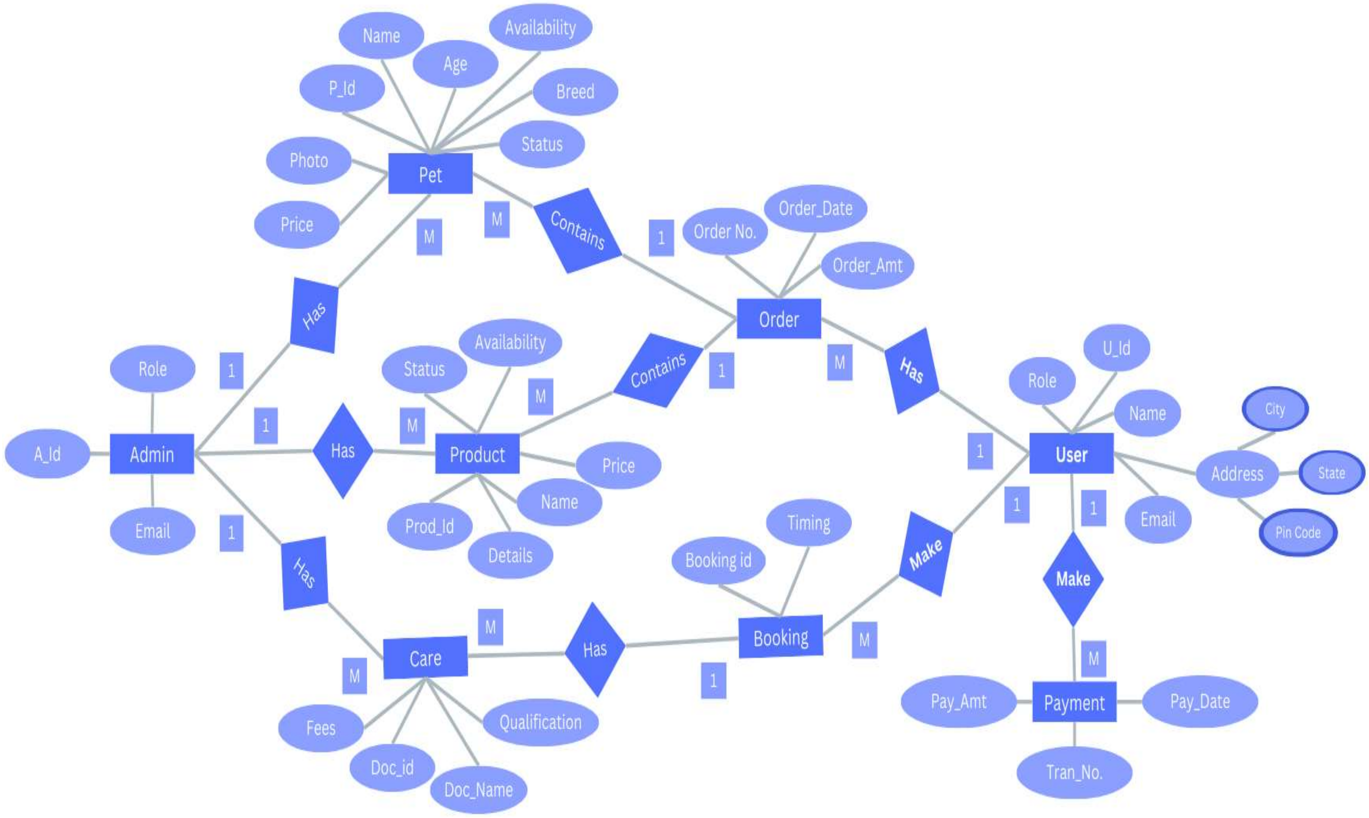
    3.2. Educational Outreach: These platforms often include resources on pet care, training, and health, helping to educate new owners and reduce the number of pets returned to shelters.

    3.3. Data-Driven Insights: The aggregation of adoption data by these platforms has enabled better understanding of trends in pet adoption, influencing policies and practices in animal welfare organizations.

    3.4. These existing platforms have laid the groundwork for the "Adopt a Pet and Care" project by demonstrating the effectiveness of online portals in promoting pet adoption and providing educational

3.5. resources. The project aims to build on this foundation by integrating more personalized support for adopters, addressing the gaps identified in previous research, and leveraging new technologies to enhance user experience and outcomes.

4. ER Diagram



CHAPTER-3

IMPLEMENTATION AND RESULTS

1. Hardware Requirements

- Processor: Intel Core i5
- RAM: Minimum 8 GB
- Network: Reliable internet connection

2. Software Requirements

- Web Server: Apache Tomcat
- Database Management System: MySQL
- Java Development kit (JDK): JDK 8
- Technologies: Java, JSP, Servlets, and JavaBeans
- Operating System: Windows 10/11

3. Implementation Details:

The implementation of the "AdoptEase" project involves a structured approach to develop a functional, scalable, and user-friendly web platform. The development process is divided into several stages, each focusing on different aspects of the project. Below is a detailed overview of the key implementation steps:

3.1. System Architecture

The "AdoptEase" platform is based on a three-tier architecture, which includes the Presentation Layer, Business Logic Layer, and Data Access Layer:

- **Presentation Layer (Front-end):**
  - **Technologies:** HTML, CSS, JavaScript, JSP (Java Server Pages)
  - **Role:** This layer is responsible for the user interface and user experience. It includes web pages that users interact with to browse pets, submit adoption applications, and access other features.

- **Business Logic Layer (Back-end):**
  - **Technologies:** Java Servlets, JavaBeans.
  - **Role:** This layer handles the core functionality of the application, including processing user requests, implementing business rules, and managing sessions. It serves as the intermediary between the Presentation Layer and the Data Access Layer.

- **Data Access Layer (Database):**
  - **Technologies:** MySQL
  - **Role:** This layer manages the storage, retrieval, and manipulation of data. It interacts with the MySQL database to perform operations such as storing pet profiles, user information, and adoption records.

3.2. Database Design

- **Database Schema:**
  - **Tables:** The MySQL database will consist of multiple tables, such as Users, Pets, Shelters, Adoptions, and Messages.
  - **Relationships:** The tables will be related using primary and foreign keys, ensuring data integrity and supporting complex queries.

- **Key Tables and Fields:**
  - **Users Table:** Stores user information (UserID, Username, Password, Email, Role).
  - **Pets Table:** Stores details of pets available for adoption (PetID, PetName, Breed, Age, AdoptionStatus, ShelterID).
  - **Shelters Table:** Stores information about shelters (ShelterID, ShelterName, Location, ContactInfo).
  - **Adoptions Table:** Records adoption transactions (AdoptionID, UserID, PetID, AdoptionDate).
  - **Messages Table:** Facilitates communication between users and shelters (MessageID, SenderID, ReceiverID, MessageContent, Timestamp).

3.3. Front-end Development

- **User Interface Design:**
  - **Wireframes and Prototypes:** Initial designs are created using tools like Adobe XD or Figma to visualize the layout and flow of the application.
  - **Responsive Design:** The platform will be responsive, ensuring that it functions well on various devices, including desktops, tablets, and smartphones.

- **JSP Integration:**
  - **Dynamic Content:** JSP pages will be used to dynamically generate HTML content based on user interactions and database queries.
  - **Form Handling:** JSP will handle form submissions, passing data to Servlets for processing.

