Requirements Engineering

Dog Kennel System

Submitted By: Joseph Diggins (t00172862)

Computing with Games Development

Date Submitted: 22/04/2022

**Table of Contents**

Contents

[1. Introduction/overview 4](#_Toc101531993)

[2. Functional Components 5](#_Toc101531994)

[3. User Requirements 6](#_Toc101531995)

[3.1. DoggyDayCare will manage accounts 6](#_Toc101531996)

[3.2. DoggyDayCare will manage services 6](#_Toc101531997)

[3.3. DoggyDayCare will manage bookings 6](#_Toc101531998)

[3.4. DoggyDayCare will perform administrative reporting 6](#_Toc101531999)

[4. System Requirements 7](#_Toc101532000)

[4.1. System Level Use Case Diagram 7](#_Toc101532001)

[4.2. Manage Accounts 8](#_Toc101532002)

[4.2.1. Open Account 8](#_Toc101532003)

[4.2.2. Update Account 10](#_Toc101532004)

[4.2.3. Register Dog 12](#_Toc101532005)

[4.3. Manage Services 14](#_Toc101532006)

[4.3.1. Add Service 14](#_Toc101532007)

[4.3.2. Withdraw Service 16](#_Toc101532008)

[4.4. Manage Bookings 18](#_Toc101532009)

[4.4.1. Make Booking 18](#_Toc101532010)

[4.4.2. Cancel Booking 22](#_Toc101532011)

[4.4.3. Add a Booking Service 24](#_Toc101532012)

[4.4.4. Check-In/Out Dog 26](#_Toc101532013)

[4.5. Perform Admin 28](#_Toc101532014)

[4.5.1. Provide Yearly Revenue Analysis 28](#_Toc101532015)

[4.5.2. Provide Service Analysis 30](#_Toc101532016)

[5. System Model 31](#_Toc101532017)

[5.1. Level-0 DFD 33](#_Toc101532018)

[5.2. Level-1 DFD 34](#_Toc101532019)

[5.3. Level-2 DFD (Process P1: Process Accounts) 35](#_Toc101532020)

[5.4. Level-2 DFD (Process P2: Process Services) 36](#_Toc101532021)

[5.5. Level-2 DFD (Process P3: Process Bookings) 37](#_Toc101532022)

[5.6. Level-2 DFD (Process P4: Process Admin) 38](#_Toc101532023)

[6. Data Model (Class Diagram) 39](#_Toc101532024)

[6.1. Class Diagram 39](#_Toc101532025)

[6.2. Relational Schema 40](#_Toc101532026)

[6.3. Database Schema 41](#_Toc101532027)

[7. Conclusion 43](#_Toc101532028)

# Introduction/overview

The Dog Kennel System allows a user (admin & manager) to perform multiple functions on the business application. An admin will have access to manage accounts, through the Add Account, Update Account and Register Dog functions, and manage bookings, using the Add Booking, Add Booking Service, Cancel Booking, Check-In/Out Dog functions, on behalf of the customer and their pets. A manager will have access to add or remove service functions to make services available or unavailable to customers. A manager will be able to perform admin to keep track of the yearly sales via the Provide Yearly Analysis function. A manager will also be able to check which months services are the most popular via the Provide Service Analysis function.

# Functional Components

# User Requirements

## DoggyDayCare will manage accounts

* + 1. DoggyDayCare will open a new account.
    2. DoggyDayCare will update an existing account.
    3. DoggyDayCare will register a dog.

## DoggyDayCare will manage services

* + 1. DoggyDayCare will add a service.
    2. DoggyDayCare will withdraw a service.

## DoggyDayCare will manage bookings

* + 1. DoggyDayCare will make a booking.
    2. DoggyDayCare will cancel a booking.
    3. DoggyDayCare will add a booking service.
    4. DoggyDayCare will check-in or check-out a dog.

## DoggyDayCare will perform administrative reporting

* + 1. DoggyDayCare will provide yearly revenue analysis.
    2. DoggyDayCare will provide service analysis.

# System Requirements

## System Level Use Case Diagram

The following system level use case diagram illustrates the high-level system requirements.

Manager

Customer

Admin

## Manage Accounts

This module provides function to open an account, update an account and register a dog.

