Requirements Engineering

Doggy Day care System

Submitted By: Sean Conway(T00196102)

Computing with Games Development

Date Submitted: 05/11/2018

**Table of Contents**

[1. Introduction/overview 3](#_Toc474162299)

[2. Functional Components 4](#_Toc474162300)

[3. User Requirements 5](#_Toc474162301)

[4. System Requirements 6](#_Toc474162302)

[4.1. System Level Use Case Diagram 6](#_Toc474162303)

[4.2. Module 1 (Use active Verb + Noun) **Error! Bookmark not defined.**](#_Toc474162304)

[4.2.1. Functional Requirement 1 (Use active verb + noun) 14](#_Toc474162305)

[4.2.2. Functional Requirement 2 (Use active verb + noun) 16](#_Toc474162306)

[4.2.3. Functional Requirement 3 (Use active verb + noun) **Error! Bookmark not defined.**](#_Toc474162307)

[4.3. Module 2 (Use active Verb + Noun) 6](#_Toc474162308)

[4.3.1. Functional Requirement 1 (Use active verb + noun) 6](#_Toc474162309)

[4.3.2. Functional Requirement 2 (Use active verb + noun) 8](#_Toc474162310)

[4.4. Module 3 (Use active Verb + Noun) 22](#_Toc474162311)

[4.4.1. Functional Requirement 1 (Use active verb + noun) **Error! Bookmark not defined.**](#_Toc474162312)

[4.4.2. Functional Requirement 2 (Use active verb + noun) **Error! Bookmark not defined.**](#_Toc474162313)

[5. System Model 30](#_Toc474162314)

[5.1. Level-0 DFD 30](#_Toc474162315)

[5.2. Level-1 DFD 31](#_Toc474162316)

[5.3. Level-2 DFD (Process P1: Title) 32](#_Toc474162317)

[5.4. Level-2 DFD (Process P2: Title) 33](#_Toc474162318)

[5.5. Level-2 DFD (Process P3: Title) 34](#_Toc474162319)

[6. Data Model (Class Diagram) 35](#_Toc474162320)

[6.1. Class Diagram 35](#_Toc474162321)

[6.2. Relational Schema 36](#_Toc474162322)

[6.3. Database Schema 37](#_Toc474162323)

[7. Conclusion 38](#_Toc474162324)

[8. Appendices **Error! Bookmark not defined.**](#_Toc474162325)

[8.1. Appendix A – Title **Error! Bookmark not defined.**](#_Toc474162326)

[8.2. Appendix B – Title **Error! Bookmark not defined.**](#_Toc474162327)

# Introduction/overview

This is a system for a doggy day-care company that will take in the details and make reservations for customers and they’re dogs respectively.

# Functional Components

# User Requirements

1. Doggy Motel will set the kennel for the dog
   1. Doggy Motel will add a Kennel type.
   2. Doggy Motel will add a kennel.
   3. Doggy Motel will update details of a kennel.
   4. Doggy Motel will remove a kennel for a dog.
2. Doggy Motel will take bookings.
   1. Doggy Motel will register a customer.
   2. Doggy Motel will register a dog for a customer wishes to check-in.
   3. Doggy Motel will allow a reservation to be made.
   4. Doggy Motel will allow a reservation to be cancelled.
3. Doggy Motel will perform administrative reporting.
   1. Doggy Motel will register arrivals.
   2. Doggy Motel will register departures.
   3. Doggy Motel will perform revenue analysis.
   4. Doggy Motel will perform kennel analysis.

# System Requirements

This is a System for a doggy day care system that will register customers and their dogs and will also manage the breed and the types of kennel required for the dog. The system will then register when a dog arrives and when they depart. The system will perform administrative reporting by listing a dog’s arrivals and departures, performing revenue analysis and kennel analysis.

## System Level Use Case Diagram

This diagram showcases the functions of a doggy day-care system and how those functions are carried out.

* 1. **Manage Kennels**

## Module 1 (Set Kennel)

**4.2.1 Set Kennel Type**

### Set Kennel Type

Determine the size of the kennel required.