3.4. Back-end Development

- **Servlets:**
  - **Request Handling:** Servlets will manage incoming HTTP requests, interact with the database through JavaBeans, and determine the appropriate JSP pages to display.
  - **Session Management:** Servlets will manage user sessions to maintain login states and track user activities.

- **JavaBeans:**
  - **Data Encapsulation:** JavaBeans will be used to encapsulate business logic and data, providing a consistent way to interact with the database.
  - **Reusable Components:** JavaBeans will be designed as reusable components, promoting code reuse across different parts of the application.

3.5. Database Operations

- **CRUD Operations:**
  - **Create:** Inserting new records into the database (e.g., new pet listings, new user registrations).
  - **Read:** Retrieving data from the database (e.g., fetching pet details for display, searching for pets).
  - **Update:** Modifying existing records (e.g., updating pet adoption status).
  - **Delete:** Removing records from the database (e.g., deleting pet listings after adoption).
  - **Joins and Queries:** Complex SQL queries, including joins, will be used to retrieve data from multiple tables and present it to the users in a meaningful way (e.g., showing pets from a specific shelter).

3.6. Testing and Debugging

- **Unit Testing:**
  - **Tools:** JUnit or similar testing frameworks will be used to test individual components (e.g., Servlets, JavaBeans) for correctness.
- **Integration Testing:** Testing will ensure that different components of the system work together seamlessly.
- **User Acceptance Testing (UAT):** The final product will be tested by end-users to ensure it meets their expectations and requirements.

3.7. Deployment

- **Server Configuration:**
    - **Web Server:** Apache Tomcat will be configured to host the web application.
    - **Database Server:** MySQL server will be set up to store and manage the application data.
- **Deployment Environment:** The application will be deployed on a staging server for final testing before going live on a production server.

3.8. Documentation

- **Code Documentation:** Comprehensive documentation will be provided for all code components, including comments and external documentation.

- **User Manuals:** User guides and manuals will be created to assist end-users and administrators in using and managing the platform.

3.9. Maintenance and Updates

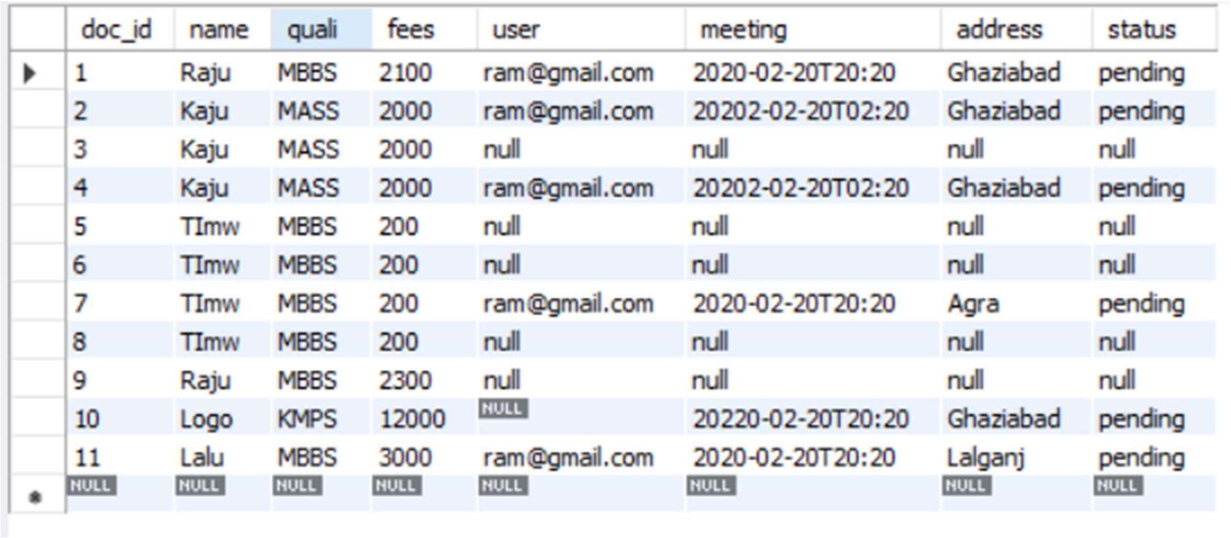
- **Ongoing Support:** Post-deployment, the project will involve ongoing maintenance, including bug fixes, security updates, and feature enhancements based on user feedback.

- **Version Control:** Git or another version control system will be used to manage changes and updates to the codebase.

4. Snapshots of Webpages, Codes and Database:

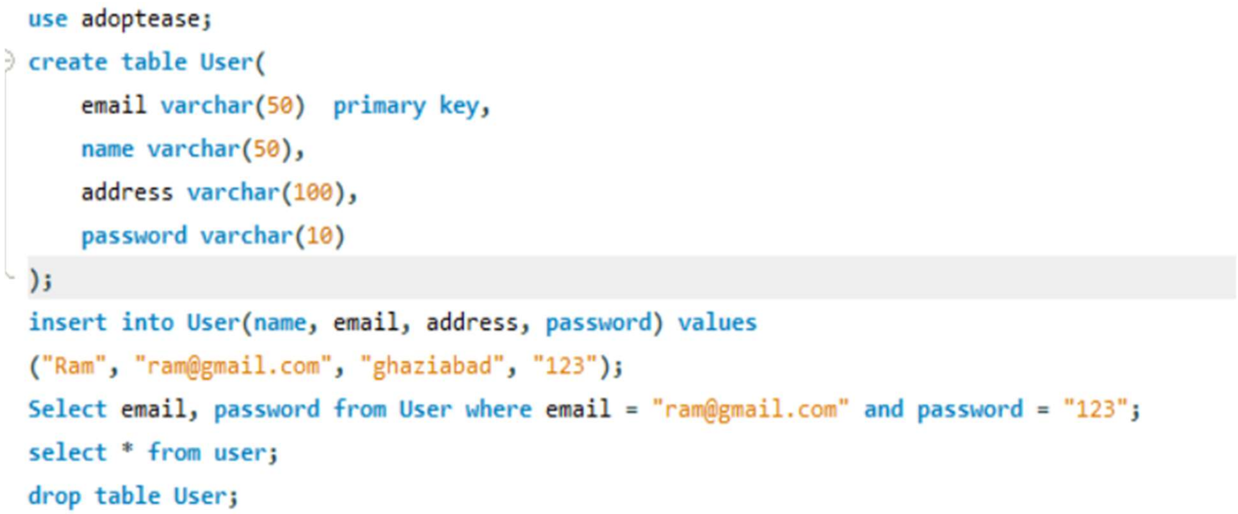
4.1 SQL Database:

- Admin Pet



	doc_id	name	quali	fees	user	meeting	address	status
▶	1	Raju	MBBS	2100	ram@gmail.com	2020-02-20T20:20	Ghaziabad	pending
	2	Kaju	MASS	2000	ram@gmail.com	20202-02-20T02:20	Ghaziabad	pending
	3	Kaju	MASS	2000	null	null	null	null
	4	Kaju	MASS	2000	ram@gmail.com	20202-02-20T02:20	Ghaziabad	pending
	5	TImw	MBBS	200	null	null	null	null
	6	TImw	MBBS	200	null	null	null	null
	7	TImw	MBBS	200	ram@gmail.com	2020-02-20T20:20	Agra	pending
	8	TImw	MBBS	200	null	null	null	null
	9	Raju	MBBS	2300	null	null	null	null
	10	Logo	KMPS	12000	NULL	20220-02-20T20:20	Ghaziabad	pending
	11	Lalu	MBBS	3000	ram@gmail.com	2020-02-20T20:20	Lalganj	pending
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