### Open Account

This function opens an account for a potential customer. The customer details are recorded in the system. A customer is identified by a unique CustID.

Customer

Admin

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Open Account** | |
| **Use Case Id** | DKS001 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function opens an account for a customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the Open Account function.  **Step 3:** The admin enters the required data:   * FirstName (Char 15) * LastName (Char 15) * DOB (Date) * Street (Char 25) * Town (Char 20) * County (Char 9) * EirCode (Char 7) * Phone (Char 10) * Email (Char 40) | **Step 2:** The system displays the UI.  **Step 4:** System validates the data:   * All fields must be entered * Name & address must not be numeric * DOB must be in format DD/MM/YYYY & not in the future. * Min age for DOB is 16. * EirCode must be valid format. * Phone number must have 10 digits & unique. * Email must be valid format and unique   **Step 5:** Generate & Assign next CustID.  **Step 6:** The system saves the Account details in the Accounts file:   * CustID (Numeric) * FirstName (Char 15) * LastName (Char 15) * DOB (Date) * Street (Char 25) * Town (Char 20) * County (Char 9) * EirCode (Char 7) * Phone (Char 10) * Email (Char 40)   **Step 7:** The system displays a confirmation message.  **Step 8:** The system resets the UI. |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **Invalid Data Entered** |  | **Step 4:** The system detects invalid data  **Step 5:** Display an appropriate error message  **Step 6:** return to step 3. |
| **Conclusions** | An account has been opened for the customer | |
| **Post conditions** | A dog can now be registered under the customers ID. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Update Account

This function allows for the customer to update their details on their current profile.

Customer

Admin

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Account** | |
| **Use Case Id** | DKS002 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows for the customers details to be updated under their existing customer ID. | |
| **Preconditions** | The customer has a customer ID already in the file. | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the update account function.  **Step 3:** Admin searches for customers using Surname (Or Part Of).  **Step 5:** Admin selects the required customer.  **Step 7:** Adminupdates the data:   * FirstName (Char 15) * LastName (Char 15) * DOB (Date) * Street (Char 25) * Town (Char 20) * County (Char 9) * EirCode (Char 7) * Phone (Char 10) * Email (Char 40) | **Step 2:** The system displays a UI.  **Step 4:** The system retrieves matching customers in the Accounts file & displays results on a UI.  **Step 6:** The system retrieves the full customer details of the selected customer from the Accounts file & loads a UI for updating.  **Step 8:** The system validates the data.   * FirstName (Char 15) * LastName (Char 15) * DOB (Date) * Street (Char 25) * Town (Char 20) * County (Char 9) * EirCode (Char 7) * Phone (Char 10) * Email (Char 40)   **Step 9:** The system updates the data in the Accounts file.   * FirstName (Char 15) * LastName (Char 15) * DOB (Date) * Street (Char 25) * Town (Char 20) * County (Char 9) * EirCode (Char 7) * Phone (Char 10) * Email (Char 40)   **Step 10:** The system displays a confirmation message.  **Step 11:** The system resets the UI. |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **Invalid Data Entered** |  | **Step 8:** The system detects invalid data.  **Step 9:** The system displays an appropriate error message.  **Step 10:** Return to step 7. |
| **Conclusions** | The customer details have been updated in their profile. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Register Dog

This function allows the customer to register their Dogs under their customer ID.

Customer

Admin

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Register Dog** | |
| **Use Case Id** | DKS003 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows for a dog to be registered under a customers existing customer ID. | |
| **Preconditions** | The customer has an existing customer ID in the Accounts file. | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the Register Dog function.  **Step 3:** Admin searches for customers using Surname (Or Part Of).  **Step 5:** Admin selects the required customer.  **Step 7:** Admin enters the required data:   * PetName (Char 15) * DOB (Date) * Breed (Char 20) * PetSize (Char 1) * Colour (Char 20) * Sex (Char 1) * Neutered (Char 1) | **Step 2:** The system displays a UI.  **Step 4:** The system checks for matching customers in the Accounts file & displays results from the search.  **Step 6:** The system displays a UI for the dog’s details to be input.  **Step 8:** System validates the data:   * All fields must be entered * Name, Breed & Colour must not be numeric * DOB must be in format DD/MM/YYYY & not in the future.   **Step 9:** Generate & Assign next PetID.  **Step 10:** The system saves the dog’s details in the Pets file:   * PetID (Numeric) * PetName (Char 15) * DOB (Date) * Breed (Char 20) * PetSize (Char 1) * Colour (Char 20) * Sex (Char 1) * Neutered (Char 1) * CustID (Numeric)   **Step 11:** The system displays a confirmation message.  **Step 12:** The system resets the UI. |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **Invalid Data Entered** |  | **Step 8:** The system detects invalid data.  **Step 9:** The system displays an appropriate error message.  **Step 10:** Return to step 7. |
| **Conclusions** | The dog’s details have now been registered in the pets file under the customers ID. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Services

