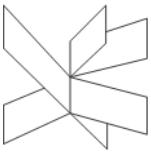


Software System for VIAPets – Project Report

By Daniel Weldesilasie (354818), Jwan Hasan (3545289), Lea Maj Gazel (355157), Shishir Gajurel (355402) and Youssef Topaji (355437)

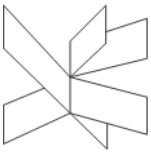
Allan Henriksen, Jacob M. Wang and Line Linhardt Egegaard

Number of characters : 30276

Dep. Of Software Technology and Engineering

Semester-1

20.12.2024



Abstract

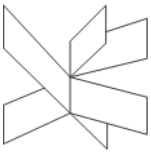
VIAPets, owned by Bob Oldenuff, operates as a combined pet shop and kennel. The business experienced a significant increase in the pet sales during the COVID-19 pandemic as people sought companionship while staying at home. However, as life normalized and people became busier, Bob recognized the need to expand his services by opening a kennel.

The growth of VIAPets has introduced operational challenges, including the need to manage detailed pet information, kennel bookings, and avoid costly errors, such as mistakenly selling pets owned by customers. To address these challenges and ensure efficient operations, the project aims to develop software system based on the needs of VIAPets.

The software system will enable employees to:

- *Reserve kennels and sell pets while preventing mix-ups between kennel pets and pets for sale.*
- *Add, edit, and delete customer, reservations and pet information within the system.*
- *Provide a reliable overview of kennel and pet availability through a website, allowing customers to view this information online. However, all purchases and reservations must still be made in person at the shop.*

The solution will be built using Java, incorporating Scene Builder to design and implement the graphical user interface (GUI). Mathematical concepts such as time complexity and Big-O notation, learned in Mathematics for Software Engineering (MSE), will guide code optimization. For the website, we used HTML, CSS, JavaScript, jQuery, and Bootstrap to deliver an intuitive and visually appealing interface.



By implementing this software system, VIAPets will have a functional solution to effectively manage its pet shop and kennel operations. The system will support employee workflows, improve accuracy, and provide customers with greater transparency into available pets and kennel bookings. This will ensure that VIAPets continues to differentiate itself in the competitive pet shop and kennel market.

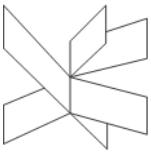
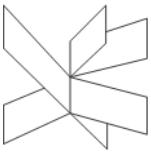


Table of content



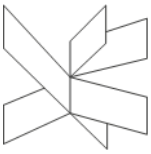
1. Introduction

Efficient business operations require well-designed management systems to handle data accuracy, streamline processes, and support growth (Hernaus, Bach, & Vukšic, 2012). For VIAPets, a pet shop and kennel owned by Mr. Bob, these needs are even more pronounced due to the dual nature of its services and the challenges of maintaining clear distinctions between pets for sale and those in the kennel. In response to a major operational mistakes, the "Chihuahua catastrophe", VIAPets identified the necessity of implementing a software solution to prevent future errors.

VIAPets began as a pet shop offering dogs, cats, birds, fish, and rodents, with each pet having detailed records such as breed, age, and other specific characteristics. Unique to the shop is its "recycling" service, where pets returned by owners are resold to ensure they find suitable homes. The COVID-19 pandemic led to a surge in pet adoptions, and as people returned to their busy lives post-pandemic, the demand for kennel services grew. Mr. Bob responded by adding a kennel service, offering peace of mind to customers leaving their pets in his care.

This expansion, while beneficial, introduced complexities that exposed the limitations of the current system. Key challenges include maintaining separate records for pets for sale and kennel pets, tracking kennel bookings, and handling payments. As noted by Martin (2002), systems managing dual operations require object-oriented designs to ensure clarity and reduce errors. In VIAPets' case, the absence of integrated systems resulted in the mismanagement of pet records, creating operational inefficiencies and potential reputational damage.

To address these challenges, the project aimed to develop a software solution based on VIAPets' needs. Leveraging insights from Gaddis (2015) on Java programming and Brooks (2024) on software engineering principles, the solution integrates pet and customer data into a unified system. It simplifies kennel booking management, tracks recycled pets, and ensures accurate differentiation between kennel pets and pets for



sale. The adoption of local file storage, as recommended by IBM (2024), aligns with Mr. Bob's preference for offline systems.

The report follows a structured methodology, including requirements elicitation (GeeksforGeeks, 2024), UML diagrams for system design (Martin, 2002), and Agile principles for iterative development (Martin, 2014). This systematic approach ensures the final product is both functional and user-friendly, meeting VIAPets' operational requirements while addressing the key problem areas.

In the following sections, we detail the problem domain, the design process, and the implementation of the software system. This comprehensive report highlights how a well-executed software solution can transform VIAPets into a model of efficiency and reliability, setting it apart from competitors in the pet care market.

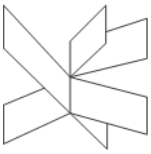
2. Main section

2.1 Analysis

2.1.1 Requirement

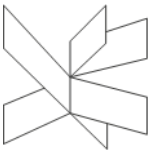
The existing system at VIAPets, while functional, has shown critical weaknesses, as demonstrated by the "Chihuahua incident." This incident highlighted the risks and inefficiencies inherent in the current system, especially as Bob's pet shop and kennel operations expand. Concerned about future issues, Bob reached out to propose the development of a new system. After an in-depth interview, we gathered insights into his needs and expectations.

The goal of the new system is to prevent similar errors and provide an efficient platform for managing both the pet shop and kennel. The following functional and non-functional requirements were derived from the discussion with Bob to ensure the system meets his expectations and business demands.



Functional requirements:

1. As an employee, I want to be able to register a pet in the system, including information about the pet's name, age, gender, species, color, so I can easily search for the pet in the system.
   Additionally, I want to be able to add comments about the pet:
   a. For dogs and cats, their breed and breeder information
   b. For rodents, behavior (if they bite)
   c. For fish, whether they live in salt water or fresh water. Whether they are predators or prey.
   d. For birds, special dietary needs.
   e. For all, comment like "under observation" (for health issues or behavior), so all staff are aware of which animals need special attention.
2. As an employee, when registering a pet for sale, I want to be able to add a price for it, so it is easily recognizable whether it is a pet shop pet or an animal currently staying in the kennel.
3. As an employee, I want to be able to change the price, in case I want to offer a discount to the customer.
4. As an employee, I want to have a system that separates the pets in the pet shop from the pets in the kennel, so we avoid another chihuahua catastrophe.
5. As an employee, I want to be able to remove/delete a booking from the system in case the customer wishes to cancel.
6. As an employee, I want to be able to modify an existing reservation in case the customer wishes to change the date or time.



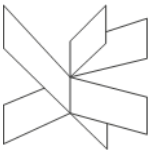
7. As an employee, I want to be able to input the date in the system to check the availability of rooms in the kennel during that period.
8. As an employee, if there are any available rooms, I want to be able to add a new booking to the system, so the availability will be updated.
9. As an employee, I want to be able to make a reservation only if there is open availability.
10. As an employee, I want to only be able to register a maximum of ten bookings simultaneously, so we ensure that there is open availability.
11. As an employee, I want to only be able to change an existing reservation to another date, if there is open availability on the day the customer wishes to change to.

12. As an employee, I want to have a record of each pet's owner, so I can reference this information if necessary.

13. As an employee, I want to be able to modify existing information about the pet in case the pet gets a new owner.

14. As an employee, I want to be able to remove/delete a pet from the system in case the pet passes away.

15. As an employee, I want to be able to register a new customer in the system by registering the customer's name, email address and phone number, so I can easily search for them in the system and contact them, if necessary.
16. As an employee, I want to be able to modify an existing customer's information, in case they change their name, email address and/or phone number.



17. As an employee, I want to be able to remove/delete a customer's information from the system in case they do not wish for the shop to store their information anymore.
18. As an employee, I want to be able to link a pet registered in the system to a customer, newly registered or already registered in the system, so I can keep track of who the owner of the pet I have sold is.
19. As an employee, I want to be able to add new pets to the pet shop list of pets (adding a new pet for recycling).
20. As an employee, I want to be able to display information on a website including photos, information about pets for sale and room availability, so the customer can see what we offer.

21. As a customer I want to be able to see the availability of the rooms in the kennel through a website so I can easily know if there is room for my pet in the desired period.

22. As a customer I want to be able to view a list of all available pets with their prices, so I can decide which one I'd like to buy.

Non-functional requirements:

1. As an employee, I want to be able to store the data in files, and not in a database system.
2. As an employee, I want the website to update the information automatically, as I update the application.
3. Every update in the system includes writing to a file.

3.2.1 Use case diagram:

The Use case diagram for the VIAPets system illustrates the various functionalities and interactions between the actors (primarily employees and customers) and the system. A use case represents a specific interaction between an actor and the system, detailing the actions they can perform. This diagram highlights the main processes and workflows within the system, making it easier to understand how users interact with the system and what actions they can execute.

The system is designed to streamline the operations of a pet shop and kennel by providing different functionalities. These functionalities include pet registration, managing pet data, making and managing reservations, adding and modifying customer information, making sales by the employee and displaying information on a website to customers.

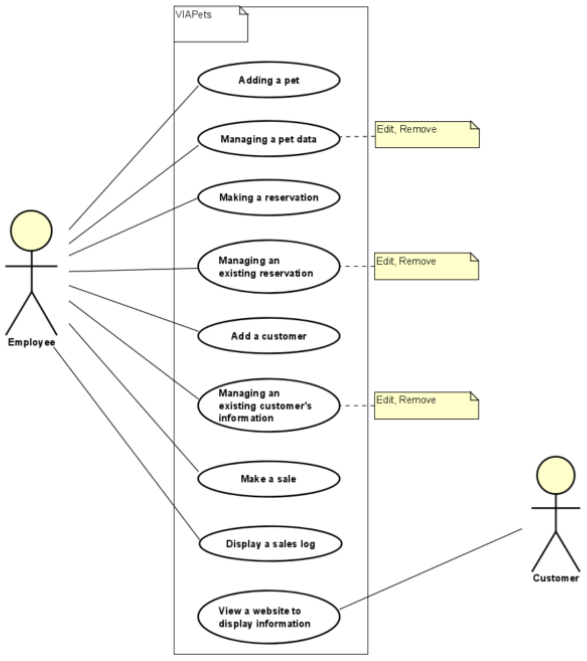
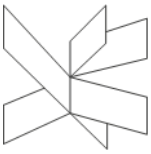


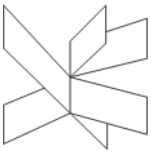
Figure 1 Use case diagram



3.2.2 Link table between requirements and use cases

Table 1 link table

Requirements	Use cases
1, 2, 3, 4, 19	Adding a pet
12, 14, 18	Managing a pet data
5, 7, 8, 9, 10,	Make a Reservation
5, 6, 7, 11	Managing an existing reservation
15	Add a customer
16, 17	Managing an existing customer's information
3, 13	Make a Sales Log
12	Display a Sales Log
20, 21, 22	View a website to display information

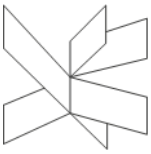


3.2.3 Use case descriptions:

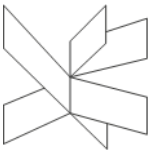
Use case descriptions provide a clear understanding of system operations by outlining the necessary preconditions and expected postconditions for each action. For instance, when making a reservation, the system must be operational, space must be available in the kennel (with a maximum capacity of ten pets), and the customer and pet data must already exist in the system. Once these conditions are satisfied, a new reservation is created and stored in the system files, kennel availability is updated, and the system automatically reflects the updated availability. Additionally, all validation rules must be met to ensure the process is completed successfully. These descriptions ensure the system operates efficiently and reliably.

