cuACS Project

Requirement Analysis Document

Team 070 – Segmentation Fault

Weihang Chen

Liyongshi Chen

Richard Xia

Xiran Zhou

Submitted to:

Dr. Christine Laurendeau

COMP3004 Object-Oriented Software Engineering

School of Computer Science

Carleton University

Table of Content:

1. Introduction

For decades, there have been concerns about the welling-being of the homeless animals, which enabled the increasing demands for animal adoption and their associated services. To make the adoption process more efficient while achieving the best animal-client matching result, the need for creating an easy-to-use online tool/system has been raised. Such system can be used by the staffs of an animal shelter, or by potential clients who wish to adopt animals. Carleton University Animal Care System (cuACS) proposes to accomplish this goal by providing a user-friendly software system with optimal matching capability.

1.1. Purpose of System

The goal of the cuACS project is to improve the existing animal adoption process by providing an intelligent software system that is capable of managing animal and human information, as well as computing an optimal set of animal-client matches.

The system provides users with several main features. Clients can view the information of the animals available for adoption, as well as updating their matching preferences or other personal information by editing their own profile. Shelter staff can view, add, and edit animal information. They can also view existing clients and add new clients. Moreover, they can launch the execution of an animal-client matching algorithm and view the resulting matches.

Currently, the animal adoption process of the animal shelters has flaws: one is that it sometimes allows animals to be adopted by humans with whom they are not fully compatible. This incompatibility results in many disadvantages, including a mismatch of needs, personality, and expectations. Such unresolved issue in the existing systems promoted the creation of cuACS.

1.2. Overview of Document

The main focus of this document is to analyze the requirements from the client and output a system specification that satisfies those requirements. The document is divided into multiple sections involving an introduction of the project, the functional and non-functional requirements of the system, use cases, and object models.

Section 1 provides an introduction to the cuACS system and an overview of the document.

Section 2 and 3 lists and describes the functional requirements and the non-functional requirements of the cuACS system.

Section 4 covers the high-level and detailed use cases of the system.
Section 5 covers the object models of the system, where each object is identified and described in the data dictionary sub-section, and the associations between objects are modeled in UML class diagrams sub-section.

2. Functional Requirements

The functional requirement defines the high-level functionality that the system should support and the features that the system must provide. The table below shows the functional requirements of the cuACS.

Table 1 – Functional Requirements

FR-01: The Client should be able to view animals' information.
FR-01-01: The Client can view a list of all animals available for adoption.
FR-01-02: The Client can view the detailed profile of a selected animal.
FR-02: The Client can edit their own profile.
FR-02-01: The Client can update their personal information.
FR-02-02: The Client can update their matching preferences.
FR-02-03: The Client can add new preferences.
FR-03: The Staff can view animal information.
FR-03-01: The Staff can view a list of all animals available for adoption.
FR-03-02: The Staff can view the detailed profile of a selected animal.
FR-04: The Staff can add animal information.
FR-04-01: The Staff can edit animal profile information.
FR-04-02: The Staff can add a new animal.
FR-05: The Staff can view clients' information.

FR-05-01: The Staff can view a list of all existing clients.
FR-05-02: The Staff can view the detailed profile of a selected client.
FR-06: The Staff can add new clients.
FR-07: The Staff can launch the execution of the ACM algorithm and view the resulting matches.

3. Non-Functional Requirements

Non-functional requirement describes user-level requirements that are not directly related to functionality. The table below shows the non-functional requirements of the cuACS.

Table 2 – Non-Functional Requirements

NFR-01: Usability
NFR-01-01: Clients must be able to access cuACS system without prior registration.
NFR-01-02: The system supports online help service.
NFR-01-03: The system provides users with specifications to help user understand how to use the system.
NFR-02: Supportability
NFR-02-01: The Staff should be able to update the ACM algorithm when required.
NFR-02-02: The Staff should be able to upgrade cuACS system when required.
NFR-02-03: When a software bug is found, the technical support group should be able to fix it within 5 business days. The system shouldn't be interrupted during this process.
NFR-03: Operations
NFR-03-01: The Staff and the Client need to log into the cuACS system.
NFR-03-02: The Client can only view and edit their own preference. Their permission is limited.
NFR-03-03: The staff cannot change clients' matching preferences.