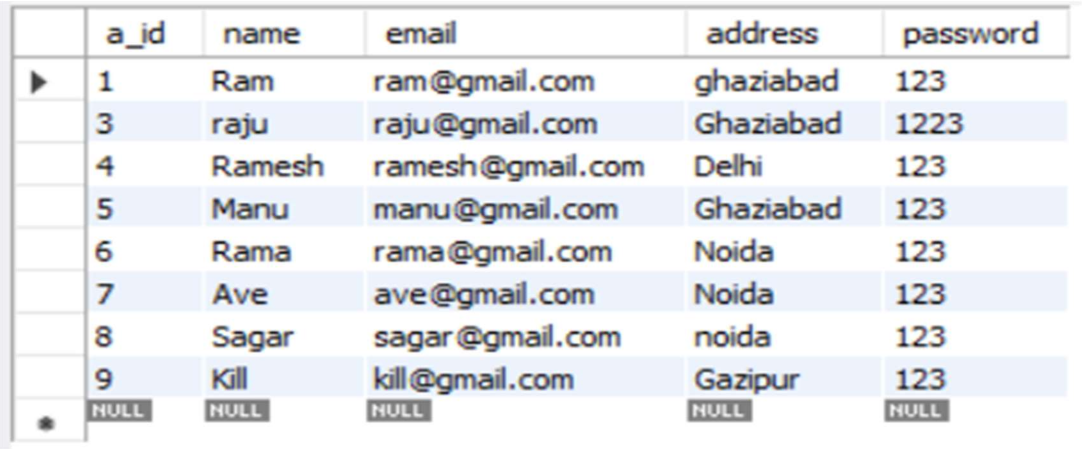
Fig. Table



```
use adoptease;
create table User(
    email varchar(50)  primary key,
    name varchar(50),
    address varchar(100),
    password varchar(10)
);
insert into User(name, email, address, password) values
("Ram", "ram@gmail.com", "ghaziabad", "123");
Select email, password from User where email = "ram@gmail.com" and password = "123";
select * from user;
drop table User;
```

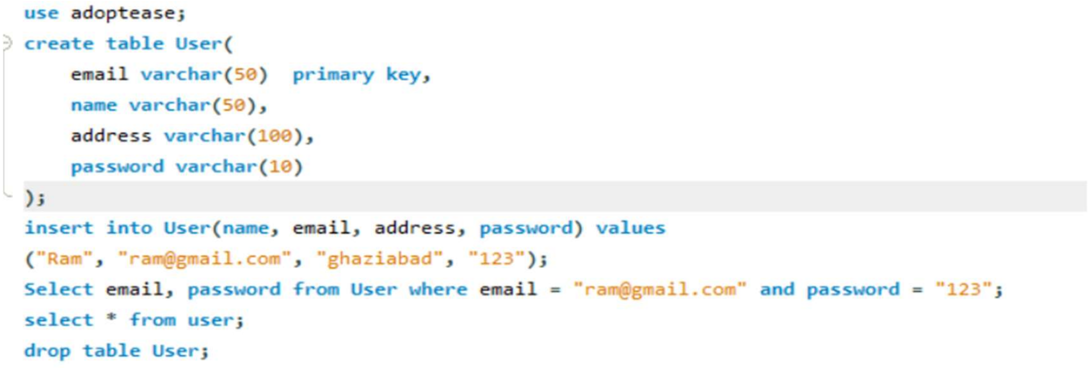
Fig. Query

- User Table:



	a_id	name	email	address	password
▶	1	Ram	ram@gmail.com	ghaziabad	123
	3	raju	raju@gmail.com	Ghaziabad	1223
	4	Ramesh	ramesh@gmail.com	Delhi	123
	5	Manu	manu@gmail.com	Ghaziabad	123
	6	Rama	rama@gmail.com	Noida	123
	7	Ave	ave@gmail.com	Noida	123
	8	Sagar	sagar@gmail.com	noida	123
	9	Kill	kill@gmail.com	Gazipur	123
*	NULL	NULL	NULL	NULL	NULL

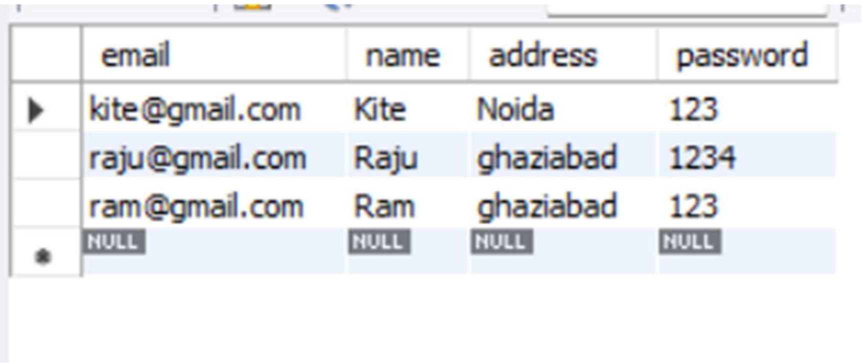
Fig. Table



```
use adoptease;
create table User(
    email varchar(50)  primary key,
    name varchar(50),
    address varchar(100),
    password varchar(10)
);
insert into User(name, email, address, password) values
("Ram", "ram@gmail.com", "ghaziabad", "123");
Select email, password from User where email = "ram@gmail.com" and password = "123";
select * from user;
drop table User;
```

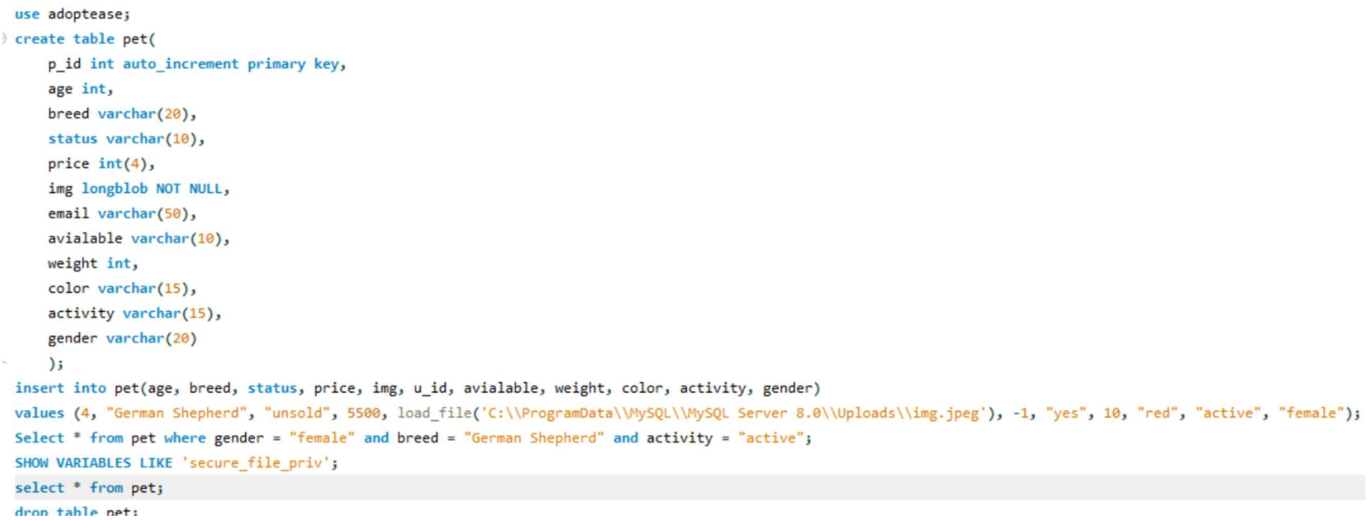
Fig. Query

- Admin:



	email	name	address	password
▶	kite@gmail.com	Kite	Noida	123
	raju@gmail.com	Raju	ghaziabad	1234
	ram@gmail.com	Ram	ghaziabad	123
*	NULL	NULL	NULL	NULL