This module provides function to add and withdraw services to the system.

### Add Service

This function adds services to the system. This will set charges/rates to all services the shop offers.

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Service** | |
| **Use Case Id** | DKS004 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function adds a service to the system. This will set a flat rate to all services the shop offers. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager invokes the Add Service function.  **Step 3:** The manager enters the required data:   * Service\_Code (Char 3) * Description (Char 50) * Rate (Decimal) | **Step 2:** The system displays the UI.  **Step 4:** System validates the data:   * All fields must be entered * Service\_Code must be 3 char string * Description must not be numeric * Rate must be numeric (decimal) and greater than zero * Service\_Code must not already exist   **Step 5:** The system saves the service detail in the Services file:   * Service\_Code (Char 3) * Description (Char 50) * Rate (Decimal) * Status (Char 1)   **Step 6:** The system displays a confirmation message.  **Step 7:** The system resets the UI. |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Invalid Data Entered** |  | **Step 4:** The system detects invalid data  **Step 5:** Display an appropriate error message  **Step 6:** return to step 3. |
| **Conclusions** | The service has been recorded in the database. | |
| **Post conditions** | The customer may now avail of this service for their dogs. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Withdraw Service

This function withdraws services from the system. This will take them off the system & the customers can no longer avail of them.

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Withdraw Service** | |
| **Use Case Id** | DKS005 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function withdraws services from the system. This will remove them from the system & the customers can no longer avail of them | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager invokes the Withdraw Service function  **Step 3:** The manager enters the service he wishes to withdraw using the required data:   * Service\_Code (Char 3) | **Step 2:** The system displays the UI.  **Step 4:** The system searches and retrieves Service details from the services file and displays a UI for viewing only.  **Step 5:** The system updates the status of the service to ‘W’ in the Services file.  **Step 6:** The system displays a confirmation message.  **Step 7:** The system resets the UI |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **No Results Found** |  | **Step 4:** There are no services which match the search  **Step 5:** Display an appropriate error message.  **Step 6:** return to step 3. |
| **Conclusions** | The service will no longer appear in the Add Booking Service section | |
| **Post conditions** | The customers will no longer be able to avail of this service. | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Bookings

This module provides function to make bookings, cancel bookings, add services to a booking and check-out a dog.

### Make Booking

This function allows for a customer to book a stay for their dog.

**Make Booking**

**ADMIN SYSTEM**

Display UI

Start Make Booking

Enter Surname

Retrieve summary of matching customers & display UI

[N]

[Y]

None?

Select Customer

Display Message

Retrieve pet details

Display UI

[N]

[Y]

Pets?

Select Pet

Find available Kennels for Dates & Dog Size

Enter Dates Required

Available?

[Y]

[N]