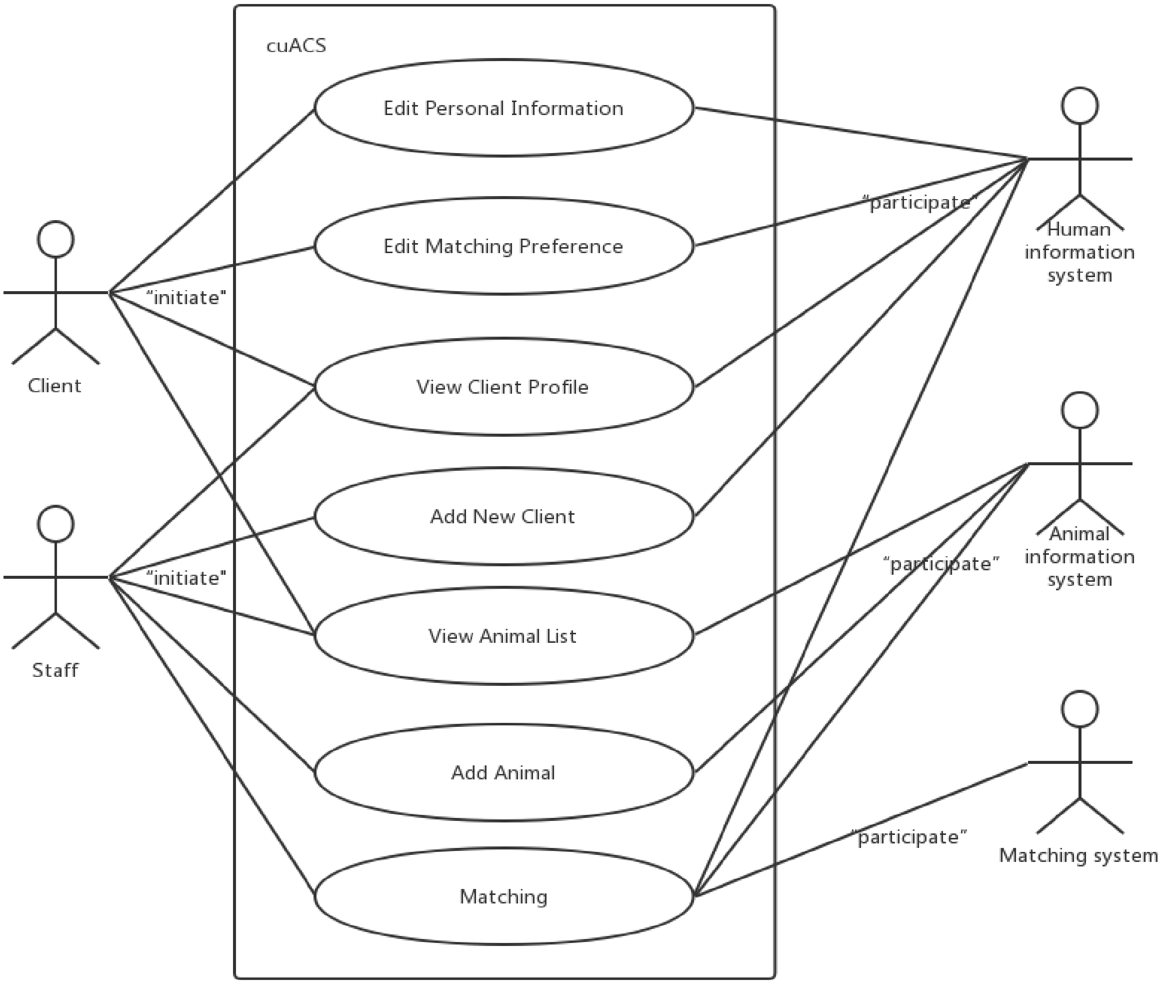
NFR-04: Reliability
NFR-04-01: System crashes should happen at most once per year.
NFR-04-02: When the system is interrupted because of a crash, only data added within 10 business days can be lost at most.
NFR-04-03: When the system crashes, the system will repair the data stored from the last time to avoid more losses.
NFR-04-04: If the user inputs an invalid value or get an error warning, the system can roll back to the previous step and provide support / suggestions to the user.
NFR-05: Implementation
NFR-05-01: All users should be able to access the system with a single command on a UNIX OS.
NFR-05-02: The system should run on most UNIX operating systems.
NFR-05-03: The system supports compilation by C++ language.
NFR-06: Packaging
NFR-06-01: The Client only needs to download a package file, e.g., zip.
NFR-06-02: The data file is independent. Regular updates of the data file shouldn't influence the rest of the system.
NFR-06-03: The system's size is limited to 500MB.
NFR-07: Performance
NFR-07-01: The system must support many concurrent clients (e.g., 10000).
NFR-07-02: The running time of each process is at most 2 seconds.
NFR-07-03: The system can match multiple clients with multiple animals.
NFR-08: Interface
NFR-08-01: The system supports the implementation to interact with other organization's system.
NFR-08-02: The Client and the Staff can run the system on most animal shelter website.
NFR-08-03: The system should notify the Client about the matching result by email.