Fig. Table

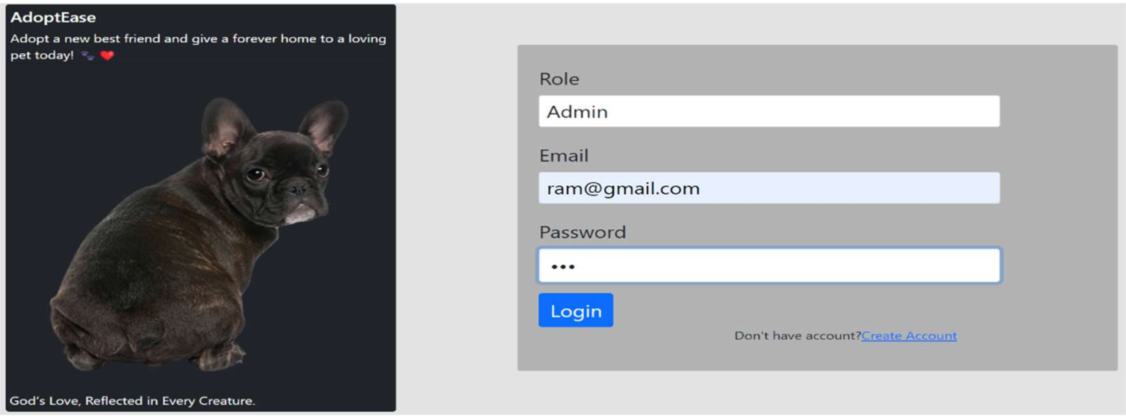


```
use adoptease;
create table pet(
    p_id int auto_increment primary key,
    age int,
    breed varchar(20),
    status varchar(10),
    price int(4),
    img longblob NOT NULL,
    email varchar(50),
    avialable varchar(10),
    weight int,
    color varchar(15),
    activity varchar(15),
    gender varchar(20)
    );
insert into pet(age, breed, status, price, img, u_id, avialable, weight, color, activity, gender)
values (4, "German Shepherd", "unsold", 5500, load_file('C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\img.jpeg'), -1, "yes", 10, "red", "active", "female");
Select * from pet where gender = "female" and breed = "German Shepherd" and activity = "active";
SHOW VARIABLES LIKE 'secure_file_priv';
select * from pet;
drop table pet;
```

Fig. Query

4.2 Webpages and Code:

- Admin Login:



AdoptEase

Adopt a new best friend and give a forever home to a loving pet today! 🐾 ❤️

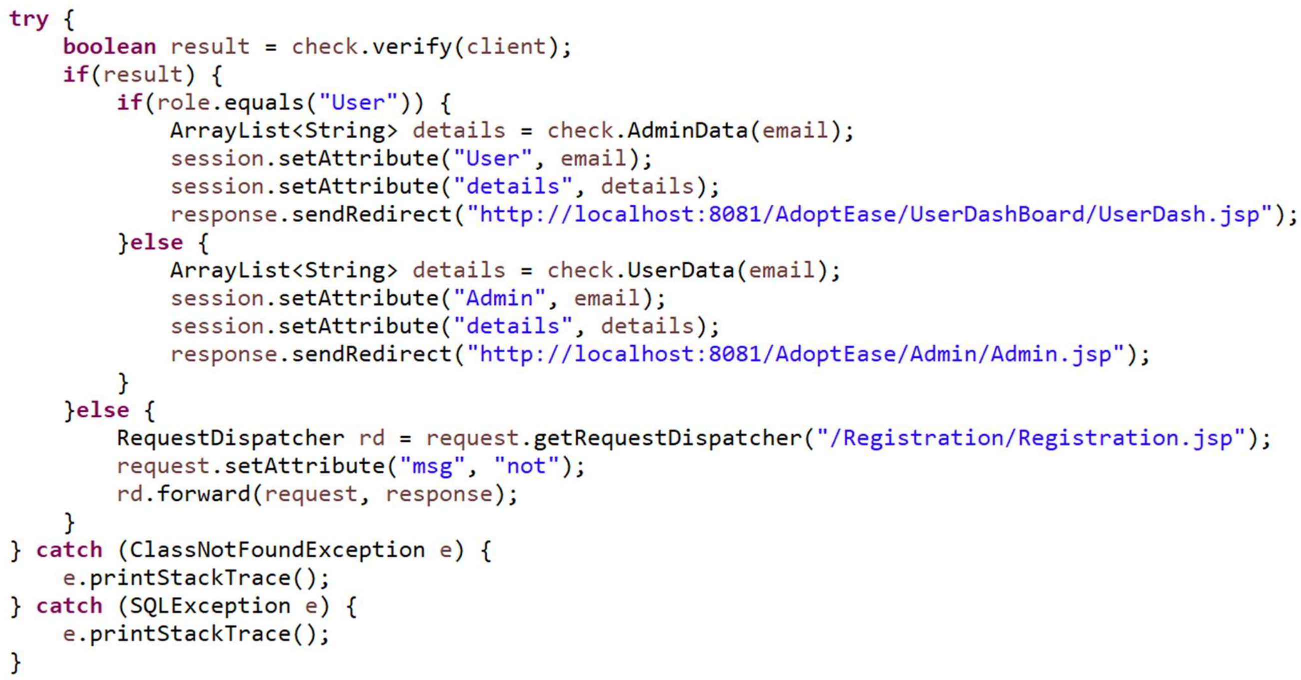
God's Love, Reflected in Every Creature.