Table 2 use case

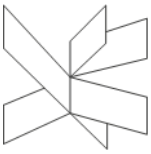
Use case	**Making a reservation**
Summary	Create a new kennel reservation for a customer's pet
Actor	Employee
Precondition	• System is operational • Space is available in the kennel (max 10 pets) • Customer and pet data exist in system



Postcondition	• New reservation is created and stored in system files • Kennel availability is updated • Website availability information is automatically updated • All validation rules are satisfied

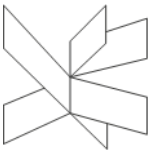


Base sequence	1. System shows current kennel availability. 2. Employee enters desired reservation period [ALT1] 3. System displays availability for specified period [ALT2] 4. Employee searches for customer [ALT3] 5. System displays customer's registered pets [ALT4] 6. Employee selects pet for boarding [ALT5] 7. System validates the booking: • Pet registration status • Existing reservations 8. Employee confirms booking details 9. System creates reservation and stores in files [ALT6] 10. System generates confirmation and updates availability display

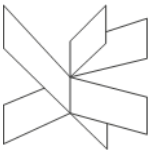


Scenario A. **NEW CUSTOMER**	A.1. Employee selects "Register New Customer" A.2. Employee enters customer details: Name, Email, Phone number A.3. System validates contact information A.4. System saves customer data A.5. Return to base sequence step 5

Scenario B. **NEW PET**	B.1. Employee selects "Register New Pet" B.2. Employee enters pet details following registration requirements B.3. System validates pet data B.4. System links pet to customer B.5. Return to base sequence step 6



Alternative sequence (branch for exception)	- [*ALT0]: Process can be cancelled at any step. Use case ends. [ALT1]: If dates invalid: 1. System shows error message 2. System highlights valid date range 3. Return to step [ALT2]: If no space available: 1. System shows next available dates 2. Employee can choose alternate dates or end use case [ALT3]: Customer search options: 1. Search by name 2. Search by phone 3. Search by email 4. If not found, go to Scenario A [ALT4]: If customer has no registered pets: 1. System shows warning 2. Option to go to Scenario B [ALT5]: System validates pet status: 1. Checks if pet is already booked 2. Checks for health flags/special needs



	[ALT6]: File storage validation: 1. Verify write success 2. Create backup 3. Update website data
Note	Covers requirements 7, 8, 9, 10, 15, 18

Please check Appendix B for the rest of use case descriptions.

3.2.4 Activity diagram:

An activity diagram illustrates the flow of actions and decisions from the start of an activity to its completion. It provides a clear visual representation of processes, showing how tasks progress, decisions are made, and outcomes are reached within a system.

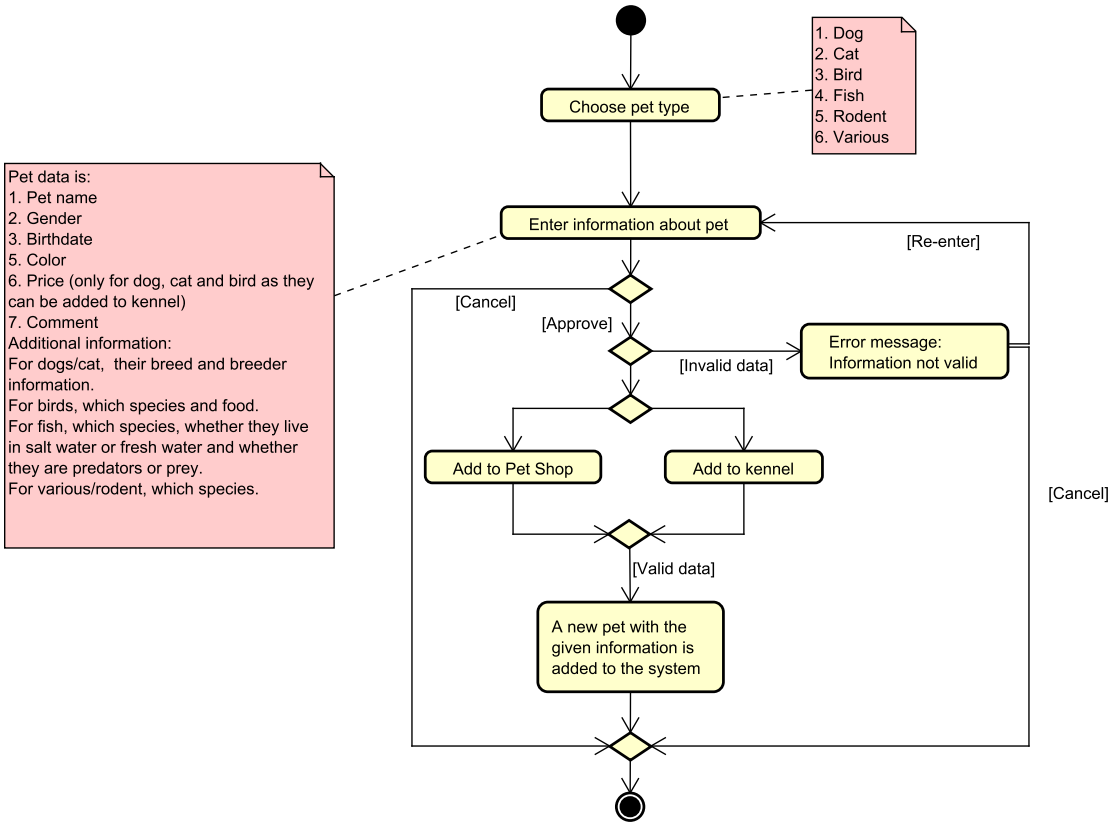


Figure 2 activity diagram for add pet

The activity diagram for the Add Pet use case in VIAPets illustrates the flow of actions required to successfully add a pet to the system. The process begins with entering the pet's basic information, including name, type, age, gender, color and price. For certain types of pets, additional requirements are necessary; for instance, adding a fish requires specifying its water living condition and whether it is a predator. If the employee confirms the pet to addition, the process advances to a checkpoint where the system validates the provided information. If the information is invalid, an error message is displayed, prompting the employee to re-enter the data. After successful validation, a decision node determines whether the pet should be added to the pet shop or the kennel business. Once the decision is made, the system sends a confirmation message. Upon approval, the pet is added to the designated section, completing the process.

3.2.5 Domain model:

The domain model illustrates the relationships between the classes within the VIAPets system. For instance, specific pet types such as fish, rodents, birds, dogs, cats, and others inherit from the Pet class. Each Pet class is associated with a single birthdate, represented by the MyDate class. In contrast, the Reservation class is linked to two instances of the MyDate class, representing the start and end dates of the reservation.

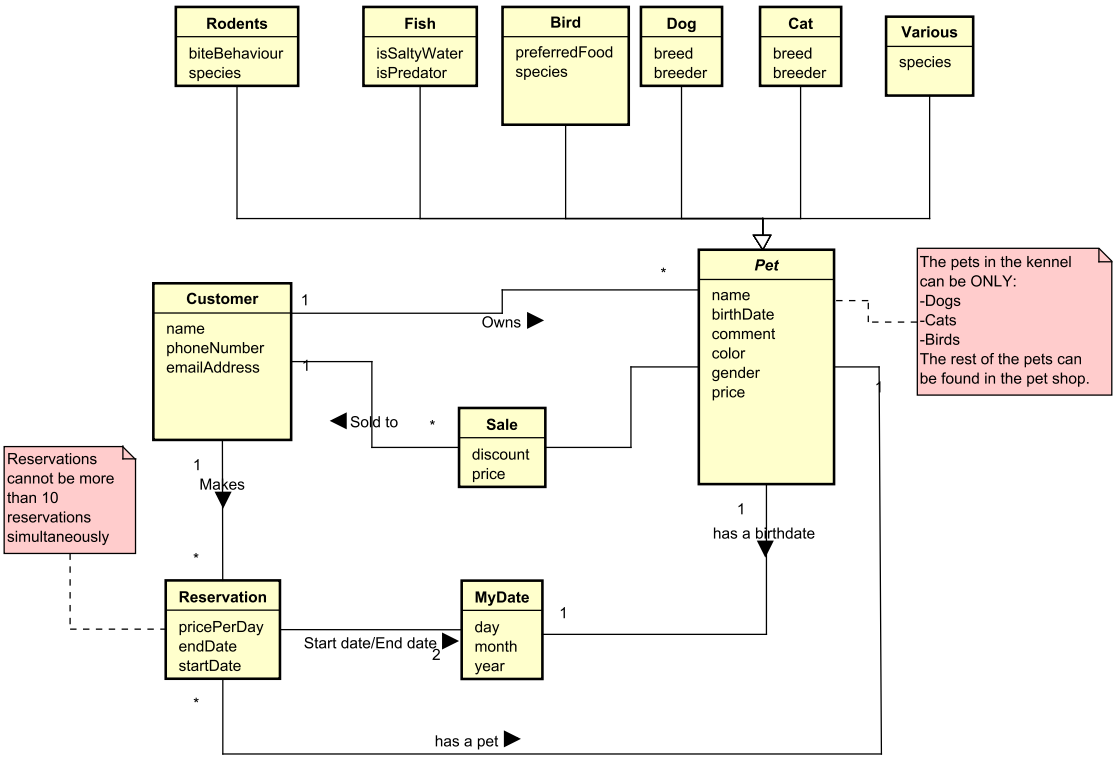
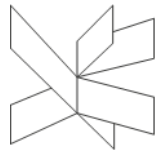


Figure 3 domain model

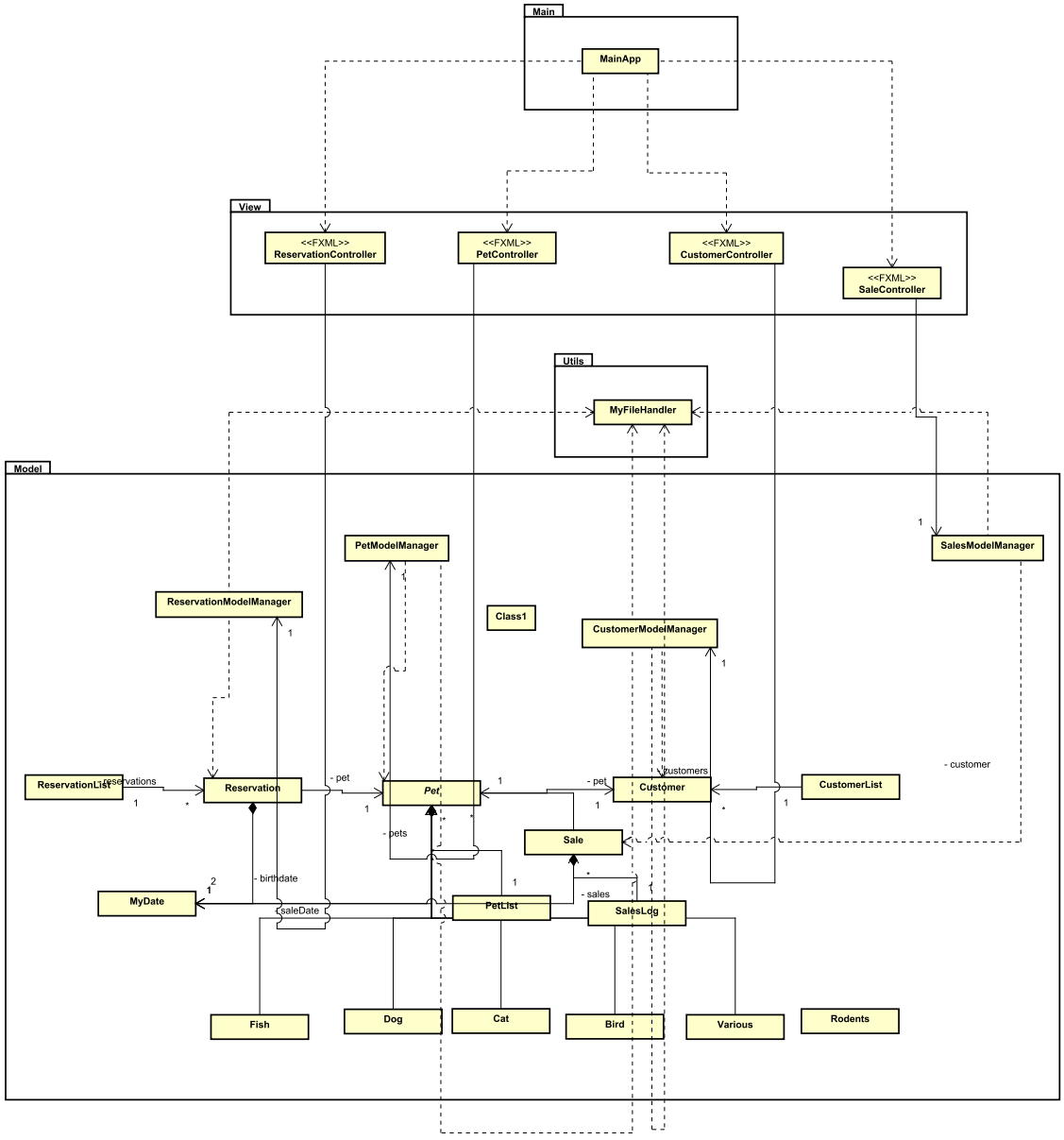
2.2 Design

This section outlines the structure of the VIAPetshop management system, translating the analysis findings into system development. The design focuses on user interfaces and functional modules to meet the pet shop's operational needs, ensuring efficiency, scalability, and user-friendliness.