NFR-09: Legal
NFR-09-01: Client information must be encrypted to protect their privacy.
NFR-09-02: The lawful object needs to hold an operating license for business.
NFR-09-03: The lawful object needs to follow the rules and regulations of the local Animal Protection Society.

4. Use Case Model

Use-case models simulate different interactions between the user and the system.
High-level use cases are the interactions between systems and users.
The figure below depicts the high-level use cases of the cuACS.

Figure 1 -- High-level Use Case Diagram



EditMatchingPreference	The Client can edit their Preference information
EditPersonalInformation	The Client can edit their Personal information
ViewAnimalList	The Client or Staff can view all the information of Animals
ViewClientProfile	The Staff can view Client’ s profile
AddNewClient	The Staff can create a new account for the Client
AddAnimals	The Staff can add new animals into the list
Matching	The Staff can launch a term of matching to match animals with their reasonable adopter(Client)

Use Case IDs and Descriptions:

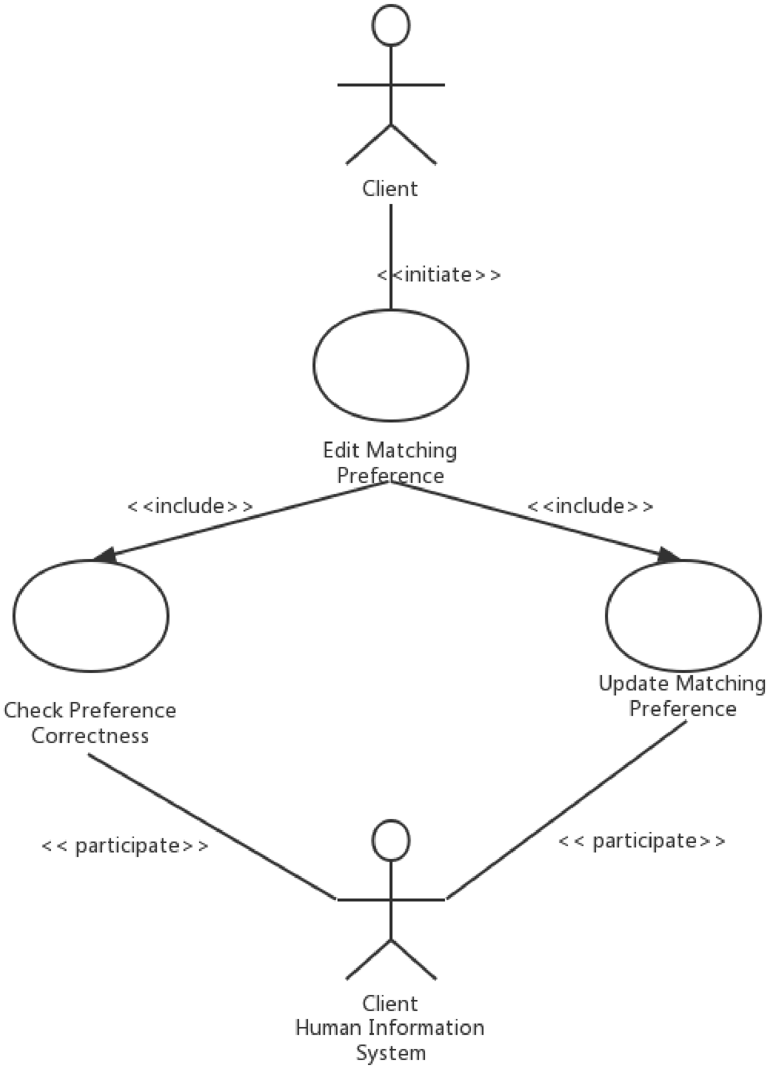
UC-01 EditMatchingPreference	The Client can edit their Preference information.
UC-02 UpdateMatchingPreference	The Client can update their Preference information.
UC-03 CheckPreferenceCorrectness	The system checks if the preference is correct.
UC-04 NonePreference	The system checks if no preference added.
UC-05 RepeatedAttributes	The system checks if attribute is repeated.
UC-06 EditPersonalInfomation	The Client can edit their own personal information.
UC-07 UpdatePersonalInformation	The Client can update their own personal information.
UC-08 CheckPersonalDetailsCorrectness	The system checks if the personal details is correct.