Role: Admin

Email: ram@gmail.com

Password: •••

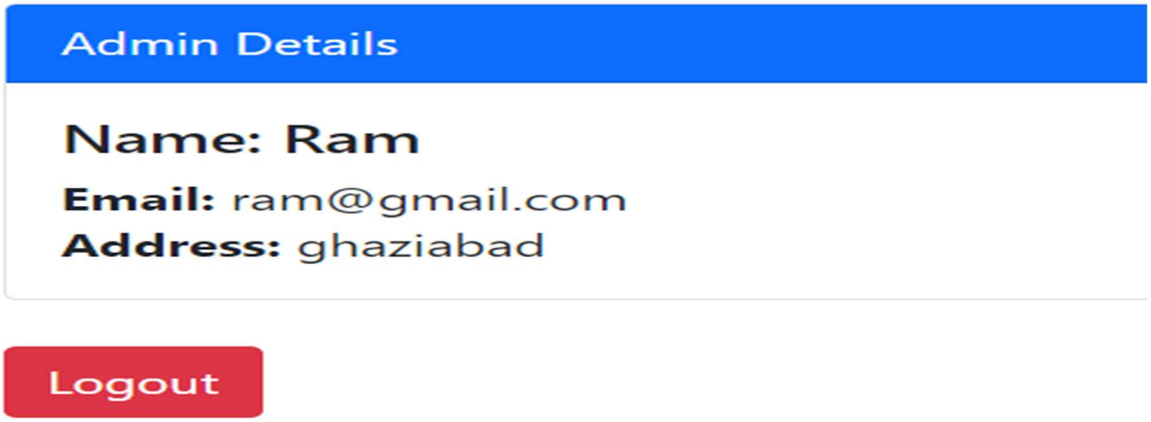
Fig. Webpage



```
try {
    boolean result = check.verify(client);
    if(result) {
        if(role.equals("User")) {
            ArrayList<String> details = check.AdminData(email);
            session.setAttribute("User", email);
            session.setAttribute("details", details);
            response.sendRedirect("http://localhost:8081/AdoptEase/UserDashBoard/UserDash.jsp");
        }else {
            ArrayList<String> details = check.UserData(email);
            session.setAttribute("Admin", email);
            session.setAttribute("details", details);
            response.sendRedirect("http://localhost:8081/AdoptEase/Admin/Admin.jsp");
        }
    }else {
        RequestDispatcher rd = request.getRequestDispatcher("/Registration/Registration.jsp");
        request.setAttribute("msg", "not");
        rd.forward(request, response);
    }
} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
}
```

Fig. Code

- Admin Logout:

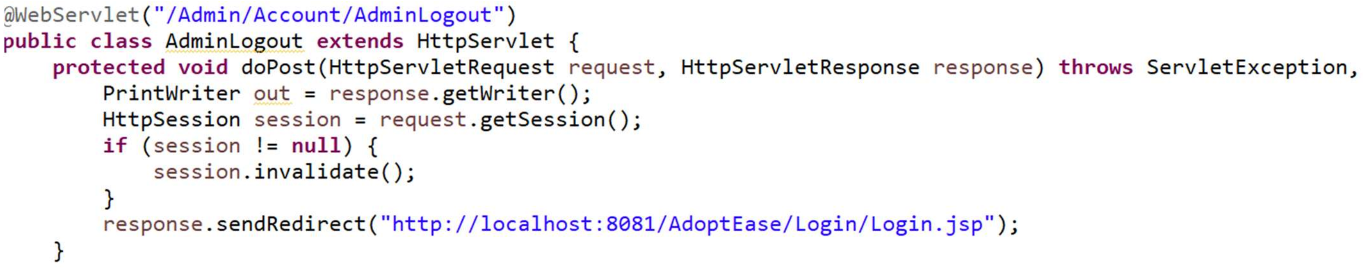


Admin Details

Name: Ram

Email: ram@gmail.com
Address: ghaziabad

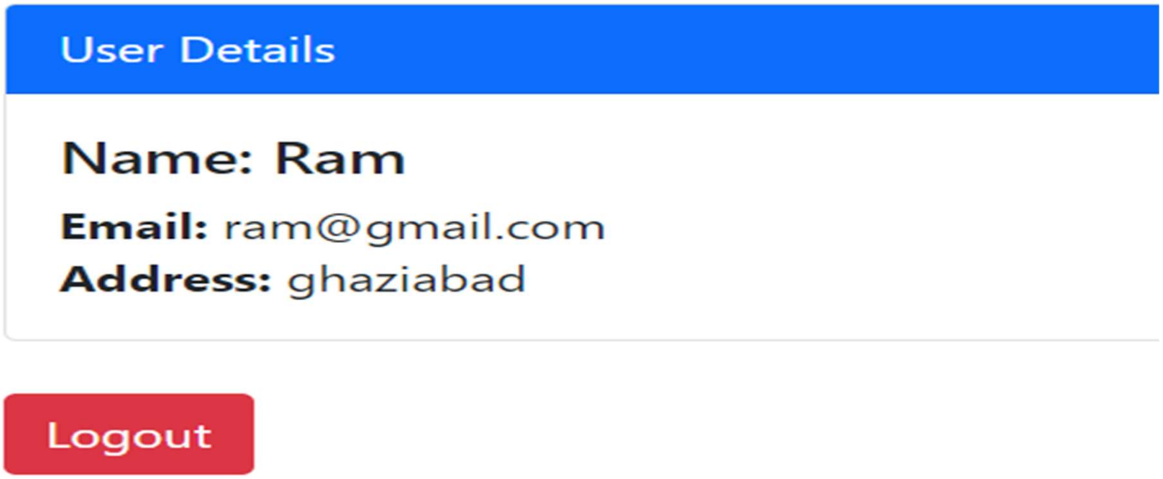
Fig. Webpage



```
@WebServlet("/Admin/Account/AdminLogout")
public class AdminLogout extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession();
        if (session != null) {
            session.invalidate();
        }
        response.sendRedirect("http://localhost:8081/AdoptEase/Login/Login.jsp");
    }
```

Fig. Code

- User Logout:

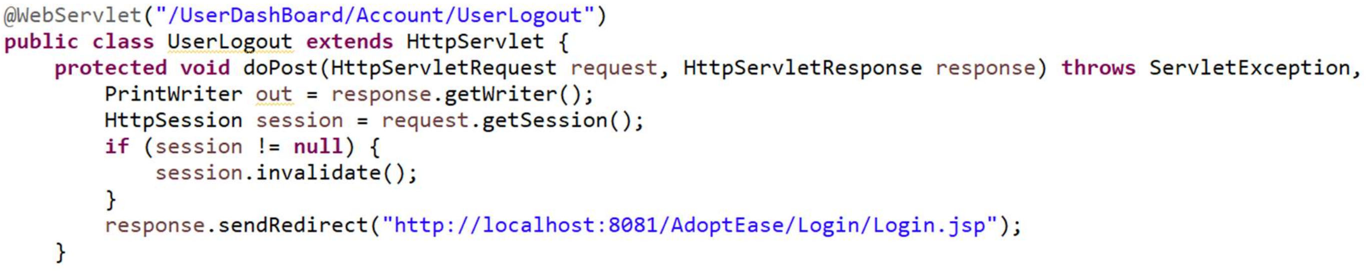


User Details

Name: Ram

Email: ram@gmail.com
Address: ghaziabad

Fig. Webpage



```
@WebServlet("/UserDashBoard/Account/UserLogout")
public class UserLogout extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
        PrintWriter out = response.getWriter();
        HttpSession session = request.getSession();
        if (session != null) {
            session.invalidate();
        }
        response.sendRedirect("http://localhost:8081/AdoptEase/Login/Login.jsp");
    }
```

Fig. Code

- Admin/User Registration:

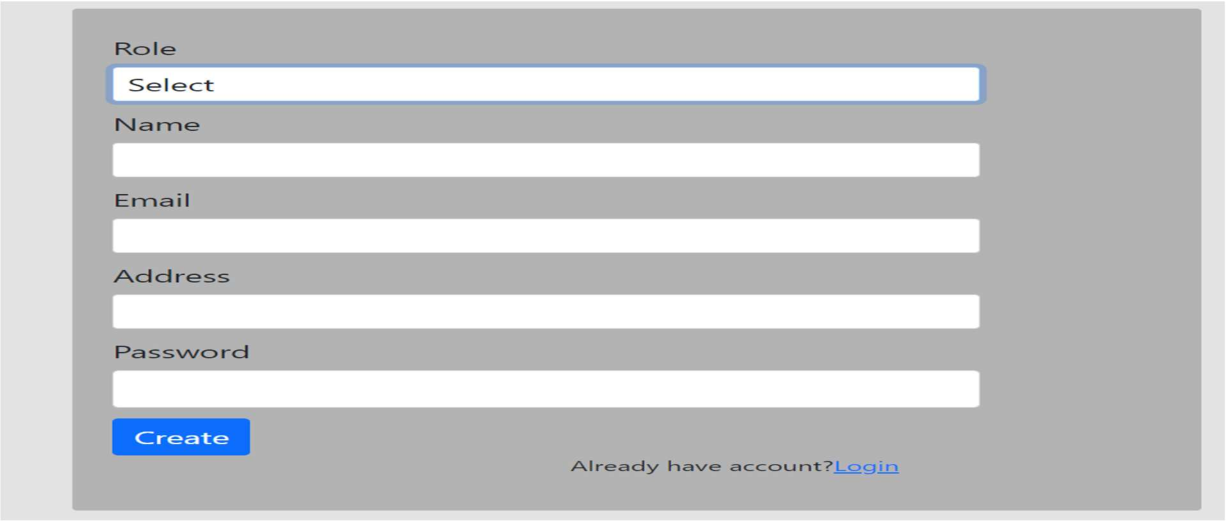
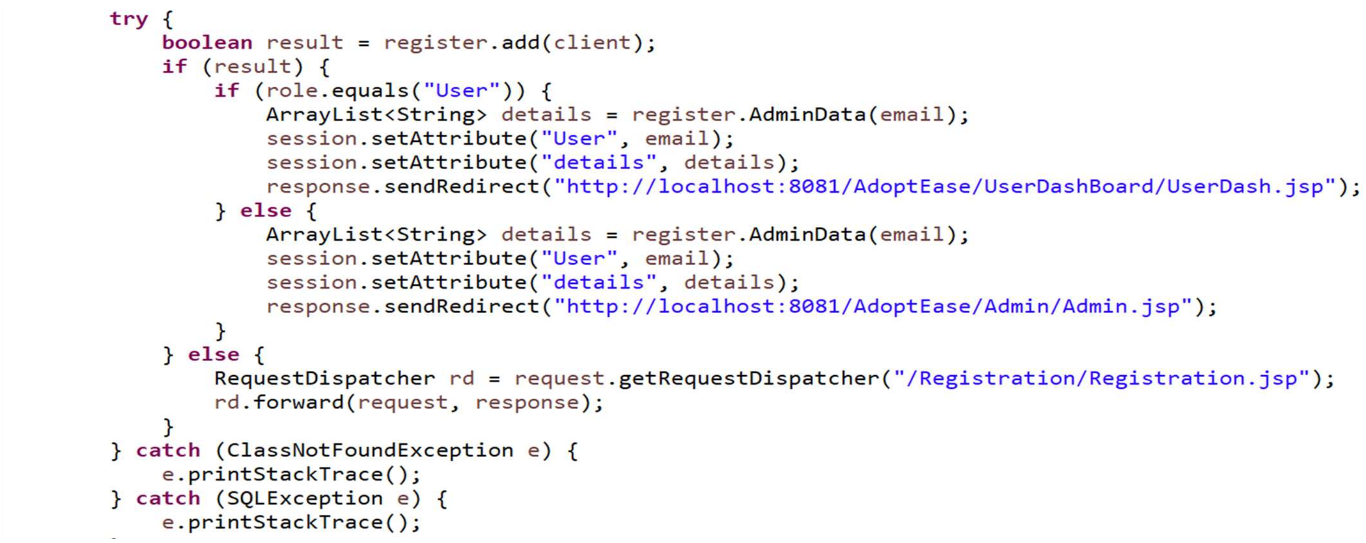


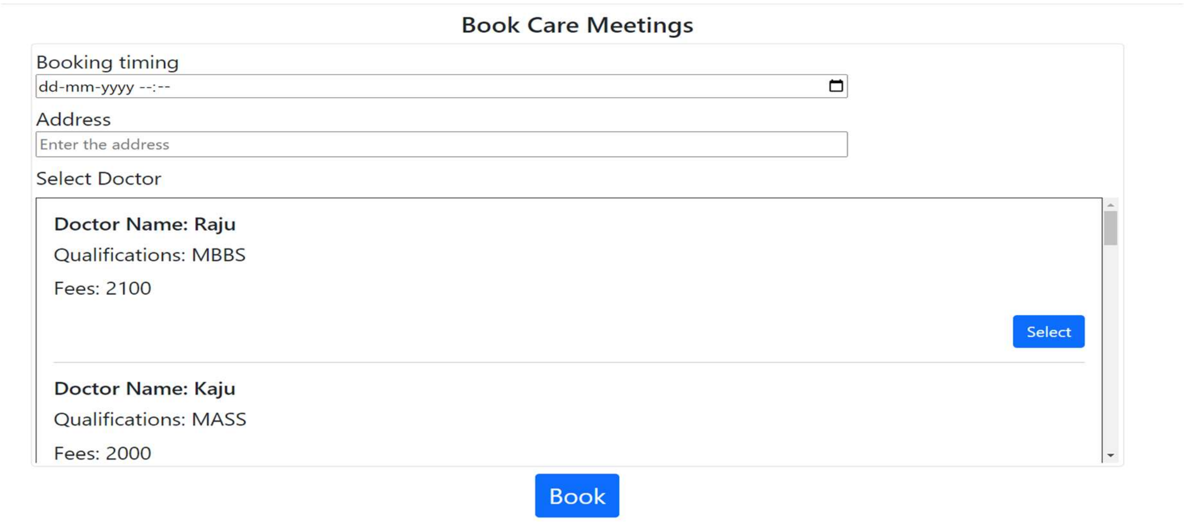
Fig. Webpage



```
    try {
        boolean result = register.add(client);
        if (result) {
            if (role.equals("User")) {
                ArrayList<String> details = register.AdminData(email);
                session.setAttribute("User", email);
                session.setAttribute("details", details);
                response.sendRedirect("http://localhost:8081/AdoptEase/UserDashBoard/UserDash.jsp");
            } else {
                ArrayList<String> details = register.AdminData(email);
                session.setAttribute("User", email);
                session.setAttribute("details", details);
                response.sendRedirect("http://localhost:8081/AdoptEase/Admin/Admin.jsp");
            }
        } else {
            RequestDispatcher rd = request.getRequestDispatcher("/Registration/Registration.jsp");
            rd.forward(request, response);
        }
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
```