2.2.1 Class Diagram

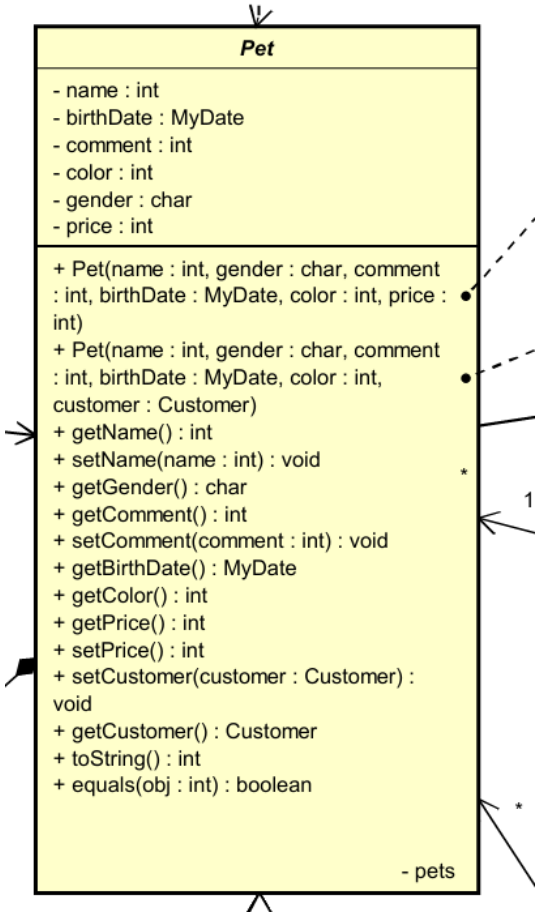


The class diagram for VIAPetshop is derived from the domain model. It identifies the primary methods required to carry out various operations within the system. Additionally, the diagram illustrates the relationships and dependencies among the classes, providing a clear structure for system implementation.



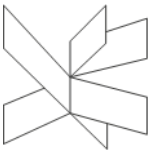
The class diagram for the VIAPetshop system highlights the structure and relationships between key components, ensuring efficient management of pet-related operations.

The Pet class serves as the central entity, managing pet details such as name, birth date, color, gender, price, and comments. It is further specialized in subclasses like Dog, Cat, Bird, Fish, Rodents, and Various, each with specific attributes (e.g., breed, preferred food, or species). The system distinguishes pets in the kennel, which can only include dogs, cats, and birds, while other pets remain in the shop.

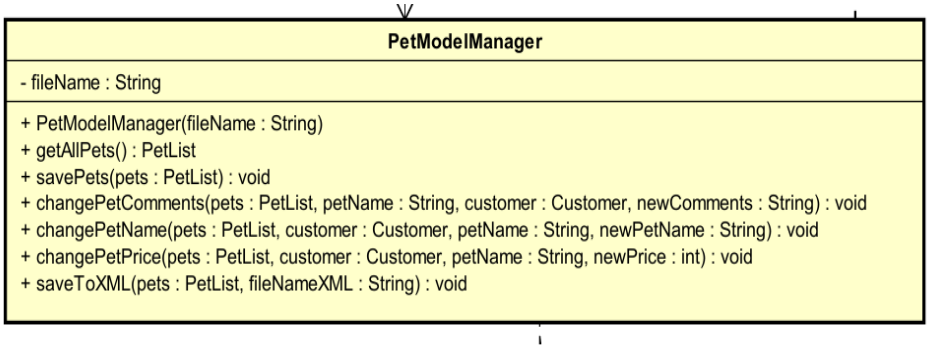


The Customer class manages customer details, including name, phone number, and email address. Customers can own multiple pets, with transactions being handled by the Sale class, which records discounts and prices.

The Reservation class facilitates booking services for pets, with attributes such as price per day, start date, and end date. A constraint ensures that no more than 10 reservations can exist simultaneously, maintaining system efficiency. Reservations and pets are associated with the MyDate class, which tracks essential dates (day, month, year).



The PetModelManager class connects to MyFileHandeler and it handles file operations in binary formats to ensure efficient data storage and retrieval. Overall, the design ensures smooth interaction between classes while maintaining data integrity and enforcing system rules.



Please refer to Appendix E for the complete Class Diagram

2.2.2 Sequence Diagram

The sequence diagram illustrates the flow of interactions between objects in the system to perform specific tasks.

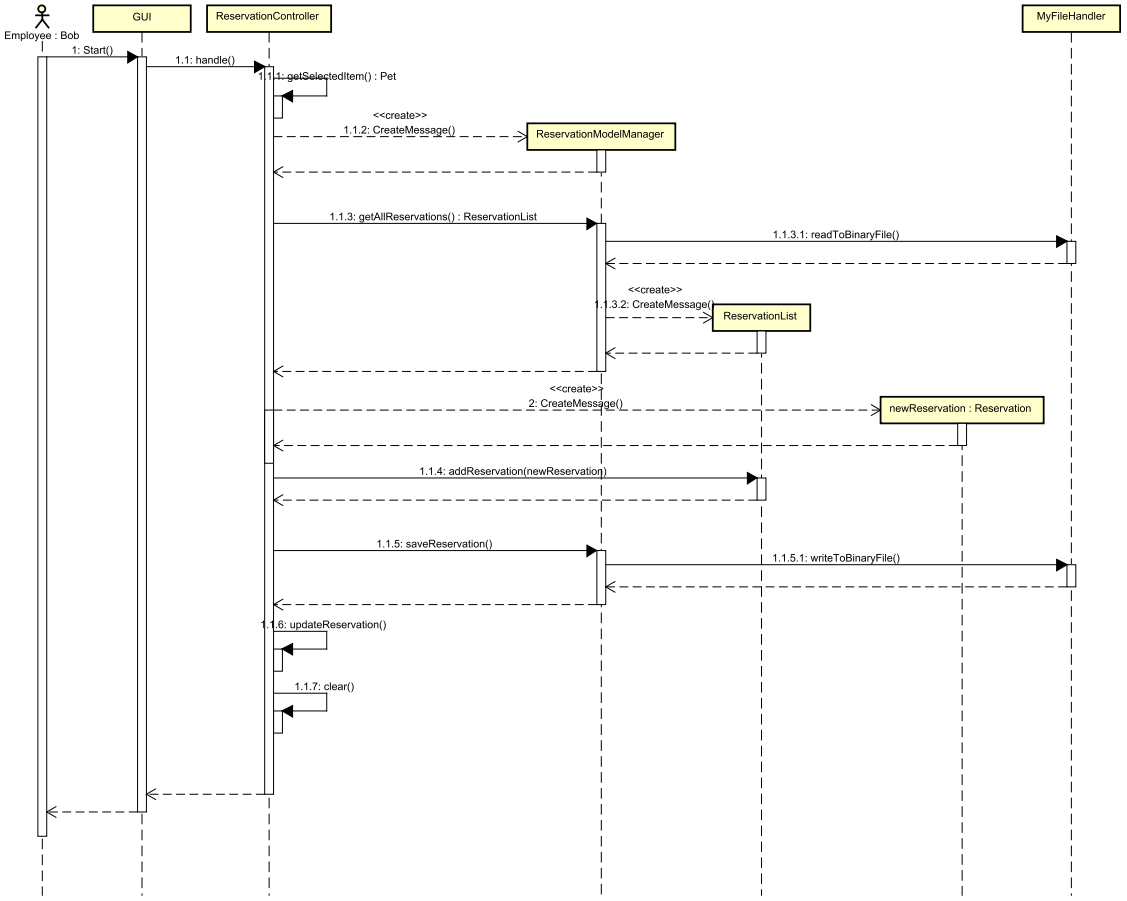


Figure 4 sequence diagram

Sequence of Events for the above diagram will be illustrated bellow;

a) Employee starts the process:

   Employee interacts with GUI.

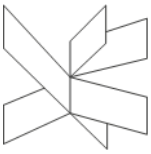
b) GUI handles the input:

   The handle() method is called in the ReservationController.

c) Get pet information:

   The ReservationController calls getSelectedItem() on the GUI, retrieving pet information (:Pet).

d) Create message:



A CreateMessage() request is initiated to communicate with the ReservationModelManager.

e) Retrieve all reservations:

The ReservationController calls getAllReservations() to fetch the reservation list from the ReservationModelManager. readToBinaryFile() operation occurs in response, retrieving the ReservationList.

f) Create a new reservation:

A CreateMessage() operation creates a new ReservationList and subsequently a CreateMessage() to initialize the newReservation object.

g) Add new reservation:

The ReservationController calls addReservation(newReservation) to add the new reservation into the system.

h) Save reservation:

The saveReservation() method is invoked, which triggers the MyFileHandler to persist the data via writeToBinaryFile().

i) Update reservation:

After saving, the ReservationController calls updateReservation() to finalize changes.

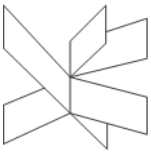
j) Clear GUI:

The GUI clears input fields with the clear() method.

2.2.3 Graphical User Interface (GUI)

The GUI of the VIAPet system is organized into four main tabs: Customer, Pet, Sale and Reservation.

Customer Tab:



This tab allows the employee to add and modify customer information. It also displays an updated list of all customers, ensuring changes are reflected in real time.

Pet Tab:

The Pet tab enables the employee to add pets either to the pet shop or the kennel. Users can also modify pet details based on specific requirements.

Sale Tab:

The Sale tab provides functionality to search for pets and customers. It allows the employee to set a price for pets, apply discounts, and view the sales log for tracking transactions. The following figure shows the sale tab(please see on the appendix described bellow to see the rest class tabs, and functionalities).