UC-09 InvalidInputDateType	The system checks if the data type is correct.
UC-10 LackNecessaryDetails	The system checks if any necessary detail missed.
UC-11 ViewAnimalList	The Staff or the Client can view a list of animals.
UC-12 ViewAnimalOverview	The Staff or the Client can view the general information of all animals.
UC-13 ViewAnimalDetail	The Staff or the Client can view the details for a selected animal.
UC-14 ViewClientProfile	The Staff can view a list of client’s profile.
UC-15 ViewClientOverview	The Staff can view general information of all clients.
UC-16 ViewClientDetail	The Staff can view the details of a selected client.
UC-17 AddNewClient	The Staff can add a new client.
UC-18 CheckNameConflicts	The system checks if the name already exists.
UC-19 RegisterClient	The Staff can create a profile for new clients.
UC-20 AddAnimals	The Staff can add a new animal.

UC-21	AddAnimalProfile	The Staff can create a profile for an animal.
UC-22	EditAnimalInfo	The Staff can edit the information of a selected animal.
UC-23	Matching	The Staff can launch the matching algorithm.
UC-24	GetAnimalInfo	The system gets animal information for matching.
UC-25	GetClientPre	The system gets client preference for matching.
UC-26	MatchInfo	The system matches animal with preference using ACM algorithm.

Table 3 -- High-level and Detailed Use Case Description

The figures and tables in this sub-section are to show the description of the high-level use cases along with their detailed use cases and extended use cases.

Detailed use cases refining the EditMatchingPreference high-level use case.



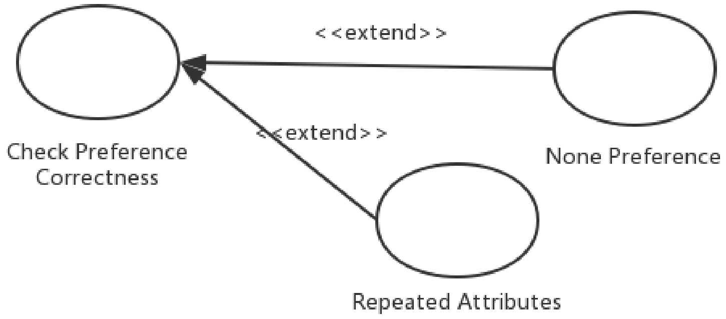
Detailed use case for EditMatchingPreference:

User case name	EditMatchingPreference
ID	UC-01
Participating actors	Initiated by Client Communicates with Human Information system
Flow of events	1. The Client clicks the button to Edit Matching Preference 2. If the Client chooses their preference(s) and clicks the button to submit, the Human Information system checks the correctness of preference form (include use case CheckPreferenceCorrectness). 3. If the Client got the correct preference(s), the Human Information system will update the new matching preference (include use case UpdateMatchingPreference).
Entry condition	The Client must be logged in to cuACS.
Exit condition	1. The Client submits their data successfully or cancels this operation. 2. Or the request is failed to pass the correctness check.
Quality Requirement	The system should provide online help to the first-time user.
Traceability	FR-02-02, NFR-01-02.

User case name	UpdateMatchingPreference
ID	UC-02
Participating actors	Communicates with Human Information System
Flow of events	1. Update and store the new matching preference in the Human Information System; 2. Display a message to the Client to inform the success;
Entry condition	1. A request for update matching preference (include use case EditMatchingPreference) has been submitted to Human Information System

	2. This request gets a pass from correctness check (include use case CheckPreferenceCorrectness).
Exit condition	Human Information has updated the data.
Quality Requirements	The system should prompt the user once finishes updating.
Traceability	FR-02-02.

Exceptions occurring in Edit Matching Preference as extending use cases:



NonePreference:	Inform Client that at least one preference is required.
RepeatedAttributes:	Inform Client that they can not select attributes as both favorite and tabooing.

Detailed use case for CheckPreferenceCorrectness

User case name	CheckPreferenceCorrectness
ID	UC-03
Participating actors	Communicates with Human Information System
Flow of events	1. Check whether none attributes assigned as a preference; 2. Check whether some attributes are both favorite and tabooing; 3. Return the check result to Human Information System.

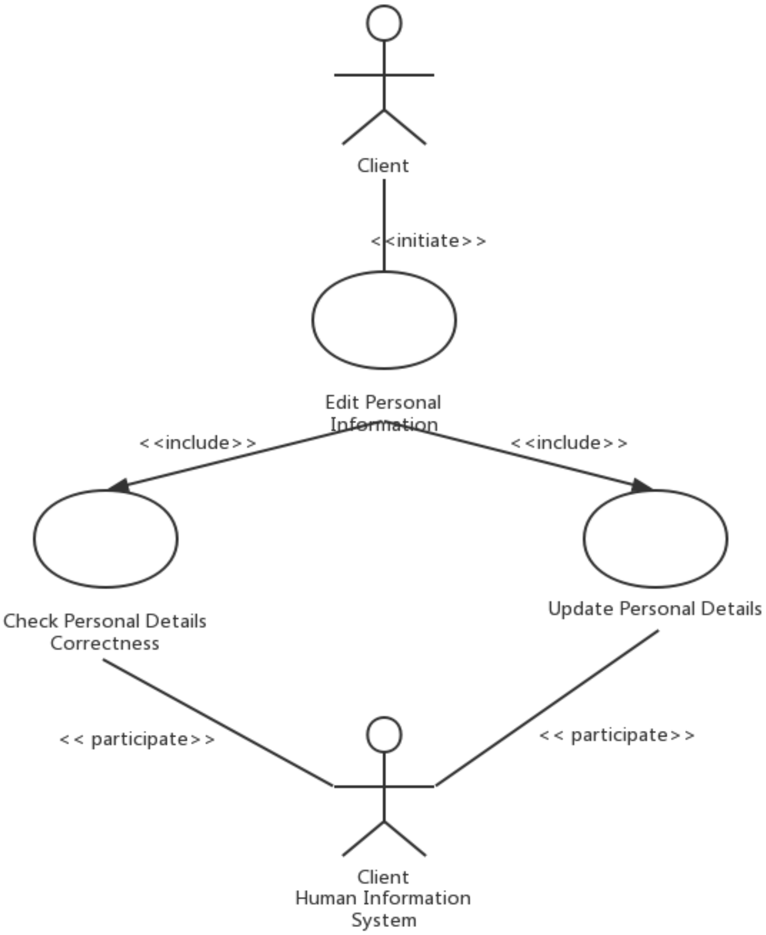
Entry condition	A request of update matching preference (include use case EditMatchingPreference) has been submitted to Human Information System
Exit condition	The process of checking is finished.
Quality requirement	The system should prompt the user an error message and provide online help option if correctness check fails.
Traceability	NFR-01-02, NFR-01-03.