Receptionist

<<Extends>>

<<Includes>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Set Kennel Type** | |
| **Use Case Id** | 5 | |
| **Priority** | 5 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will set the required kennel to suit the dog’s size. | |
| **Preconditions** | A kennel of the appropriate type must be available. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Set Kennel Type function.  **Step 3:** The receptionist enters the Kennel details:   * Kennel type * Description * Daily Rate | **Step 2:** The system displays the UI.  **Step 4:** The system validates the data entered:   * All fields must be entered   **Step 5:** The system savesthe in the **Kennel Types File:**   * Kennel Type * Description * Daily Rate   **Step 7:** The system displays a confirmation message.  **Step 8:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | The kennel for a dog is set. | |
| **Post conditions** | The dog will stay at the kennel determined by the function. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Add Kennel**

### Add Kennel

Add a new Kennel to the system.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Kennel** | |
| **Use Case Id** | 6 | |
| **Priority** | 6 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function will add a kennel if required. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The manager invokes the Add Kennel function.  **Step 5:** The manager enters the new Kennel details:   * KennelID * Type | **Step 2:** The system assigns a KennelID.  **Step 3:** The system retrieves kennel types from kennel Type file.  **Step 4:** The system displays the UI.  **Step 6:** The system validates the data entered:   * All fields must be entered   **Step 7:** The system sets kennel status = ‘Available’  **Step 8:** The system savesthe Kennel details in the **Kennels File.:**   * KennelID * Type * Status   **Step 9:** The system displays a confirmation message.  **Step 10:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A new Kennel is added. | |
| **Post conditions** | The dog can stay at the new kennel | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Update Kennel**

### Update Kennel

Change the status of the Kennel.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Kennel** | |
| **Use Case Id** | 7 | |
| **Priority** | 7 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will update the status of a kennel. | |
| **Preconditions** | Kennel analysis must have been performed. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Update Kennel function.  **Step 3:** The receptionist enters the Kennel details:   * KenellID   **Step 5:** The receptionist enters the status of the Kennel | **Step 2:** The system displays the UI.  **Step 4:** The system validates the data entered:   * All fields must be entered   **Step 6:** The system savesthe Kennel details in the **Kennels File:**   * KennelID * Current Status   **Step 7:** The system displays a confirmation message.  **Step 8:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A kennels status is updated. | |
| **Post conditions** | The status of a kennel is changed to suit the current state of the kennel. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Remove Kennel**

### Remove Kennel

Remove a particular kennel specified by the user.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Remove Kennel** | |
| **Use Case Id** | 8 | |
| **Priority** | 8 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will change the kennel of a dog if required. | |
| **Preconditions** | The dog must already be in a kennel. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Remove Kennel function.  **Step 3:** The receptionist enters the Dog and the kennel   * Name * Destination * Reason for being moved | **Step 2:** The system displays the UI.  **Step 4:** The system validates the data entered:   * All fields must be entered   **Step 5:** The system savesthe dog details in the **Customers File** and kennel details in the **kennels file:**   * Name * Destination * Reason for being moved   **Step 6:** The system displays a confirmation message.  **Step 7:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A dog is moved to a new kennel. | |
| **Post conditions** | The dog is now moving to a new kennel. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* 1. **Manage Customers**

## Module 2 (Make Bookings)

Outline the functions to be delivered in this component.

### Register Customer

Register a new customer.

**4.3.1 Register Customer**

Bookings can only be made for registered customers. This function registers a customer details on the system.

Customer

Receptionist

<<Includes>>

<<Extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Register Customer** | |
| **Use Case Id** | 1 | |
| **Priority** | 1 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function registers a customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Register Customer function.  **Step 4:** The receptionist enters the customer details:   * Name * Phone * Email | **Step 2:** The system assigns a CustID.  **Step 3:** The system displays the UI.  **Step 5:** The system validates the data entered:   * All fields must be entered   **Step 6:** The system savesthe customer details in the **Customers File:**   * CustID * Name * Phone * Email   **Step 7:** The system displays a confirmation message.  **Step 8:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A new customer is created. | |
| **Post conditions** | The new customer can now board their dogs at the kennels. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Register Dog**

This function registers a dog that a customer would like to check in.

### Register Dog

Register the dog to stay here.

Customer

Receptionist

<<Includes>>

<<Extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Register Dog** | |
| **Use Case Id** | 2 | |
| **Priority** | 2 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer, Dog | |
| **Description** | This function registers the dog. | |
| **Preconditions** | The customer needs to have been registered beforehand | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Register dog function.  **Step 3:** The receptionist searches for Customer by inputting CustID  **Step 5:** The receptionist enters the dog details:   * Name * Breed * Medical conditions * Gender | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves the customer details from the Customers File and displays on UI  **Step 6:** The system validates the data entered:   * **All fields must be entered** * **Breed/Medical Conditions must not be numeric**   **Step 7:** The system assigns a DogID to the dog  **Step 8:** The system savesthe details in the Dog File:   * DogID * CustId * Name * Breed * MedCond * gender   **Step 8:** The system displays a confirmation message.  **Step 9:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A new dog is registered. | |
| **Post conditions** | The new dog can now stay at the kennels. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Make Reservation**

### Make Reservation

Schedule a reservation to stay at the hotel.