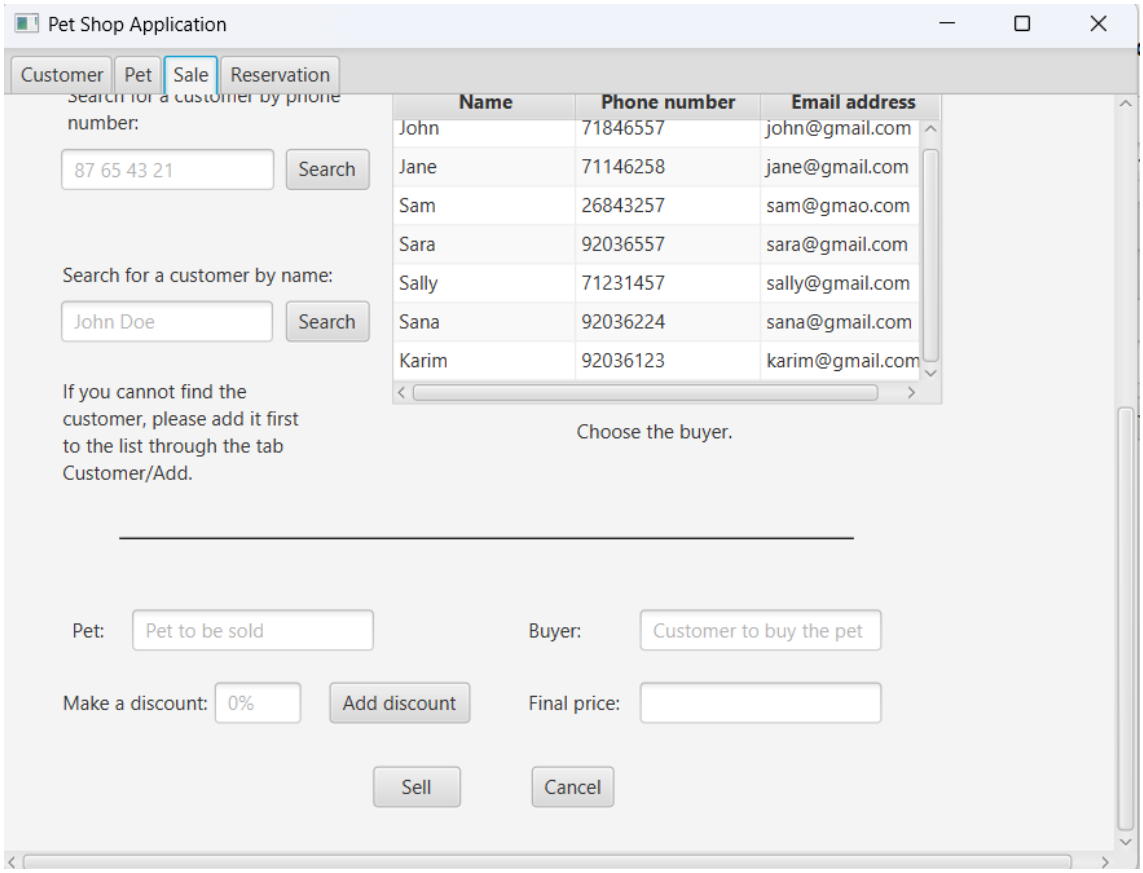


Figure 5 showing SaleTab

Reservation Tab:

This tab allows the employee to add pets to the kennel and check the availability based on its capacity. Reservations are managed using a start date and end date, with a maximum capacity of 10 pets. Only specified pets (cats, dogs, and birds) are allowed in the kennel.

This design ensures a user-friendly interface for managing customers, pets, sales, and reservations efficiently.

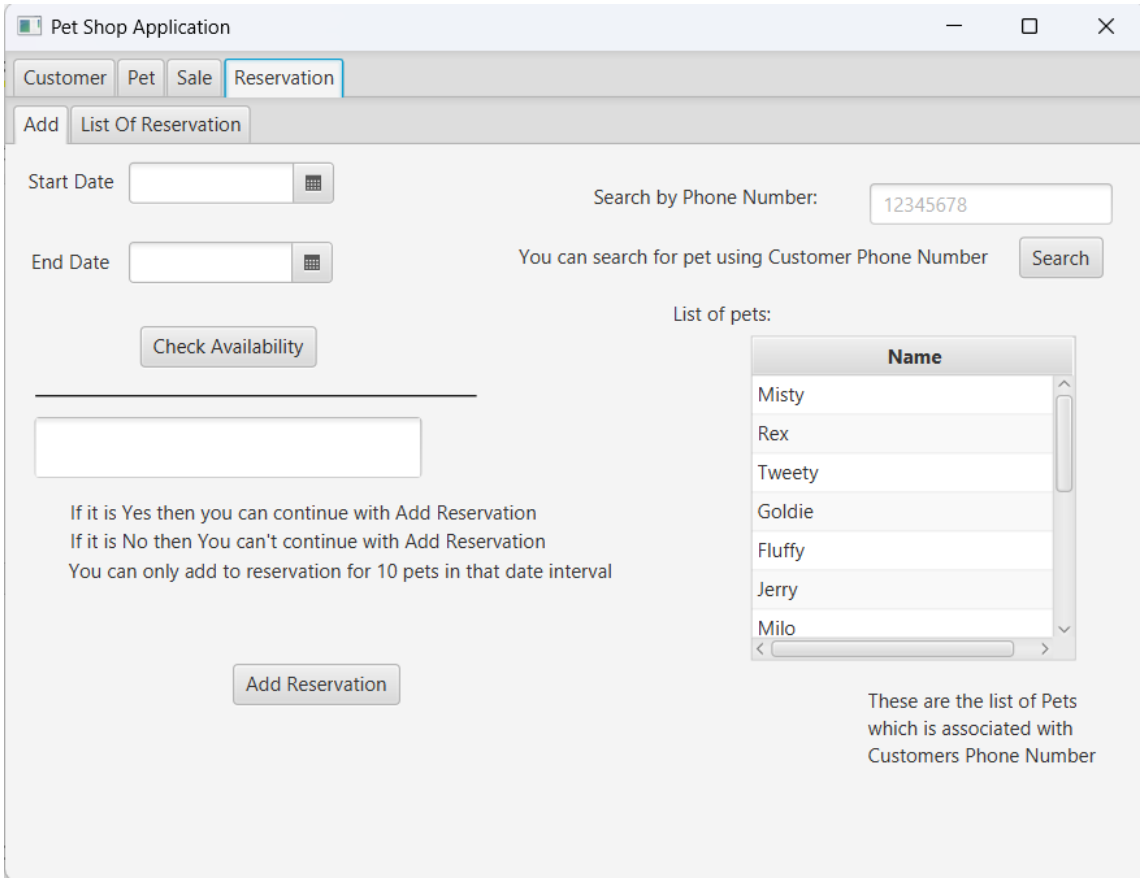


Figure 6 Reservation

Please check Appendix I and run the system to see more GUI samples.

2.2.4 Website

Website Description

The VIApets website serves as an online platform for displaying available pets and kennel room availability. Initially, a draft version of the website was created using Balsamiq Wireframes (see Apendix H) to better visualize its layout and functionalities. This process facilitated the selection of background colors and font styles, primarily utilizing Bootstrap properties.

As per Bob's request, pets cannot be purchased through the website since he values face-to-face interactions with customers. The website is designed purely for

informational purposes, providing an overview of the pets available for sale at the pet shop and the current availability of rooms in the kennel. While customers can browse this information online, all purchases and bookings must be conducted in person at the pet shop.

In the following figure we show the date is available for reservation, when the customer chooses the desired date then the system pop ups as whether the desired date have availability or not.

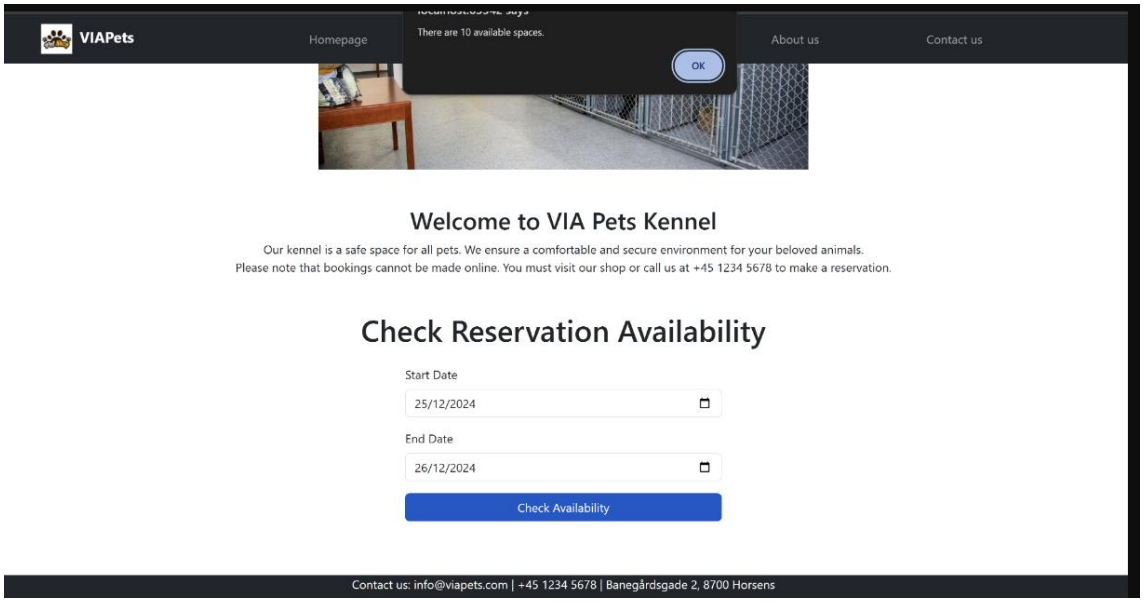


Figure 7 Kennel, illustrates availability by pop op message

Layout

Desktop layout

- The homepage includes a navigation bar with links to "Homepage" "Pet shop", "Kennel ", "About us" and "Contact."
- Key information is displayed prominently with a clear hierarchical structure.