User case name	NonePreference
ID	UC-04
Participating actors	Communicates with Client
Flow of events	1. Display a message to inform Client that at least one preference is required. 2. No data will be saved or updated, and the page to gather information will remain the same.
Entry condition	A request to update matching preference (include use case EditMatchingPreference) has been submitted to Human Information System.
Exit condition	This use case extends the CheckPreferenceCorrectness use case. It is initiated by the Client chooses none attributes as their preferences.
Traceability	NFR-01-02.

User case name	RepeatedAttributes
ID	UC-05
Participating actors	Communicates with Client
Flow of events	1. Display a message to inform Client that at least one preference is required. 2. No data will be saved or updated, and the page to gather information will remain the same.

Entry condition	This use case extends the CheckPreferenceCorrectness use case. It is initiated by the Client chooses none attributes as their preferences and taboo at the same time.
Exit condition	The Client read and closes the message.
Quality requirement	The system should prompt the user to select a valid input.
Traceability	NFR-01-03.

Detailed use cases refining the Edit Personal Information high-level use case



The detailed use case for EditMatchingPreference:

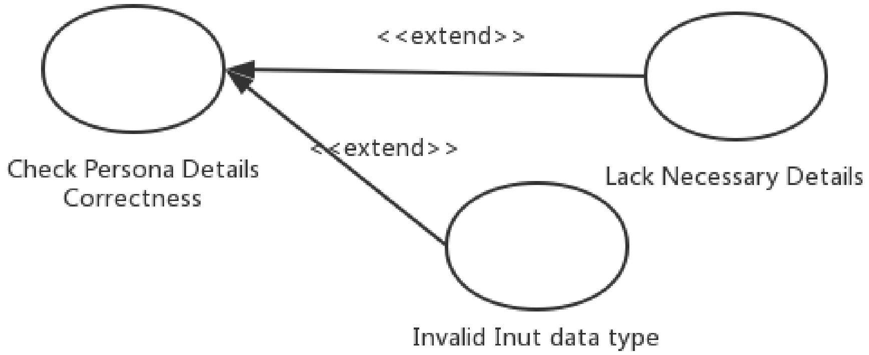
User case name	EditPersonalInformation
ID	UC-06
Participating actors	Initiated by Client Communicates with Human Information
Flow of events	1. The Client clicks the button to Edit Personal Information. 2. The Human Information System provides a sheet or list for the Client to collect the personal information from the Client. 3. The Client edits their preference(s) and clicks the button to submit. 4. The Human Information System receives the request and checks the correctness of request form (include use case CheckPersonalDetailCorrectness). 5. If the request gets the pass from correctness check, the Human Information System will update the new personal information (include use case UpdatePersonalDetails).
Entry condition	The Client must be logged in to cuACS.
Exit condition	1. The Client submits their data successfully or cancels this operation. 2. Or the request is failed to pass the correctness check.
Traceability	FR-02-01.

User case name	UpdateMatchingPreference
ID	UC-07
Participating actors	Communicates with Human Information System
Flow of events	1. Update and store the new personal information in the Human Information System; 2. Display a message to the Client to inform the success;
Entry condition	1. A request to edit personal information (include use case EditMatchingPreference) has been submitted to Human Information System. 2. This request gets pass from correctness check (include use case CheckPersonalDetailsCorrectness).

Exit condition	Human Information has updated the data.
Traceability	FR-02-02.