Display Message

Select Kennel

Assign Booking ID

Confirm Kennel

Calculate Cost of Stay

Confirm Booking

Display UI

Save Booking Details

Display Confirmation & Email Customer

Reset UI

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Make Booking** | |
| **Use Case Id** | DKS006 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows a customer to make a booking for their dog. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the Make Booking function.  **Step 3:** Admin searches for customers using Surname (Or Part Of).  **Step 5:** Admin selects the required customer.  **Step 7:** Admin selects dog they wish to make booking for.  **Step 8:** Admin enters the details of the booking into the system:   * Arrival\_Date * Departure\_Date | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves matching customers in the customers file & displays results on a UI.  **Step 6:** The system retrieves customers pet(s) from pets file & displays UI.  **Step 9:** The system validates the data:   * Arrival\_Date & Departure\_Date must be dates & not in the past.   **Step 10:** System retrieves bookings from booking file and checks if Kennels are available on the selected dates.  **Step 11:**  Assign BookingID & KennelNo.  **Step 12:** Calculate total cost of stay.  **Step 13:** The system saves the booking details in the bookings file:   * BookingID (Numeric) * KennelNo.(Numeric) * Arrival\_Date (Date) * Dept\_Date (Date) * Cost (Decimal) * PetID (Numeric) * Status (Char 1)   **Step 14:** The system displays a confirmation message.  **Step 15:** The system resets the UI. |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **Chosen Dates Unavailable** |  | **Step 10:** The dates selected are unavailable  **Step 11:** Display an appropriate error message.  **Step 12:** return to step 8. |
| **Conclusions** | The booking has been made & saved in the bookings file. | |
| **Post conditions** | Customer is now able to cancel or add services to their booking | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Cancel Booking

This function allows for a customer to cancel an existing booking for their dog.

Customer

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Booking** | |
| **Use Case Id** | DKS007 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows a customer to cancel an existing booking for their dog. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the Cancel Booking function.  **Step 3:** Admin searches for customers using Surname (Or Part Of).  **Step 5:** Admin selects the required booking. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves current bookings in the bookings file & displays results from the search.  **Step 6:** The system removes the booking from bookings file:   * BookingID   **Step 7:** The system displays a confirmation message  **Step 8:** The system resets the UI |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **No Matches Found** |  | **Step 4:** The system did not retrieve any matches.  **Step 5:** Display an appropriate error message.  **Step 6:** return to step 3. |
| **Conclusions** | The booking has now been removed from the system | |
| **Post conditions** |  | |
| **Business Rules** | Cannot be cancelled if current date is Arrival\_Date or later. | |
| **Implementation Constraints** |  | |

### Add a Booking Service

This function allows a customer to cancel an existing booking for their dog.

Customer

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Booking Service** | |
| **Use Case Id** | DKS008 | |
|  |  | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows a customer to add a service to an existing booking for their dog. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** Admin invokes the Add Booking Service function.  **Step 3:** Admin searches for customers using Surname (Or Part Of).  **Step 5:** Admin selects the required booking.  **Step 7:** Admin selects the service to add to the booking. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves matching bookings in the bookings file & displays results from the search.  **Step 6:** The system retrieves services from the services file that are currently available & displays them on a UI.  **Step 8:** The system saves the booking service details into the booking services file:   * BookingID (Numeric) * Service\_Code (Char 3) * Cost (Decimal)   **Step 9:** The system displays a confirmation message  **Step 10:** The system resets the UI |
| **Alternate Scenarios** | **Admin** | **System Response** |
| **No Matches Found** |  | **Step 4:** The system did not retrieve any matches.  **Step 5:** Display an appropriate error message.  **Step 6:** return to step 3. |
| **Conclusions** | The service has now been added to the booking & rate applied to the account. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Check-In/Out Dog

This function allows for a dog to be checked out.

Customer

Admin

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check-In/Out Dog** | |
| **Use Case Id** | DKS009 | |
| **Priority** | 1 | |
| **Source** | Admin | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** | Customer | |
| **Description** | This function allows a dog to be checked in. | |
| **Preconditions** | They must have a booking in the bookings file. | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
| **Check In Radio Button** | **Step 1:** Admin invokes the Check-In/Out Dog function.  **Step 3:**  Admin selects the check in radio button.  **Step 5:** Admin selects the required booking to check in. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves dogs due to arrive today from the bookings file and displays them on the UI.  **Step 6:** The system updates the status of the booking in the bookings file.  **Step 7:** The system displays a confirmation message  **Step 8:** The system resets the UI |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Check Out Radio Button** | **Step 1:** Admin invokes the Check-In/Out Dog function.  **Step 3:**  Admin selects the check out radio button.  **Step 5:** Admin selects the required booking to check out. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves dogs due to depart today from the bookings file and displays them on the UI.  **Step 6:** The system updates the status of the booking in the bookings file.  **Step 7:** The system calculates the total amount due.  **Step 8:** The system displays a confirmation message with the total amount.  **Step 8:** The system resets the UI |
| **Conclusions** | The dog has been checked in/out & the kennel status has been updated. If checking out the amount due is calculated and displayed. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Perform Admin

This module provides function to provide yearly revenue analysis & yearly service analysis.