Fig. Code

- User Meeting Booking



Book Care Meetings

Booking timing

dd-mm-yyyy --:--

Address

Enter the address

Select Doctor

Doctor Name: Raju

Qualifications: MBBS

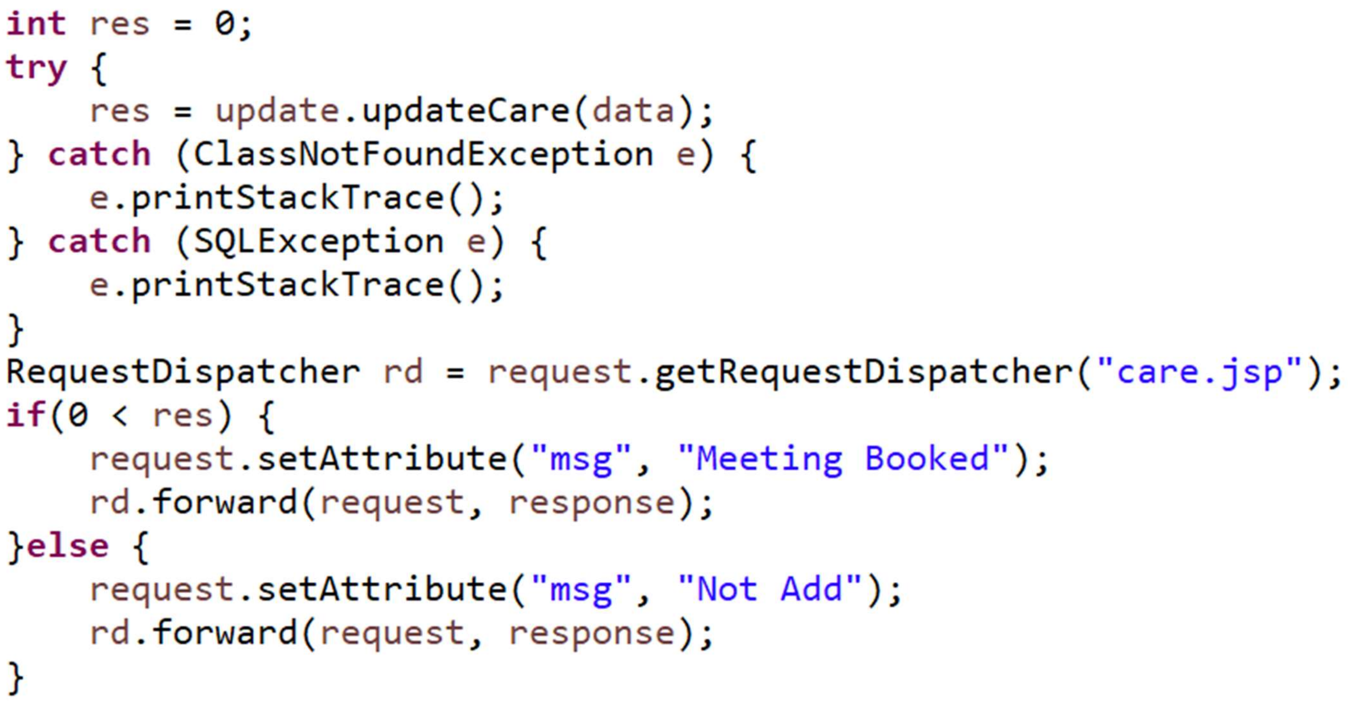
Fees: 2100

Doctor Name: Kaju

Qualifications: MASS

Fees: 2000

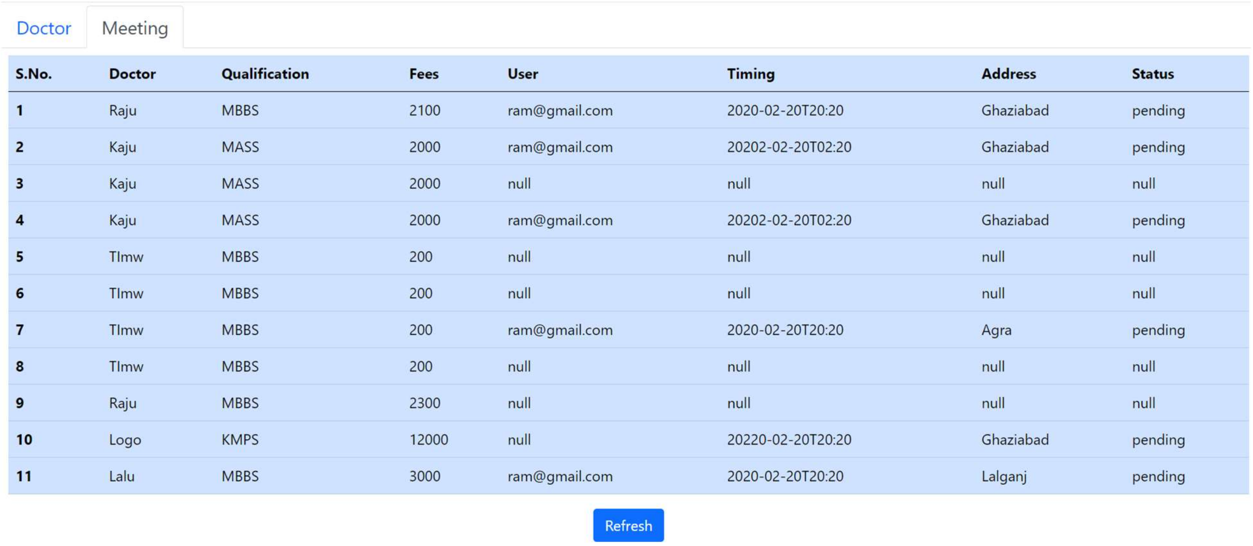
Fig. Webpage



```
int res = 0;
try {
    res = update.updateCare(data);
} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
}
RequestDispatcher rd = request.getRequestDispatcher("care.jsp");
if(0 < res) {
    request.setAttribute("msg", "Meeting Booked");
    rd.forward(request, response);
}else {
    request.setAttribute("msg", "Not Add");
    rd.forward(request, response);
}
```