User case name	FailToEditPersonalDetails
ID	UC-08
Participating actors	Communicates with Client
Flow of events	1. An error message is displayed to the Client; 2. Request of Edit Personal Information is canceled;
Entry condition	Failed correctness check of personal information (include use case CheckPersonalDetailsCorrectness).
Exit condition	None.
Quality requirement	The system should provide online support option to the user.
Traceability	NFR-01-02.

Exceptions occurring in Check Personal details Correctness as extending use cases:



InvalidInputDateType Inform Client that the data types of input are invalid
LackNecessaryDetails Inform Client that at they need to provide some necessary information.

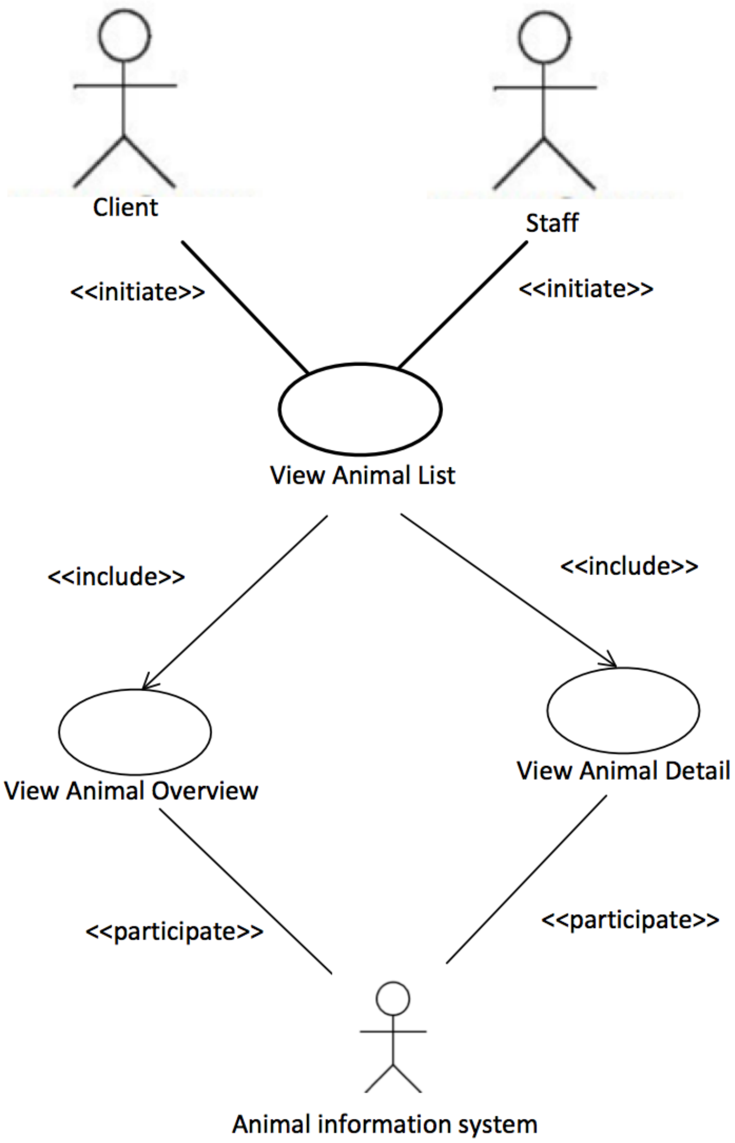
Extended use cases for CheckPersonalDetailsCorrectness

User case name	CheckPersonalDetailsCorrectness
ID	UC-09
Participating actors	Communicates with Human Information System
Flow of events	1. Check whether necessary information is provided; 2. Check whether the data types of input are corrective; 3. Return the check result to Human Information System
Entry condition	A request to edit personal information (include use case EditPersonalInformation) has been submitted to Human Information System.
Exit condition	The process of checking is finished.
Quality requirement	The system should provide specification to help the user understand how to use the system.
Traceability	NFR-01-03.