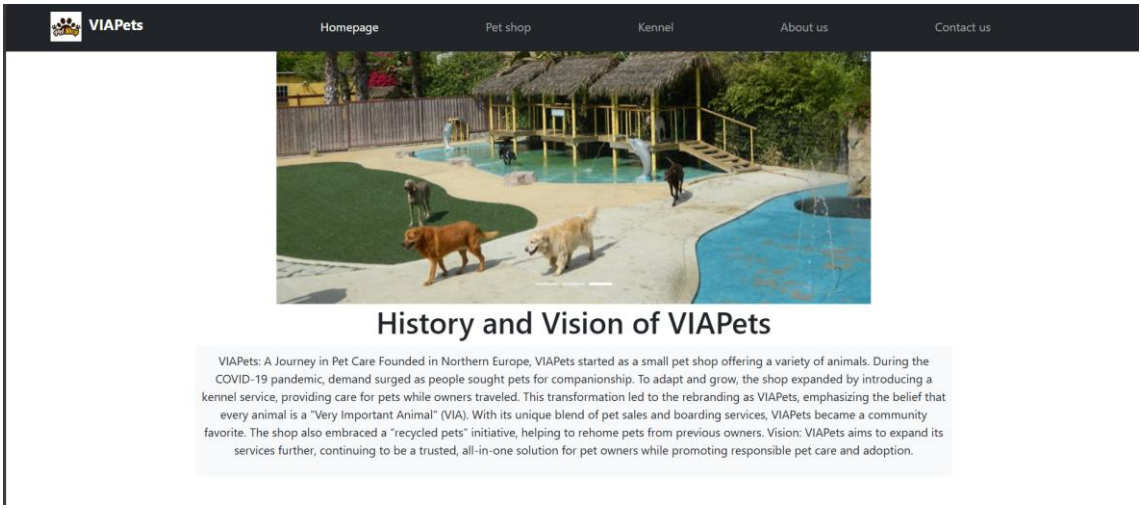
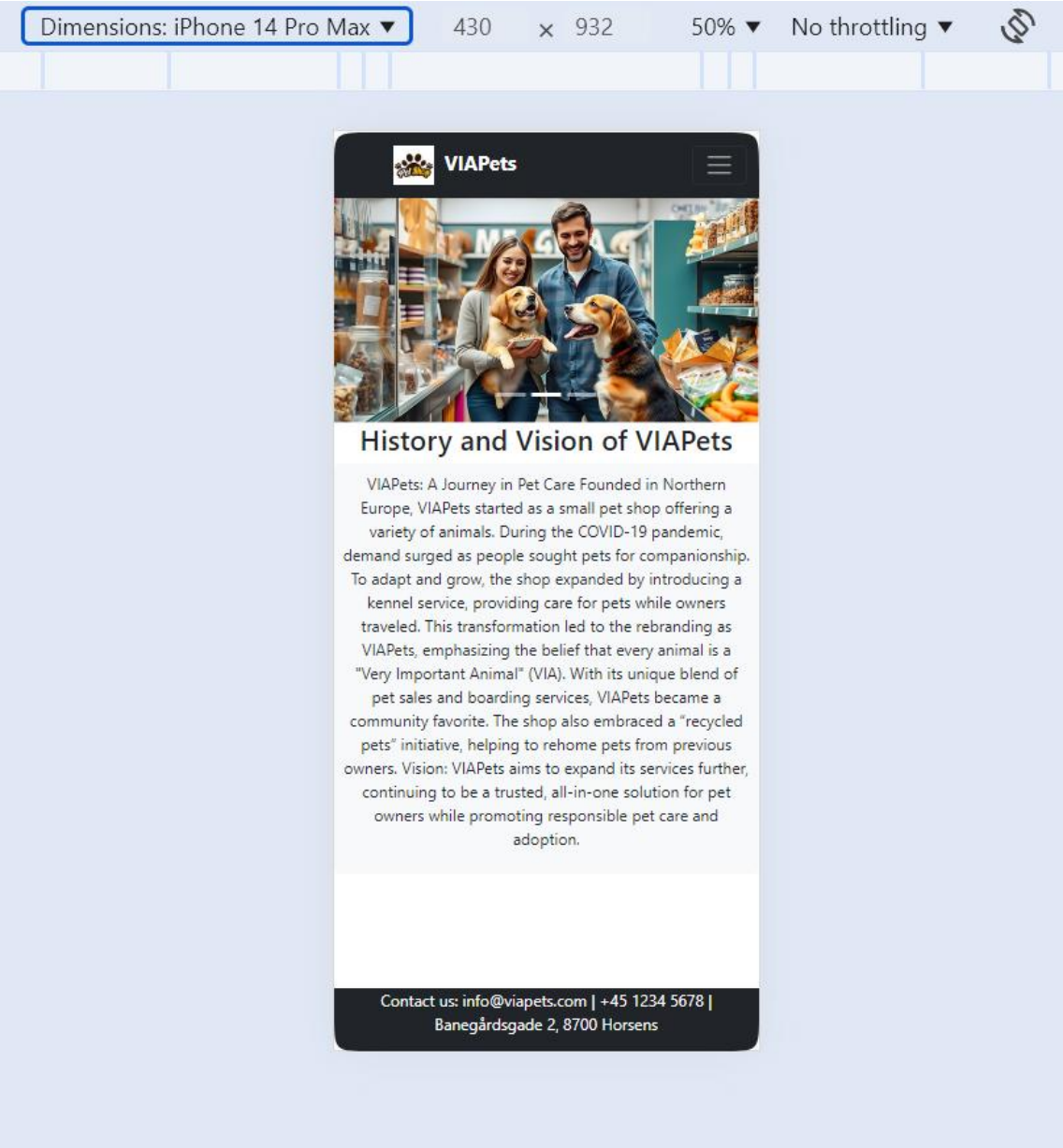


Figure 8 VIAPets Desktop layout

Mobile layout

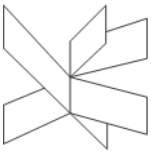
- Elements are resized and rearranged to fit smaller screens.
- Unnecessary visual elements are hidden to focus on core functionalities.



Wireframes

Provide wireframes for both desktop and mobile versions showcasing layouts for key pages.

(Please see Appendix H for wireframes)



2.2.5 Implementation

In the implementation part one of the interesting method that is used to count how many overlaps between any potential reservation, and all the existing reservations, using simple decisions in JS is shown in JS code bellow (figure 11, after java codes).

```java
/**
 * Checks if this reservation overlaps with another reservation.
 *
 * @param other the other reservation.
 * @return true if the reservations overlap, false otherwise.
 */
public boolean overlaps(Reservation other)  3 usages  Youssef Topaji +1
{
  return !(endDate.isBefore(other.startDate) || startDate.isAfter(other.endDate));
}
```

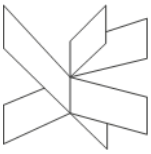
Figure 9 Java code that checks if the reservation overlaps

The same sequence is also used in the application in java, to avoid adding more than 10 simultaneous reservations.

```java
/**
 * Counts the number of reservations that overlap with a given reservation.
 *
 * @param newReservation the reservation to check.
 * @return the count of overlapping reservations.
 */
public int getOverlappingReservationsCount(Reservation newReservation)  5 usages
{
  int count = 0;
  for (Reservation reservation : reservations)
  {
    if (!reservation.equals(newReservation) && reservation.overlaps(
        newReservation))
    {
      count++;
    }
  }
  return count;
}
```

Figure10 Java code that counts the number of reservations that overlap

In java the implementation is more complicated due to easier usage of comparison in between dates in JS.



```
// check availability button click event
$("#checkAvailability").click(function () {
  let userStartDate = new Date($("#startDate").val());
  let userEndDate = new Date($("#endDate").val());
  // check if the user entered valid dates
  if (isNaN(userStartDate) || isNaN(userEndDate)) {...}

  // check if the start date is before the end date
  if (userStartDate >= userEndDate) {...}

  // check if the start date is in the future
  if (userStartDate < new Date()) {...}

  // counter to keep track of the number of overlapping reservations
  let overlapCount = 0;

  // check if the user's dates overlap with any of the reservations
  reservations.forEach((reservation) => {
    if (
      userStartDate <= reservation.endDate &&
      userEndDate >= reservation.startDate
    ) {
      overlapCount++;
    }
  });

  // Display results
  if (overlapCount >= 10) {
    alert("There are no available spaces.");
  } else {
    alert(`There are ${10 - overlapCount} available spaces.`);
  }
});
});
```

Figure 9 JS check availability, dates, overlaps,

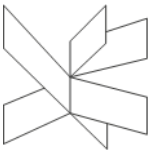
```java
/**
 * Checks if the date is before another date.
 * @param date the date to compare with.
 * @return true if this date is before the other date, false otherwise.
 */
public boolean isBefore(MyDate date)  9 usages  Daniel Weldesilasie
{
  if (year < date.year)
  {
    return true;
  }
  else if (year == date.year)
  {
    if (month < date.month)
    {
      return true;
    }
    else if (month == date.month)
    {
      if (day < date.day)
      {
        return true;
      }
    }
  }
  return false;
}
```

Figure 10 java code checks if the date is before another date

This method counts how many reservations overlap simultaneously using overLaps().

This overlaps method uses isBefore and isAfter methods from MyDate

This implementation of isBefore method is relatively complicated, whereas in JS it requires only a comparison in between dates to return the same result.



3. Discussion

3.1 GUI

3.1.1 Reservation

The Reservation class in the VIApets system plays a critical role in managing bookings for the kennel. It ensures that reservations are handled efficiently and without conflicts, thereby preventing scheduling errors. Two essential methods, overlaps and getOverlappingReservationsCount, enhance the system's capability to handle bookings:

overlaps Method:

This method is designed to check if one reservation conflicts with another by comparing their start and end dates. The logic ensures no overlap exists unless explicitly intended.

```
/**
 * Checks if this reservation overlaps with another reservation.
 *
 * @param other the other reservation.
 * @return true if the reservations overlap, false otherwise.
 */
public boolean overlaps(Reservation other)  3 usages  Youssef Topaji +1
{
  return !(endDate.isBefore(other.startDate) || startDate.isAfter(other.endDate));
}
```

Figure 11 overlap

This provides a robust way to prevent errors during reservation creation, safeguarding the integrity of the booking system.

getOverlappingReservationsCount Method:

This method counts how many existing reservations overlap with a new reservation. By iterating through all current reservations and using the overlaps method, the system ensures no double bookings.

```java
/**
 * Counts the number of reservations that overlap with a given reservation.
 *
 * @param newReservation the reservation to check.
 * @return the count of overlapping reservations.
 */
public int getOverlappingReservationsCount(Reservation newReservation)  5 usages
{
  int count = 0;
  for (Reservation reservation : reservations)
  {
    if (!reservation.equals(newReservation) && reservation.overlaps(
        newReservation))
    {
      count++;
    }
  }
  return count;
}
```

Figure 12 count the nr of reservations

This method is instrumental in maintaining a limit of ten simultaneous bookings. Overall, these methods are integral to the system's functionality and efficiency.

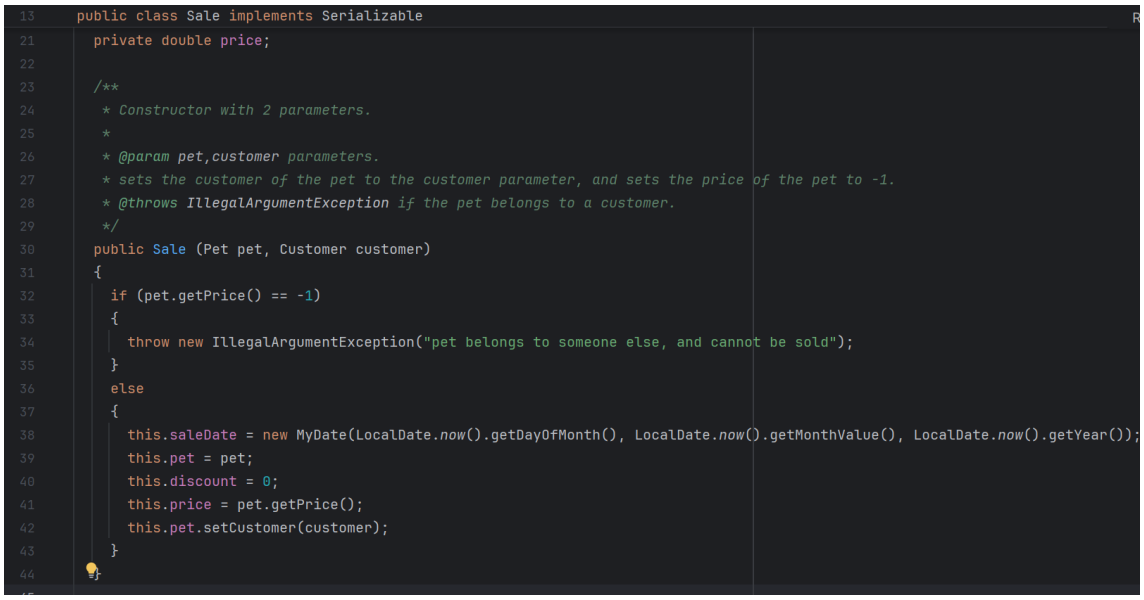
3.1.2 Sale

This enables employees to manage pet sale effectively, ensuring pets are sold only when they are pet for sale. Two key aspects of the Sale functionality include:

Sale Constructor:

The Sale constructor validates that the pet is available for sale by checking if its price is not set to -1 (indicating it belongs to the kennel and is not for sale).