### Provide Yearly Revenue Analysis

This function provides all the revenue analysis from the past 12 months.

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Provide Yearly Revenue Analysis** | |
| **Use Case Id** | DKS010 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function provides a revenue analysis for all the bookings made in the past 12 months. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager invokes the Provide Yearly Revenue Analysis function.  **Step 3:** Manager selects the year they wish to view. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves the sales details from the sales file and displays the results. |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **No Matches Found** |  | **Step 4:** There are no details available for the selected year  **Step 5:** Display an error message.  **Step 6:** Return to Step 3. |
| **Conclusions** | The yearly revenue analysis was provided on screen for the manager. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Provide Service Analysis

This function provides all the revenue analysis from services added to bookings.

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Provide Service Analysis** | |
| **Use Case Id** | DKS011 | |
| **Priority** | 1 | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** |  | |
| **Description** | This function provides a revenue analysis for all the services added to bookings. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Manager** | **System** |
|  | **Step 1:** Manager invokes the Provide Yearly Revenue Analysis function.  **Step 3:** Manager selects the year they wish to view. | **Step 2:** The system displays the UI.  **Step 4:** The system retrieves the service sales details from the sales file and displays the results. |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **No Matches Found** |  | **Step 4:** There are no details available for the selected year  **Step 5:** Display an error message.  **Step 6:** Return to Step 3. |
| **Conclusions** | The yearly revenue analysis for services added to bookings was provided on screen for the manager. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

The following dataflow diagrams have been produced for the system:

**Data Flow Diagram – Elements**

***External Entities:***

Customer

***Data Stores:***

D1: Accounts File

D2: Pets File

D3: Services File

D4: Bookings File

D5: Kennels File

D6: Booking Services File

D7: Sales File

***Processes***

P1: Process Accounts

P1.1: Open Account

P1.2: Update Account

P1.3: Register Dog

P2: Process Services

P2.1: Add Services

P2.2: Remove Services

P3: Process Bookings

P3.1: Make Booking

P3.2: Cancel Booking

P3.3: Add Booking Service

P3.4: Check-In/Out

P4: Process Admin

P4.1: Provide Yearly Revenue Analysis

P4.2: Provide Service Analysis

## Level-0 DFD

Customer

Invoice

Booking Details

Doggy Kennel System

## Level-1 DFD

D1

Accounts File

Customer

Customer Details

Customer Details

P1

Process Accounts

Bookings Confirmation

Bookings Details

Pets Details

P3

Process Bookings

D2

Pets File

Status

D5

Kennels File

Bookings Details

Bookings Details

P2

Process Services

D4

Bookings File

D6

Booking Services File

D7

Sales File

Sale Details

Service Details

D3

Services File

P4

Process Admin

Service Details

## Level-2 DFD (Process P1: Process Accounts)

Customer

Customer Details

Customer Details

P1.2

Update Accounts

P1.1

Open Account

Customer Details

Customer Details

Customer Details

D1

Accounts File

Customer Details

P1.3

Register Dog

Pet Details

Pet Details

D2

Pets File

Customer

## Level-2 DFD (Process P2: Process Services)

P2.1

Add Service

Service Details

D3

Services File

Service Details

P2.2

Remove Service

Service Details

## Level-2 DFD (Process P3: Process Bookings)

Booking Details

Booking Details

Account Details & Pet Details

D6

Booking Services File

Booking Service Details

Booking Details

Booking Details

Pet Details

Account Details

Booking Details

Kennel Status

P3.4

Check-In/Out Dog

D4

Bookings File

D2

Pets File

D1

Accounts File

Customer

Booking Details

P3.2

Cancel Booking

Booking Details

Customer

Service Details

D3

Services File

P3.1

Make Booking

P3.3

Add Booking Service

D5

Kennels File

## Level-2 DFD (Process P4: Process Admin)

D7

Sales File

Total Cost

Services Cost

P4.2

Provide Service Analysis

P4.1

Provide Yearly Revenue Analysis

# Data Model (Class Diagram)

Below is a class diagram for Doggy Day Care showing the system’s classes (Customers, Pets, Bookings, Kennels & Services), the attributes for each of the classes and the associations between the classes.