Customer

Receptionist

<<Extends>>

<<Includes>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Make Reservation** | |
| **Use Case Id** | 3 | |
| **Priority** | 3 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will allow a reservation to be made. | |
| **Preconditions** | The dog and the customer need to have been registered beforehand. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Make Reservation function.  **Step 5:** The receptionist enters Surname of the customer.  **Step 7:** The receptionist selects the customer.  **Step 9:** The receptionist selects the dog for reservation.  **Step 10:** The receptionist enters the details of the reservation:   * Date of Arrival * Date of Departure * Kennel Type   **Step 13:** The receptionist selects a Kennel.  **Step 17:** The receptionist confirms the details | **Step 2:** The system assigns a ResID.  **Step 3:** The system loads Kennel types from Kennel types file.  **Step 4:** The system displays the UI.  **Step 6:** The system retrieves all customers with Surname from the Customers File and displays on UI  **Step 8:** The system retrieves all Dogs for the selected Customer from the Dogs File and displays on UI  **Step 11:** Thesystem searches for available Kennels of the right size for the required date.  **Step 12:** The system displays the available Kennels.  **Step 14:** System retrieves the rate for the selected kennel from the Kennel Types file  **Step 15:** The system calculates the cost of the reservation (No days \* Rate)  **Step 16:** The system sets the reservation status to ‘Reserved’  **Step 18:** The system saves the details in the Reservations file:   * ResID * KennelNo * DogID * Cost * Status   **Step 19:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **There are no available Kennels** |  | **Step 12:** If no kennels available, alert the manager and return to step 10 |
| **Conclusions** | A reservation is made. | |
| **Post conditions** | A reservation has been made. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Cancel Reservation**

### Cancel Reservation

Cancel a reservation made previously.

Customer

Receptionist

<<Extends>>

<<Includes>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Reservation** | |
| **Use Case Id** | 4 | |
| **Priority** | 4 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows a reservation to be cancelled. | |
| **Preconditions** | A reservation had to have already been made. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Cancel Reservation function.  **Step 3:** The receptionist enters the customer’s name.  **Step 5:** The receptionist selects which reservation which is to be cancelled  **Step 8:** The receptionist confirms that the reservation is to be cancelled. | **Step 2:** The system displays the UI.  **Step 4:** The system finds the future reservations relating to that name from the Reservations file.  **Step 6:** The system retrieves the reservation details from the reservations file for the selected reservation and displays the UI.  **Step 7:** The system displays the reservation details.  **Step 9:** The system deletes reservation from bookings file.  **Step 10:** The system displays a confirmation message.  **Step 11:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | A reservation is cancelled. | |
| **Post conditions** | The dog no longer can stay at the kennel for the agreed upon time. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* 1. **Administrative reporting**

## Module 3 (Perform Admin)

**4.4.1 Arrival**

### Register Arrival

Take note of arrival times for the dog.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Arrival** | |
| **Use Case Id** | 9 | |
| **Priority** | 9 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will register arrivals | |
| **Preconditions** | A reservation had to have been made beforehand. | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist searches for reservations meant for today.  **Step 3:** The receptionist selects the reservation meant for the customer.  **Step 4:** the receptionist confirms the arrival.  **Step 5:** The receptionist set status to ‘arrived’. | **Step 2:** The system displays all reservations with arrival date equal to today. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  | **Step 4:** Invalid data is entered.    **Step 5:** The system displays an appropriate error message. |
| **Conclusions** | The arrival time for a dog is logged. | |
| **Post conditions** | The time of arrival is registered. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Departure**

### Register Departure

Take note of departure times for the dog.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Departure** | |
| **Use Case Id** | 10 | |
| **Priority** | 10 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will register departures. | |
| **Preconditions** | The dog had to have arrived | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist enters the customer’s name.  **Step 3:** The receptionist selects the correct reservation.  **Step 4:** The receptionist sets the status of the reservation to complete. | **Step 2:** The system retrieves reservations where status is equal to ”Arrived”  **Step 5:** The system savesthe customer details in the **Customers File.**  **Step 6:** The system prints an invoice.  **Step 6:** The system displays a confirmation message.  **Step 7:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  |  |
| **Conclusions** | The departure time for a dog is logged. | |
| **Post conditions** | The time of departure is registered. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Revenue Analysis**

### Perform Revenue Analysis

Determine whether the system made a profit or a loss.