If the pet is already associated with a customer, an exception is thrown to prevent accidental resale.



```java
public class Sale implements Serializable
{
  private double price;

  /**
   * Constructor with 2 parameters.
   *
   * @param pet,customer parameters.
   * sets the customer of the pet to the customer parameter, and sets the price of the pet to -1.
   * @throws IllegalArgumentException if the pet belongs to a customer.
   */
  public Sale (Pet pet, Customer customer)
  {
    if (pet.getPrice() == -1)
    {
      throw new IllegalArgumentException("pet belongs to someone else, and cannot be sold");
    }
    else
    {
      this.saleDate = new MyDate(LocalDate.now().getDayOfMonth(), LocalDate.now().getMonthValue(), LocalDate.now().getYear());
      this.pet = pet;
      this.discount = 0;
      this.price = pet.getPrice();
      this.pet.setCustomer(customer);
    }
  }
```

Figure 13 sale constructor

This ensures the system maintains data integrity and prevents accidental sales.

SalesLog Class:

The SalesLog class maintains a record of all completed sales.

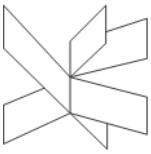
Pets are added to the sales log only if they are marked as available for sale (i.e., belong to the pet shop).

```java
public class SalesLog implements Serializable
{

  /**
   * add a sale to the list if the pet belongs in the pet shop.
   * @param sale to add
   */
  public void addSale(Sale sale)
  {
    sales.add(sale);
  }

  public int size()
  {
    return sales.size();
  }
```

Figure 14 add sale



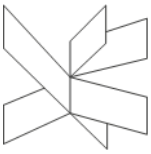
This simplifies the sale list and ensures a clear distinction between pets for sale and those in the kennel.

3.2 Time complexity analysis and Big-O notation

The overall time complexity of the class is O(n).

```
public void savePets(PetList pets)
{
  try
  {
    // Writes the PetList object to a binary file
    MyFileHandler.writeToBinaryFile(fileName, pets); // Time
complexity depends on the size of PetList, O(n)
                                    // where n is the number of pets in
the list.
  }
  catch (FileNotFoundException e)
  {
    System.out.println("File not found or could not be opened pet"); //
O(1)
  }
  catch (IOException e)
  {
    System.out.println("IO Error writing to file pet"); // O(1)
  }
}
```

- **Write Operation**: The dominant operation is MyFileHandler.writeToBinaryFile(). Its complexity depends on:
    - The number of pets in PetList (n).
    - Serialization cost of each pet (k, where k is an integer), which may involve converting each pet's data into binary format.
    - Overall complexity of writeToBinaryFile: O(k.n)
- **Catch Blocks**: Printing error messages in the catch blocks is O(1) and insignificant compared to the file write operation.



Dominating Term Analysis

The dominating term is O(k.n), where n is the number of pets in the list, and **k is the cost** of serializing a single pet.

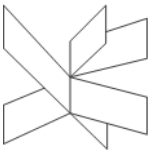
Optimization Suggestions

1. **Buffering the Write Operation**:
    - Ensure the writeToBinaryFile method uses a buffered output stream to optimize the write operation and reduce disk I/O overhead.

Optimized Code Example

```
public void savePets(PetList pets)
{
  try
  {
    // Optimized writing using buffering or delta saving
    MyFileHandler.writeToBinaryFile(fileName, pets); // Optimized O(m * k), where m <=
n
  }
  catch (FileNotFoundException e)
  {
    System.out.println("File not found or could not be opened pet"); // O(1)
  }
  catch (IOException e)
  {
    System.out.println("IO Error writing to file pet"); // O(1)
  }
}
```

Ensures efficiency with larger datasets and frequent writes.

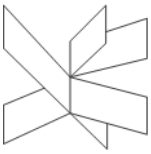


3.3 Test

Testing for VIAPet was conducted after the implementation of the GUI and website to ensure that it met the customer's requirements. Additionally, tests were performed to verify that the functionalities aligned with the use case descriptions as specified.

Table 1 test results

Use case	Test case	Result
Add pet	Verify pet details are added correctly, including specific attributes	Functional
Manage pet data	Update and retrieve pet data with correct associations	Functional
Making a reservation	Create reservations without conflicts and within availability	Functional
Modifying an existing reservation	Edit reservation dates while ensuring no overlaps	Functional
Add customer	Register new customer details into the system	Functional



Managing existing customers information	Update or delete customer details accurately	Functional
Make a sale	Validate sale process for pets, ensuring eligibility (e.g., not in kennel)	Functional
Display a sale log	Retrieve and display accurate historical sale records	Functional
View a website	Confirm the website displays accurate and up-to-date information	Functional

3.3.1 Specification

The table below indicates whether the requirements functionalities have been successfully met or not.

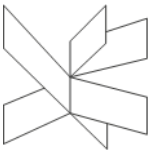
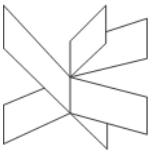
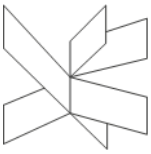


Table 2 Priority impact on the functionality of the system

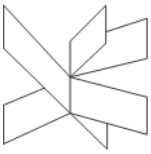
Priority	Requirement number	Requirement	Functional
Critical	1	As an employee, I want to be able to register a pet in the system, including information about the pet's name, age, gender, species, color, so I can easily search for the pet in the system. Additionally, I want to be able to add comments about the pet: a. For dogs and cats, their breed and breeder information b. For rodents, behavior (if they bite) c. For fish, whether they live in salt water or fresh water. Whether they are predators or prey. d. For birds, special dietary needs. e.For all, comment like "under observation" (for health issues or behavior), so all staff are aware of which animals need special attention.	Yes
Critical	2	As an employee, when registering a pet for sale, I want to be able to add a price for it, so it is easily recognizable whether it is a pet shop	Yes



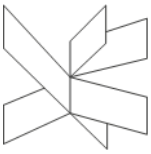
		pet or an animal currently staying in the kennel.	
High	3	As an employee, I want to be able to change the price, in case I want to offer a discount to the customer.	Yes
Critical	4	As an employee, I want to have a system that separates the pets in the pet shop from the pets in the kennel, so we avoid another chihuahua catastrophe.	Yes
High	5	As an employee, I want to be able to remove/delete a booking from the system in case the customer wishes to cancel.	Yes
High	6	As an employee, I want to be able to modify an existing reservation in case the customer wishes to change the date or time.	Yes
High	7	As an employee, I want to be able to input the date in the system to check the availability of rooms in the kennel during that period.	Yes
Critical	8	As an employee, if there are any available rooms, I want to be able to add a new booking	Yes



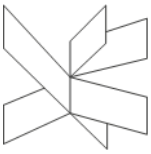
		to the system, so the availability will be updated.	
High	9	As an employee, I want to be able to add a reservation only if there is open availability.	Yes
Critical	10	As an employee, I want to only be able to register a maximum of ten bookings simultaneously, so we ensure that there is open availability.	Yes
critical	11	As an employee, I want to only be able to change an existing reservation to another date, if there is open availability on the day the customer wishes to change to.	Yes
critical	12	As an employee, I want to have a record of each pet's previous owners, if applicable, so I can reference this information if necessary.	Yes
High	13	As an employee, I want to be able to modify existing information about the pet in case the pet gets a new owner.	Yes
critical	14	As an employee, I want to be able to remove/delete a pet from the system in case the pet passes away.	Yes



High	15	As an employee, I want to be able to register a new customer in the system by registering the customer's name, email address and phone number, so I can easily search for them in the system and contact them, if necessary.	Yes
High	16	As an employee, I want to be able to modify an existing customer's information, in case they change their name, email address and/or phone number.	Yes
High	17	As an employee, I want to be able to remove/delete a customer's information from the system in case they do not wish for the shop to store their information anymore.	Yes
critical	18	As an employee, I want to be able to link a pet registered in the system to a customer, newly registered or already registered in the system, so I can keep track on who the owner of the pet I have sold is.	Yes
Low	19	As an employee, I want to be able to add new pets to the pet shop list of pets (adding a new pet for recycling).	Yes
High	20	As an employee, I want to be able to display information on a website including photos,	Yes



		information about pets for sale and room availability, so the customer can see what we offer.	
Low	21	As a customer I want to be able to see the availability of the rooms in the kennel through a website so I can easily know if there is room for my pet in the desired period.	Yes
Low	22	As a customer I want to be able to view a list of all available pets with their prices, so I can decide which one I'd like to buy.	Yes
Low	23	As an employee, I want to be able to store the data in files, and not in a database system.	Yes
Low	24	As an employee, I want the website to update the information automatically, as I update the application.	Yes



4. Conclusion and Recommendations

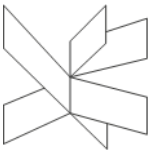
The VIAPets system successfully achieves its goal of managing pet data, reservations, and customer information efficiently. The system includes features to add pets, manage reservations, handle customer data, and store detailed information for each pet type, ensuring that specific requirements, such as unique parameters for fish or other pet types, are met. The system is operational, meeting core requirements and providing a foundation for further enhancements.

The implementation reduces manual processes and ensures accuracy by automating tasks like tracking kennel availability and customer-pet associations. Validation checks and clear error handling reduce potential data input issues, ensuring that the data remains reliable.

Recommendations

The current VIAPets system is single-user and could be extended to support multi-user access with role-based permissions, such as admin and staff logins. Future updates could include:

- Integration of a comprehensive payment system for reservation management.
- Enhanced website features, allowing customers to manage reservations, view kennel availability, and update pet details online.
- Advanced search functionality, enabling queries based on pet attributes, reservation dates, or customer preferences.
- Integration with mobile applications to enhance accessibility for users on the go.



5. References

1. Brooks, R. (2024). Mathematics for software engineering (Version 1.0).

2. Crispin, L., & Gregory, J. (2008). Agile testing: A practical guide for testers and agile teams. Addison-Wesley.

3. Duckett, J., Moore, J., Ruppert, G., & Stone, E. (2011, 2014). HTML & CSS: Design and build websites; JavaScript & JQuery: Interactive front-end web development. Wiley.

4. Gaddis, T. (2015). Starting out with Java: Early objects (5th ed.). Pearson.

5. GeeksforGeeks. (2024, October). Software engineering | Requirements elicitation. https://www.geeksforgeeks.org/software-engineering-requirements-elicitation/

6. GeeksforGeeks. (2024, October). OOPs: Object-oriented design. https://www.geeksforgeeks.org/oops-object-oriented-design/

7. IBM. (2024, October). File storage. https://www.ibm.com/topics/file-storage

8. Martin, R. C. (2002). UML for Java programmers. Object Mentor Inc.

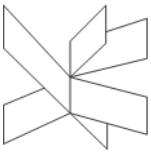
9. Martin, R. C. (2014). Agile software development: Principles, patterns, and practices (1st ed.). Pearson.

10.Oracle, (2024, October 9), What is application development? https://www.oracle.com/application-development/what-is-application-development/

11. Researchgate (2024, October), https://www.researchgate.net/publication/328653592_Risk_Management_for_IT_Projects

12.Researchgate(December, 2024), Influence of strategic approach to BPM on financial and non-financial Performance, https://www.researchgate.net/publication/251414445

13. W3Schools. (2024). W3Schools online web tutorials. https://www.w3schools.com/



Software System for VIAPets - Process Report

By Daniel Weldesilasie (354818), Jwan Hasan (3545289), Lea Maj Gazel (355157), Shishir Gajurel (355402) and Youssef Topaji (355437)

Supervisors: **Allan Henriksen, Jacob M. Wang and Line Linhardt Egegaard**

Number of characters: 26184

Software Technology Engineering

First Semester

17.12.2024

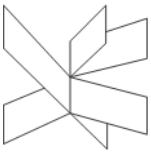
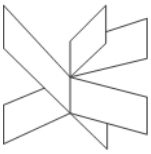


Table of content

Contents



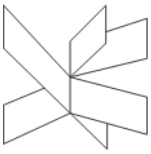
1. Introduction

This report provides an overview of the collaborative process involved in completing our group project on VIA Pets, where the main task was to address the needs of Bob, the pet owner. Our group consisted of five members from four different nationalities: Denmark, Nepal, Ethiopia, and two members from Syria. Initially starting as a group of four, an additional member joined at the onset of the project, introducing both challenges and opportunities for effective teamwork.