## Class Diagram

Customers

Pets

PetID\*: int

Name: String

DOB: Date

Breed: String

Size: Char

Colour: String

Sex: Char

Neutered: Char

CustID\*: int

FirstName: String

LastName: String

DOB: Date

Street: String

Town: String

County: String

Eircode: String

Phone: String

Email: String

1

1..\*

Has a

1

Is for

0..\*

Booking\_Service

Bookings

BookingID\*: int

Arrival\_Date: Date

Dept\_Date: Date

Booking\_Service: String

Total\_Cost: Decimal

Has a

0..\*

1

Cost: Decimal

Kennels

0..\*

KennelNo\*: int

Size: Char

Cost: Decimal

Status: Char

1

0..\*

Is for

Has a

1

Services

Service\_Code\*:String

Description: String

Rate: Decimal

## Relational Schema

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

Accounts (CustID, FirstName, LastName, DOB, Street, Town, County, Eircode, Phone, Email)

Pets (PetID, Name, DOB, Breed, Size, Colour, Sex, Neutered, CustID)

Bookings (BookingID, KennelNo, Arrival\_Date, Dept\_Date, Total\_Cost, PetID)

Booking\_Services ( BookingID, Service\_Code, Cost)

Kennels (KennelNo, Size, Cost, Status)

Service (Service\_Code, Description, Rate)

Sales (SaleID, SaleDate, Total\_Amount, Services\_Amount)

## Database Schema

**Schema**: DogKennelSys

**Relation** *Accounts*

CustID int (4) UNIQUE

FirstName String (15) NOT NULL

LastName String (15) NOT NULL

DOB Date

Street String (15) NOT NULL

Town String (15) NOT NULL

County String (9) NOT NULL

Eircode String (6) NOT NULL

Phone String (10) UNIQUE, NOT NULL

Email String (40) UNIQUE, NOT NULL

**Primary Key:** CustID

**Relation** *Pets*

PetID int (4) UNIQUE

Name String (15) NOT NULL

DOB Date

Breed String (15) NOT NULL

Size Char (1) NOT NULL

Colour String (15) NOT NULL

Sex Char (1) NOT NULL

Neutered Char (1) NOT NULL

CustID int (4) UNIQUE

**Primary Key:** PetID

**[Foreign Key:** CustID **References** Accounts]

**Relation** *Kennels*

KennelNo int (3) UNIQUE

Size Char (1) NOT NULL

Cost Decimal NOT NULL

Status Char (1) NOT NULL

**Primary Key:** KennelNo

**Relation** *Bookings*

BookingID int (3) UNIQUE

KennelNo int (3) UNIQUE

Arrival\_Date Date NOT NULL

Dept\_Date Date NOT NULL

Total\_Cost Decimal NOT NULL

PetID int (4) UNIQUE

**Primary Key:** BookingID

**[Foreign Key:** KennelNo **References** Kennels]

**[Foreign Key:** PetID **References** Pets]

**Relation** *Services*

Service\_Code String (3) UNIQUE

Description String (40) NOT NULL

Rate Decimal NOT NULL

**Primary Key:** Service\_Code

**Relation** *Booking\_Services*

BookingID int (3) UNIQUE

Service\_Code String (3) NOT NULL

Cost Decimal NOT NULL

**Primary Key:** BookingID, Service\_Code

**Relation** *Sales*

SaleID int (3) UNIQUE

Sale\_Date Date

Total\_Amount Decimal NOT NULL

Services\_Amount Decimal

**Primary Key:** SaleID

# Conclusion

In conclusion, The Dog Kennel System allows a user (admin & manager) to perform multiple functions on the business application. An admin has access to manage accounts, through the Add Account, Update Account and Register Dog functions, and manage bookings, using the Add Booking, Add Booking Service, Cancel Booking, Check-In/Out Dog functions, on behalf of the customer and their pets. A manager has access to add or remove service functions to make services available or unavailable to customers. A manager can perform admin to keep track of the yearly sales via the Provide Yearly Analysis function. A manager can also check which months services are the most popular via the Provide Service Analysis function.