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Revenue Analysis** | |
| **Use Case Id** | 11 | |
| **Priority** | 11 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | This function will perform revenue analysis. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Revenue Analysis function. | **Step 2:** The Analyses the company’s revenue for the month  **Step 3:** The system displays the revenue analysis for the year.  **Step 4:** The system savesthe revenue analysis in the company account list.  **Step 5:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
|  |  |  |
| **Conclusions** | Revenue analysis is performed. | |
| **Post conditions** | We now know if we made a profit or loss. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

* + 1. **Kennel Analysis**

### Perform Kennel Analysis

Evaluate the condition of the Kennel

Receptionist

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Kennel Analysis** | |
| **Use Case Id** | 12 | |
| **Priority** | 12 | |
| **Source** | Receptionist | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** |  | |
| **Description** | This function will perform kennel analysis. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** The receptionist invokes the Kennel Analysis function.  **Step 3:** The receptionist enters which Kennel is to be analysed   * KennelID | **Step 2:** The system displays the UI.  **Step 4:** Analysis is carried out on the Kennel chosen.  **Step 5:** The system reports back the status of the kennel in question.  **Step 6:** The system clears the UI. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Not all fields are entered** |  |  |
| **Conclusions** | Kennel analysis is performed. | |
| **Post conditions** | The kennels are checked to see if they are up to standard. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

Reservation Details

Doggy Motel

Customer

Invoice

## Level-1 DFD

Saves to

D3

Dogs File

Customer

Rates Details

P1

Manage Kennels

Customer details

P2

Process Reservations

Kennel Details

D1

Kennel Types file

Saves to

Retrieves from Customer details

Kennel Details

D5

Customers File

D2

Kennel File

K

D4

Reservation File

P3

Process Admin

Kennel Details

## Level-2 DFD (Kennel)

P2

Add Kennel

P3

Update Kennel

Kennel details

Kennel details

D1

Kennels File

Kennel details Kennel details

P1

Set Kennel Type

P4

Remove Kennel

D3

Kennel type file

Kennel Details

Kennel Details

## Level-2 DFD (Bookings)

Register customer

Customer

P1

Register Customer

D1

Dogs File

Register dog

Dog details

P2

Register Dog

D3

Reservation File

Customer details

D2

Customers File

Dog details Reservation details

P3

Make Reservation

D4

Kennels File

Cancel Reservation

P4

Cancel Reservation

## Level-2 DFD (Admin)

Register arrivals Register departures

D1

Customer File

P2

Register Departure

P1

Register Arrival

Kennel Details

Kennel Details

Kennel details

D2

Kennels File

P3

Perform Revenue Analysis

P4

Perform Kennel Analysis

Revenue Analysis

D4

Company Account list

Kennel Details

D3

Kennel Types File

# Data Model (Class Diagram)

This is the data model for Doggy Motel. It contains the class diagram, the relational schema and the database schema.

## Class Diagram

Object Model – UML Class Diagram

Class diagram shows objects & attributes

|  |
| --- |
| Customers |
| custID:int  name:String  phone:String  email:String |

|  |
| --- |
| Dog |
| dogID:int  breed:String  medicalCondition:String  gender:char |

|  |
| --- |
| Reservations |
| ResID:int  dateOfArrival:String  dateOfDeparture:String  kennelType:String |

|  |
| --- |
| Kennel |
| kennelID:int  KennelType:String |

## Relational Schema

Relational schema for the data requirements - Using ***bracket notation***

***Customer***(CustID,Name,Phone,Email)

***Dog***(DogID,Breed,MedicalCondition,Gender)

***Reservations***(ResID,DateOfArrival,DateOfDeparture,KennelType)

***Kennel***(KennelID,KennelType)

***KennelType***(KennelID,Type,Status)

## Database Schema

A definition of the database to be implemented.

This includes primary key, foreign key and other constraints to be implemented.

**Schema:** Doggy Motel

**Relation** **Customers**

CustID numeric(5)

Name char(20) NOT NULL UNIQUE

Phone numeric(20)

Email varchar(30)

**Primary Key:** CustID

**Relation Dog**

DogID numeric(5)

Breed char(20)

MedicalCondition varchar(30)

Gender char(1)

**Primary Key:** DogID

**Relation Reservations**

ResID numeric(5)

DateOfArrival Date

DateOfDeparture Date

KennelType char(5)

**Primary Key:** ResID

**Foreign Key:** KennelType References Kennel

**Relation Kennel**

KennelID numeric(5)

KennelType char(5)

**Primary Key:** KennelID

# Conclusion

This system would give the day care company the ability to register dogs and be able to cater to they’re specific sizes individually and make reservations for them and allow the company to perform admin at the end of the month.

The hardest thing about this project would be keeping the referential integrity and making sure the system is logically sound and useful.

## Appendices

**8.1 Appendix A - Title**

**8.2 Appendix B - Title**