The group included diverse individual backgrounds that added unique perspectives to the collaboration. Shishir, 23, from Nepal, brought some prior knowledge of Object-Oriented Programming Systems (OOPS) from his experience working with two companies—one in e-commerce and the other in ISP—before pursuing software engineering in Denmark. Juwan, 26, from Syria, moved to Denmark at 16 and, he started Biomedicine at SDU before VIA, after that focused on volunteering to assist others with translations and legal documentation. Youssef, 28, also from Syria, earned a Bachelor's in Business Administration and studied Software Engineering at Syrian Virtual University before circumstances forced him to pause his education; his commitment to volunteering, including with the Red Cross, has been a consistent source of personal growth. Daniel, originally from Africa, is married and has two children, holds a BSc in Environmental and developmental science and started Computer Engineering at AU, before joining VIA University; he has a background in C/C++ coding and balances full-time work with education. Lea was born and raised in Denmark, but as soon as she finished high school back in 2016, she decided that she wanted to try living in another country. She chose South Korea and lived there for around 1.5 years. She have continuously gone back there every year since then, and she is hoping to be able to find a job, which allows her to move there permanently. she chose Software Technology Engineering at VIA, as she felt like it would have a lot of job opportunities and she would be able to use her hard-earned skills all over the globe.

Working in a multicultural team brought complexities, particularly related to cultural and partial language barriers. However, we effectively navigated these challenges by adhering to our group contract, which outlined clear expectations, roles, and responsibilities, while utilizing structured communication tools to ensure smooth collaboration.

We used Discord as our primary communication platform for discussions and instant updates, GitHub for sharing and managing Java files, and OneDrive to organize project-related documents systematically to avoid confusion. These tools not only enhanced coordination but also supported our efforts to stay aligned with project goals, even when working remotely. Active listening and inclusive discussions were key to our collaboration, ensuring that every member felt heard and respected. By practicing these principles, we avoided conflicts, aligning with conflict resolution methods such as open communication, empathy, and collaborative problem-solving.



Drawing from Tuckman's group development model, our team progressed through the "forming" stage, where we established trust and clarity on roles, and the "storming" stage, where initial challenges were addressed through effective communication and support. By the time we reached the "performing" stage, we had achieved a high level of synergy and productivity, with tasks divided based on members' strengths. Members worked both as a group and partially independently on some assignments to fulfill their responsibilities, yet provided support when others faced challenges, demonstrating a commitment to teamwork principles like interdependence, shared responsibility, and mutual respect.

Maslow's hierarchy of needs also played a significant role in maintaining motivation within the group. The group contract and structured tools ensured a sense of security and clarity (safety needs), while inclusive discussions fostered a sense of belonging. These elements enabled members to contribute meaningfully and achieve a sense of self-actualization through the successful completion of the project.

Our ability to meet at the university or collaborate remotely, combined with the use of well-defined tools and methods (i.e., GitHub, Discord, and Teams), minimized potential conflicts and strengthened our collective effort. Ultimately, the combination of Tuckman's stages of team development, teamwork principles, and conflict resolution strategies ensured the successful delivery of the VIA Pets project.

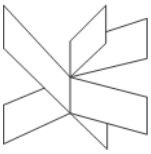
2. Group Work

2.1. Group description

Our group consists of five members from different nationalities, creating a diverse and dynamic team, where each team member brought a unique personal and technical skills shaped by their national and educational backgrounds.

Initially, some members preferred working individually, believing it to be more productive. However, as the project progressed, they adapted to the team-oriented approach and learned to collaborate effectively. The team's diversity brought various problem-solving perspectives, influenced by our prior experiences and technical skills. While some members had prior programming knowledge, problem solving techniques, project writing skills from their previous works, others had no experience, making it a diverse group.

At the beginning of the project, teamwork went smoothly, but challenges arose midway as differences approaches in different ideas and working styles led to minor conflicts. The SEP learning materials were instrumental in helping us navigate these challenges, teaching us how to resolve conflicts professionally. We referred to these issues as "idea conflicts," not personal conflicts. Through the SEP class, we learned strategies to handle conflicts, improve motivation,



and enhance teamwork. Over time, we improved in setting deadlines, meeting them, and working more cohesively as a team.

Individual contributions

- Youssef: Very active and disciplined, Youssef consistently pushed the team to meet deadlines, organized meetings, and paid close attention to details.
- Shishir: A late addition to the group after transferring from another team, Shishir proved to be creative and a problem solver. He was instrumental in finding solutions whenever the team encountered challenges.
- Lea: Lea brought enthusiasm and openness to new ideas. She excelled in report writing, provided guidance, and created a fun and supportive atmosphere during group work.
- Daniel: Daniel contributed to new ideas. He encouraged the group to aim higher, facilitated decision-making.
- Jwan: Active and creative, Jwan not only contributed to the work but also organized social activities like gaming and table football.

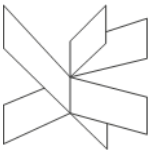
2.2. Project Description

The project description provided us with an overarching view of the tasks we needed to complete. It was essential to understand our customer, his specific need, and the expected timeline for delivery.

With the guidance and feedback from our supervisor, we gained clarity on how to approach the project and move forward effectively. We compiled a detailed list of requirements and set goals to complete the tasks with the necessary features within the given timeframe.

However, due to our limited coding experience and project management skills, completing all the planned features proved challenging, particularly during the final phase of the project. The last two weeks were especially demanding as we worked to address technical complexities and meet the project's objectives.

The goal was to develop a fully functional VIAPets software system using Java programming. The system was designed to enable VIAPets employees to manage kennel reservations and facilitate pet sales. Additionally, the system aimed to provide a customer-facing website where customers could view available pets and kennel availability. However, customers would not



have the ability to order online; they would need to visit the shop physically to reserve a kennel, purchase a pet, or sell a pet to the shop, as VIAPets operates as a recycling pet shop.

The software was intended to let the employee workflows by offering features to edit and manage reservation and customer data efficiently. It included functionality to display customer and reservation lists in a digitalized format. The website was developed to utilize XML data, allowing customers to easily check kennel and pet availability online, enhancing transparency and user convenience.

3. Project Initiation

Why did you select the topic?

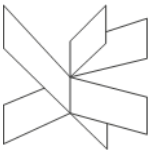
As first-semester students, our project topic was assigned to us with a written interview from the customer detailing the requirements for a system for VIAPets, a combined pet shop and kennel in Northern Europe owned by retired football manager Bob Oldenuff. The goal was to have the system ready in time for Christmas. The written interview outlined the problem domain and included all necessary requirements for the software system, which were provided by our supervisor to guide us in developing the project.

How was the group formed?

Our group members, representing different nationalities, were unfamiliar with one another at the start of the semester. We got to know each other during the first introduction week and decided to form a team when our supervisor announced the need for groups for the SEP project. Fortunately, we were allowed to self-organize, enabling us to bring together a mix of skills and perspectives to the project. Luckly we got the fifth member Shishir who was moved from another group and joined as around the beginning of the project.

How successful was your planning?

Our project planning was largely effective, as the group contract provided clear roles, responsibilities, and regular updates, fostering a shared commitment to our goals. This approach aligned with teamwork principles of mutual accountability and helped us progress through Tuckman's stages, particularly the "forming" and "storming" phases, with minimal conflict.
While our plan kept us on track during most stages, the final two weeks revealed areas for improvement. Challenges with the GUI component and a late-stage code error delayed progress, requiring reworking and re-testing. Despite these setbacks, open communication and conflict de-escalation methods enabled us to stay focused and collaborative.



Using Maslow's hierarchy, the group contract ensured security and belongingness, while our adaptability during setbacks supported motivation. Though these obstacles briefly disrupted progress, our ability to adjust and support one another underscored the strength of our planning. Future efforts should include more flexible contingency plans to handle unexpected challenges effectively.

What kind of project planning tools did you use?

To manage the project's complexity and ensure collaboration, we used a range of tools:

- IntelliJ IDEA Community Edition 2024.2.0.1: For Java programming.
- Astah Professional: For diagram creation and website development.
- Balsamiq wireframes: To create wireframes
- GitHub: For our collaboration.
- Visual Studio Code
- Discord, Microsoft Teams, and Messenger: For team communication and coordination.
- VIA OneDrive: For centralized data storage.
- Google Docs: For real-time collaborative report writing.

These tools helped us stay organized, share updates, and maintain project momentum despite challenges.

4. Project Execution

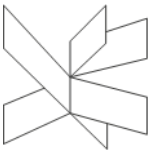
How did you follow up on your plan for the project?

We established a detailed schedule with input from all members, ensuring alignment and accountability, reflecting mutual commitment to a shared purpose and Tuckman's "forming" stage. To stay flexible, we aimed to remain ahead of schedule, allowing time for adjustments and revisions, which helped us navigate the "storming" phase effectively.

Regular progress meetings and revisiting objectives ensured steady progress and reinforced accountability, aligning with the "norming" stage as the team developed cohesive habits. This structured follow-up fostered a sense of safety and belonging, as described in Maslow's hierarchy, ensuring effective collaboration and adaptability throughout the project.

Which methods did you use and were they successful?

We implemented a predominantly Waterfall methodology while incorporating Agile-inspired flexibility when needed. This approach provided a structured framework for progression while allowing us to revisit earlier phases for refinement:



1. Analysis: We conducted analysis of customer requirements, identifying both functional and non-functional needs to ensure clarity and alignment with project goals.
2. Design: The design phase included the development of use cases, activity diagrams, wire frames and sequence diagrams to outline the system and our workflow.
3. Implementation: During implementation, we followed a structured process to build the system. Challenges that arose prompted occasional revisits to earlier stages for refinement and adjustment, ensuring the solution remained aligned with initial requirements.
4. Testing: Testing was conducted to identify and resolve problems, and see the result was reliable and met our goal.

This Waterfall-based methodology enabled us to maintain structured progress, but we did not completely implement it, since we have to revisit and refine our work several times, so we implement the agile method to address challenges dynamically.

Would you use have used other methods, if starting the project today?

If starting over, we would likely adopt an **Agile methodology**. Agile focus on iterative development and continuous feedback could have enabled us to adapt more dynamically to challenges, such as the GUI delays, while ensuring stakeholder input at each stage.

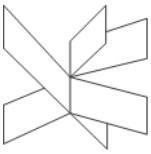
Project results:

In what way are you satisfied with the project results?

Overall, we were satisfied with the teamwork and final deliverables. Each member upheld their responsibilities, and the group functioned cohesively, especially during critical phases. While there is always room for improvement, our dedication and collaborative effort were commendable.

What kind of project risks did you identify and how did you monitor and handle the risks?

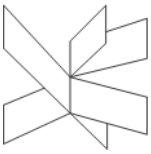
The main risks we encountered were related to time management, communication issues, and occasional distractions like discussing non-project-related topics. Delays in developing the GUI and occasional delays in personal task submissions affected our progress. To address these risks, we redistributed tasks and increased collaboration, which helped us manage setbacks and maintain momentum. This approach aligns with Tuckman's "storming" and "norming" stages, where teams refine their processes and build better cohesion.



We monitored these risks through regular progress meetings and consistently revisited our group contract to keep everyone focused and accountable. (Researchgate, 2024) Fostering a sense of shared purpose and motivation was also crucial, drawing on Maslow's hierarchy of needs to strengthen our team's bond. These strategies helped us manage risks effectively and keep the project on track.

What was less successful? Explain why?