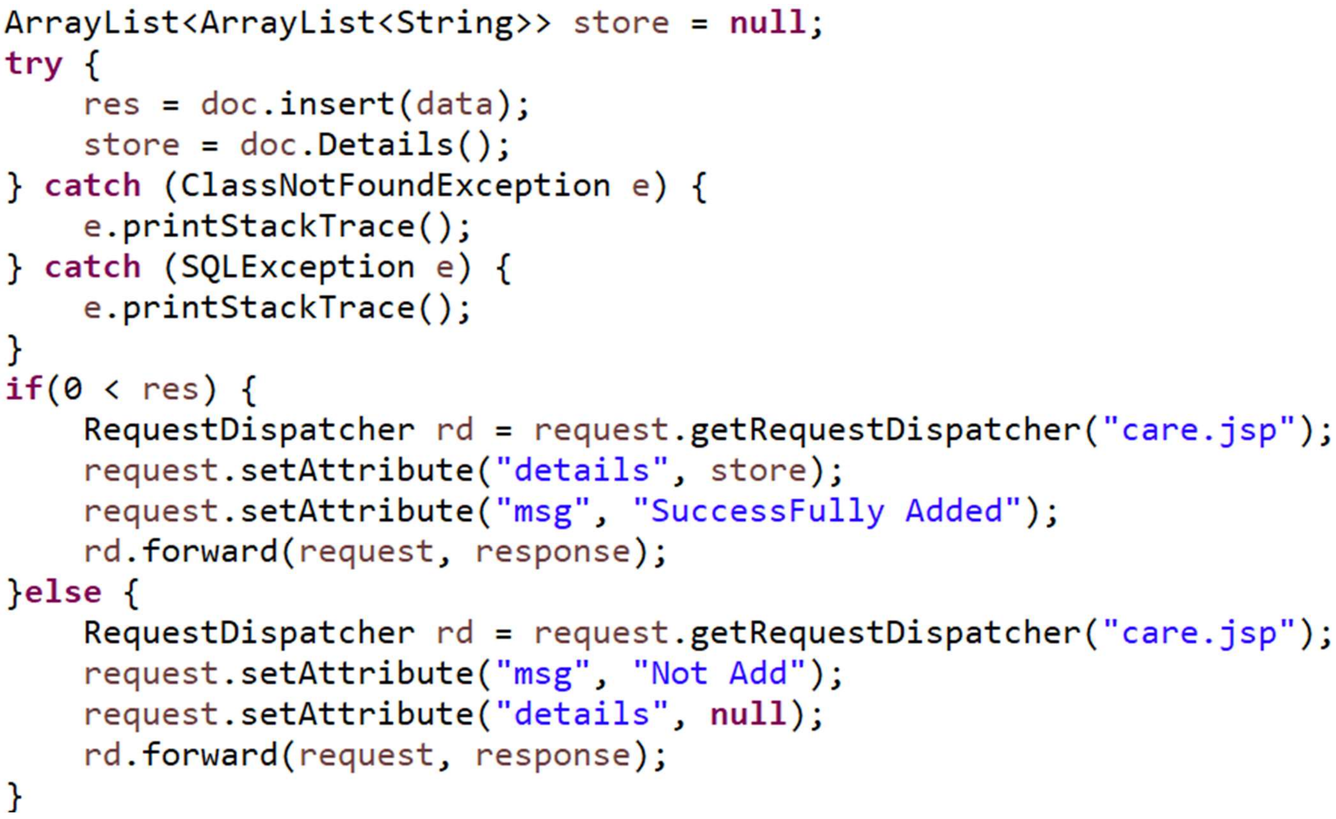
Fig. Code

- Admin Seeing Meeting



S.No.	Doctor	Qualification	Fees	User	Timing	Address	Status
1	Raju	MBBS	2100	ram@gmail.com	2020-02-20T20:20	Ghaziabad	pending
2	Kaju	MASS	2000	ram@gmail.com	20202-02-20T02:20	Ghaziabad	pending
3	Kaju	MASS	2000	null	null	null	null
4	Kaju	MASS	2000	ram@gmail.com	20202-02-20T02:20	Ghaziabad	pending
5	TImw	MBBS	200	null	null	null	null
6	TImw	MBBS	200	null	null	null	null
7	TImw	MBBS	200	ram@gmail.com	2020-02-20T20:20	Agra	pending
8	TImw	MBBS	200	null	null	null	null
9	Raju	MBBS	2300	null	null	null	null
10	Logo	KMPS	12000	null	20220-02-20T20:20	Ghaziabad	pending
11	Lalu	MBBS	3000	ram@gmail.com	2020-02-20T20:20	Lalganj	pending

Fig. Webpage



```java
ArrayList<ArrayList<String>> store = null;
try {
    res = doc.insert(data);
    store = doc.Details();
} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
}
if(0 < res) {
    RequestDispatcher rd = request.getRequestDispatcher("care.jsp");
    request.setAttribute("details", store);
    request.setAttribute("msg", "SuccessFully Added");
    rd.forward(request, response);
}else {
    RequestDispatcher rd = request.getRequestDispatcher("care.jsp");
    request.setAttribute("msg", "Not Add");
    request.setAttribute("details", null);
    rd.forward(request, response);
}
```

Fig. Code

CONCLUSION

The implementation of the "Adopt a Pet and Care" project is designed to leverage a structured approach, utilizing a Model-View-Controller (MVC) architecture to ensure a clear separation of concerns and maintainable code. By adopting Java Servlets for handling business logic, JSPs for presenting dynamic content, and JavaBeans for encapsulating data, the project aims to deliver a robust and user-friendly web application.

Key Aspects of the Implementation:

1. System Architecture: The MVC pattern effectively separates the data, business logic, and presentation layers, ensuring that each component can be developed, tested, and maintained independently.
2. Technology Stack: The use of Eclipse IDE, Apache Tomcat, and MySQL provides a solid foundation for development, deployment, and database management. The integration of Java EE technologies ensures that the application is scalable and adheres to industry standards.
3. Development Process: The structured development process, including setup, design, implementation, testing, and deployment, ensures a methodical approach to building the application. Rigorous testing and continuous monitoring further contribute to the application’s reliability and user satisfaction.

By following these implementation details, the project will address key challenges in pet adoption and care, providing an efficient platform for users to find and adopt pets, manage their profiles, and access valuable resources. The integration of feedback and ongoing support will further enhance the application’s effectiveness and user experience.

In conclusion, the "Adopt a Pet and Care" project is poised to make a significant impact by streamlining the adoption process, improving the visibility of pets in need, and offering comprehensive support to both prospective pet owners and animal welfare organizations.

Future Directions

As the "AdoptEase" project evolves, there are several potential areas for expansion and improvement that could enhance its functionality, user experience, and overall impact. These future directions include technological advancements, additional features, and broader outreach initiatives.

1. Technological Advancements

- Enhanced User Experience:
  - Mobile Application: Develop a mobile app version of the platform to reach a broader audience and provide a more convenient way for users to browse pets, manage their profiles, and apply for adoption from their smartphones.
  - Progressive Web Application (PWA): Implement a PWA to offer a more app-like experience on mobile browsers, improving accessibility and user engagement.

- Artificial Intelligence and Machine Learning:
  - Pet Matching Algorithms: Utilize AI to develop advanced algorithms that match potential adopters with pets based on behavioural characteristics, preferences, and compatibility, enhancing the adoption process.
  - Chatbots: Implement AI-powered chatbots to provide instant support, answer frequently asked questions, and guide users through the adoption process.

- Data Analytics:
  - User Behaviour Insights: Integrate analytics tools to track user interactions and behaviors on the platform, allowing for data-driven improvements and personalized user experiences.
  - Adoption Trends: Analyse adoption data to identify trends and patterns, providing insights to shelters and rescue organizations for better decision-making and resource allocation.

2. Additional Features

- Enhanced Search and Filter Options:
  - Advanced Search Filters: Expand search capabilities to include more detailed filters (e.g., pet age, breed, special needs) to help users find pets that meet their specific criteria.
  - Location-Based Services: Integrate location-based features to help users find pets and shelters near them, making the adoption process more convenient.

- Community Engagement:
  - Forums and Social Features: Add community forums or social features where users can share their adoption stories, seek advice, and connect with other pet owners.
  - Events and Campaigns: Host online and offline events, such as adoption drives, pet care workshops, and fundraisers, to engage the community and promote pet adoption.

- Educational Resources:
  - Training and Behaviour Resources: Provide access to resources on pet training and behaviour management to support new pet owners in addressing common challenges.

Appendix

The appendix provides additional information that supports and complements the main content of the project. It includes detailed technical documentation, references, and supplementary materials relevant to the "AdoptEase" project.

1. System Requirements:
    - Server:
        - Apache Tomcat version 9.0 or higher
        - Java Development Kit (JDK) 11 or higher
    - Database:
        - MySQL version 8.0 or higher
        - JDBC Driver for MySQL
    - Development Tools:
        - Eclipse IDE for Java EE Developers
        - Maven or Gradle for build automation

2. User Guide:

    - Navigating the Platform:
        - Instructions on how to search for pets, view pet details, and apply for adoption.
    - Managing User Profiles:
        - Steps to register, log in, and update user information.
    - Adoption Process:
        - Guide on how to submit an adoption request and track its status.

2. Technical References:

    - Servlets and JSP:
        - "Java Servlets and JSP: A Tutorial" by Steven Holzner
        - Official Java Servlet Documentation
    - Database Management:
        - "MySQL Cookbook" by Paul DuBois
        - Official MySQL Documentation
    - Development Tools:
        - Eclipse IDE: Eclipse Official Website
        - Apache Tomcat: Tomcat Official Website