Including pictures for each pet from our application was less successful because we have to do it through java, and for that we have to create fields for that. Which is beyond our level.



5. Personal Reflections

5.1 Personal Reflection of Daniel:

During the teamwork, I felt a strong sense of responsibility and ensured that I contributed meaningfully by completing my tasks on time and be an active participant to the team work. Besides our motivated group members, where we all were mutually committed to our common goals, the group contract was instrumental in our success, providing structure and accountability. It ensured clarity to our duties and responsibilities to the tasks to do, how to resolve/de-escalate conflicts, if they arise, what kind of teamwork approach to follow and our time management approach.

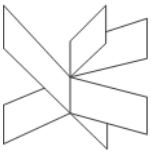
Reflecting on the project, I suggest that future group formation consider forming teams with a mix of skill levels, as well as all members must be actively participant in the common goal and discuss the right thing at the right time. Diversity can enhance team dynamics by providing opportunities for learning and mutual support. Our group functioned well, with all members contributing satisfactorily.

Motivation within the group was nice by our shared goal and the enthusiasm to learn from one another.

However, when occasional minor misunderstandings and idea conflicts arise, we solve them within the same day based on conflict resolution methods, (itadvisory, 2024). The multicultural team was both rewarding and challenging. This project helped me refine essential skills, such as conflict resolution, time management, and adaptability in a group setting. I learned the importance of clear communication and respecting diverse working styles, (hr.mit.edu Tuckman, 2024).

Group work and problem-based learning (PBL) will give learning experiences and opportunities to develop critical thinking skills like communication and problem-solving (itadvisory, 2024), that we will face when we come out in our work life or daily life. While these approaches are valuable, they are not without challenges, including miscommunication and the need for strong time management skills. Clearly defining the problem and project scope is crucial.

Overall, this experience underscored the value of collaboration, adaptability, and clear communication in achieving success, while also highlighting areas for personal and professional growth (Conflict handling models, 2024).



5.2 Jwan's Reflection

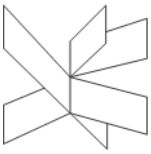
Working on this project, I made it a priority to stay accountable by completing my tasks on time and contributing actively to the group. Our success came down to clear communication, a shared sense of commitment, and having a structured approach to managing our roles and deadlines. One thing that stood out was how much diversity within the team added value. It encouraged creativity and gave us a chance to learn from each other's strengths. When we had minor disagreements, we addressed them quickly through open discussions, which kept the team on track. This experience taught me a lot about adaptability, time management, and resolving conflicts—especially when working with people from different backgrounds. Looking back, I've realized how important teamwork, clear goals, and strong communication are in achieving success. This project not only helped me grow personally but also gave me skills I know will be useful in the future (Tuckman's Model of Team Development, 2024).

5.3 Lea's Reflection

In our recent group project, I felt responsible for ensuring the quality and structure of our written work, particularly in Astah, Case Diagrams, and Domain Models. My role involved synthesizing ideas into cohesive sections, aligning with Belbin's "Completer-Finisher" role, which ensures thorough and timely completion of tasks.

The group contract set clear expectations for communication, deadlines, and responsibilities, helping us avoid conflicts and maintain accountability. However, the lack of consequences for missed deadlines was challenging. For future contracts, I suggest adding clearer consequences and a midway evaluation to allow timely adjustments, aligning with Tuckman's model, which emphasizes structure and norms.

Our group generally worked well together, each member contributing unique skills, though contributions were sometimes uneven due to time constraints or unclear roles. This reflects Hackman's model, which stresses aligning roles with expertise. Motivation fluctuated, initially high but dipping with challenges, then rising with successes. Compliments and problem-solving boosted morale, while unclear tasks and imbalanced efforts were demotivating. Working in a multicultural group enriched our project but posed



communication challenges, aligning with the IPO model, which highlights the importance of motivation and communication.

I learned that I thrive in clear roles and excel at synthesizing ideas and managing deadlines but need to improve communication regarding workload imbalances. In future projects, I'll advocate for structured task division from the outset and regular check-ins to ensure accountability.

Group work and problem-based learning pool diverse skills, fostering creative solutions and critical thinking. However, they can lead to unequal participation and coordination challenges. Clear problem formulations provide direction but can cause confusion if too broad. A project description offers an organized plan but can be restrictive if the project evolves. Integrating Tuckman's stages of development and Belbin's roles highlights the importance of structured roles, clear communication, and regular evaluations to enhance group work efficiency.

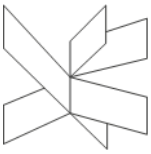
5.4 Shishir’s reflection

During my time working in Nepalese companies, I experienced various dynamics of teamwork that highlighted the importance of clear communication, collaboration, and adaptability. I felt a strong responsibility for the success of our group project, utilizing my expertise in object-oriented programming (OOP) to take on specific technical tasks while also assisting team members through challenges.

Initially, I was part of a group where members lacked focus and cohesion, leading to significant delays and conflicts. This phase mirrored Tuckman’s “storming” stage, where misaligned goals and interpersonal challenges often emerge. Recognizing the impact on our productivity, I reported the issue to our supervisor, which facilitated my transition to a new group. This group emphasized structured teamwork, with clearly defined roles, responsibilities, and deadlines. As a result, we advanced to the “performing” stage, where collaboration and mutual support were at their peak.

In the new team, we effectively leveraged each member’s strengths, creating an environment of mutual accountability and interdependence. Our teamwork resonated with principles outlined by Maslow’s hierarchy of needs. By establishing clear roles and fostering a sense of belonging through valued input, we met foundational needs and progressed toward self-actualization, achieving collective success. Multicultural collaboration further enriched my experience, broadening my communication skills and offering diverse perspectives. However, it also introduced challenges, such as language barriers and differing work styles, which required adaptability and patience.

This experience underscored the importance of structured collaboration. Dividing tasks, setting realistic deadlines, and maintaining regular communication enabled us to overcome



challenges like miscommunication and time management. Problem-based learning and group work provided opportunities for real-world application and critical skill development, such as problem-solving and conflict resolution. Although preparing problem formulations and project descriptions demanded significant time and effort, these processes offered structure and clarity to our work.

Overall, this experience strengthened my abilities in teamwork, time management, and conflict resolution. It also encouraged me to proactively share ideas and take initiative in group settings. By blending diverse skill levels and accommodating varying work styles, we not only delivered a successful project but also reinforced the value of shared learning and responsibility in a collaborative environment.

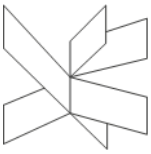
5.5 Youssef's reflection

In our team, we shared specific tasks, and we tried to work together all the time in the forming stage, and that revealed a strong feeling of community of helping each other. And as an important part of our group contract and of the bigger picture of teamwork, we have emphasized the importance of sharing our learning experience with one another. As many studies have demonstrated that

"cooperative learning experiences encourage higher achievement" (Maughan & Webb, 2001).

We have been working together on some tasks, in addition to working separately and we were sharing our learning path and frustrations in an objective manner. and that has shifted the focus from WHO we are to WHAT the group is doing. But during different stages especially in norming and performing, the need to have a leader to coordinate our work has revealed itself from the notice of how much wasted time could be saved if we combine our work time efficiently, and how to keep the work going on the right track, instead of continuously working together, we started dividing our work load into smaller tasks and distribute it between us, and this approach has resulted in a better performance, with maintaining the cooperative learning experience.

This approach revealed to be eventually somewhere in between compromising and collaborating, and I did not feel the competing energy. Therefore, I felt strongly encouraged



and motivated mostly, that has pushed me to be a significant addition in the team. As my motivation was mainly autonomous before starting the project work, and during the teamwork it shifted to be in between extrinsic and intrinsic motivation. And this change came from the realization of our relatedness, and from realizing that the importance of my role comes from the importance of our teamwork as an entity.
Eventually, this has resulted in a great product, and even better understanding of each other and of myself.

6. Reflect on Supervision

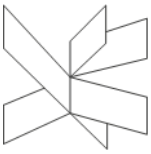
During our semester project, we were fortunate to have three supervisors: Allan Henriksen, Jacob M. Wang, and Line Lindhardt Egsgaard. Each of them played a great role in guiding and supporting us throughout our first-semester journey.

Allan supervised us in programming part and report writing. He was always approachable and quick to respond, offering timely solutions to the challenges we faced. For example, during the project week, Allan was readily available on campus without requiring prior appointments. He was consistently willing to assist us in person whenever necessary. Allan's expertise in programming and his dedication ensured that we could navigate the technical aspects of our project, even though we encountered difficulties due to our limited experience and knowledge, we appreciated his quick response to our email and arrive where we are, on the campus, immediately.

Jacob M. Wang oversaw group formation and teamwork building. He emphasized the importance of effective group collaboration, teaching us skills in conflict management and teamwork (PBL). Jacob's encouragement helped us address issues as a team and complete tasks efficiently while minimizing conflicts. In addition to supervising these aspects, he also taught MSE and PBL, making a significant impact before his parental leave.

Line provided supervision in web development part, as she taught us WEB1. Her guidance was invaluable in helping us implement the web components of our project effectively for the supervision of process and project report.

Overall, all three supervisors played crucial roles in motivating and supporting us. Their combined expertise and commitment were instrumental in helping us complete our semester project successfully, despite the challenges we faced.

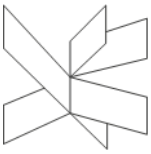


7. Conclusion

To sum up, the project was both a rewarding and challenging experience, offering us the chance to apply theoretical knowledge in a practical context while doing our teamwork and problem-solving skills (Problem-Based Learning, PBL). Despite encountering obstacles, the full support from our supervisors and the dedication of our group enabled us to deliver a functional system.

We successfully met all our objectives and delivered on the expectations we set at the outset. Throughout the project, we gained valuable insights at every phase from group formation and teamwork to analysis, design, implementation, and testing. This aligns with Tuckman's model of team development, where we progressed through the "forming," "storming," and "norming" stages, ultimately achieving a high level of performance and collaboration during the "performing" stage.

Additionally, working as a group provided a fantastic opportunity to enhance our teamwork, communication, and task delegation skills. Drawing from Maslow's hierarchy, we learned the importance of meeting our safety, belongingness, and esteem needs within the team to maintain motivation and collaboration. The effective follow-ups and collaboration we developed throughout the project will undoubtedly serve us well in future endeavors, reinforcing the value of strong team dynamics and proactive risk management in project work.



8. References

1. Conflict Resolution(2024, December), https://itadvisory.dk/ledelse/konflikttrappen-spot-konflikten-og-intervener/

2. Conflict handling models (2024, December), https://mspguide.org/2022/03/18/conflict-styles/

3. Crispin, L., & Gregory, J. (2008). Agile testing: A practical guide for testers and agile teams. Addison-Wesley.

4. Deci, E. L., & Ryan, R. M. (2000). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. Contemporary Educational Psychology, 25(1), 54-67.

5. Martin, R. C. (2014). Agile software development: Principles, patterns, and practices (1st ed.). Pearson.

6. Researchgate(2024,October), https://www.researchgate.net/publication/328653592_Risk_Management_for_IT_Projects

7. Ryan, R. M., & Deci, E. L. (2000). Self-Determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-Being. American Psychologist, 55(1), 68-78.

8. Tuckman (2024, December), https://hr.mit.edu/learning-topics/teams/articles/stages-development

9. Belbin, R. M. (2010). Team Roles at Work. Butterworth-Heinemann.

10. Hackman, J. R. (2002). Leading Teams: Setting the Stage for Great Performances. Harvard Business School Press.

11. McGrath, J. E. (1984). Groups: Interaction and Performance. Prentice-Hall.

12. Tuckman, B. W. (1965). Developmental sequence in small groups. Psychological Bulletin, 63(6), 384-399.