User case name	InvalidInputDateType
ID	UC-10
Participating actors	Communicates with Client
Flow of events	1. Display a message to inform Client that the data types of input are invalid. 2. No data will be saved or updated; and the page to gather information will remain the same.
Entry condition	This use case extends the CheckPersonalDetailsCorrectness use case. It is initiated by the Client assigns some attributes with unexpected data types.
Exit condition	The Client closes the message.
Traceability	NFR-03-01.

User case name	LackNecessaryDetails
ID	UC-11
Participating actors	Communicates with Client
Flow of events	1. Display a message to inform Client that some necessary information is required. 2. No data will be saved or updated; the page that gathers information will remain the same.
Entry condition	This use case extends the CheckPersonalDetailsCorrectness use case. It is initiated by the Client lack to provide some necessary details when they require to update personal information.
Exit condition	The Client closes the message.
Traceability	NFR-01-02.

Detailed use cases refining the ViewAnimalList high-level use case:



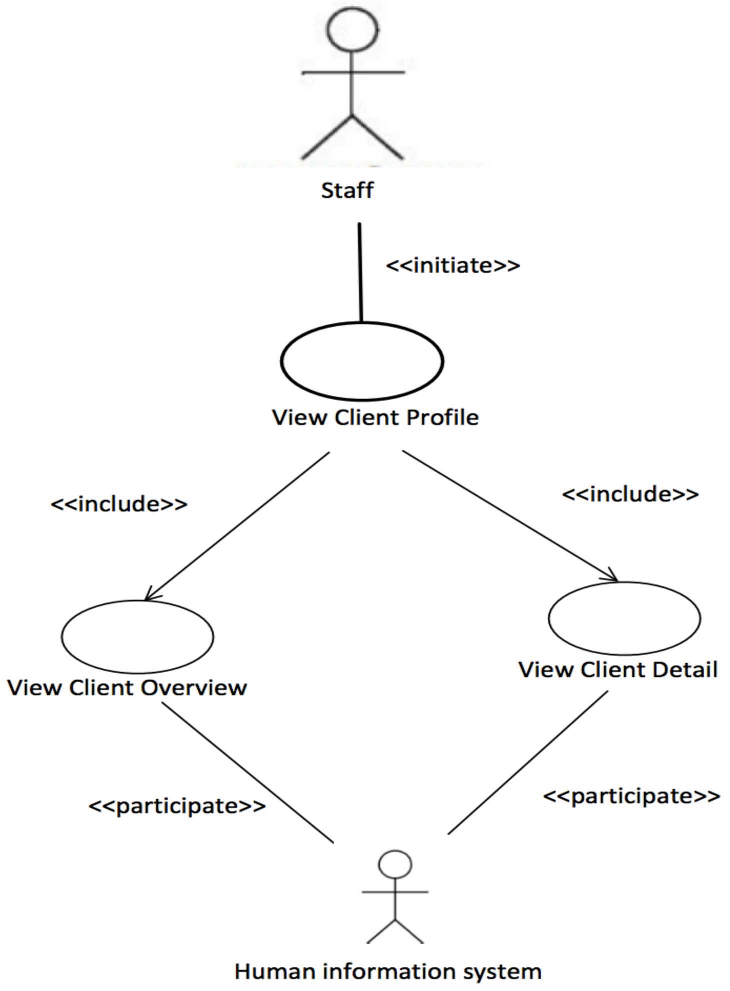
User case name	ViewAnimalList
ID	UC-12
Participating actors	Communicates with Client, Staff Communicates with Animals Information System

Flow of events	1. The Client or Staff can choose to view a list of animals with detailed information. 2. The Animal Information System display a list of overview for all animals, such as id, name, gender, type and age (include use case ViewAnimalOverview). 3. The Client or Staff can view more details of one specific animal. 4. The Animal Information System display all of details about the one specific animal (include use case ViewAnimalDetail).
Entry condition	The Client or Staff choose to view a list of all animals for adoption.
Exit condition	Animal Information System returns animal list and have completed actor's request.
Quality requirement	The system should be able to display the database of animals in a customizable manner.
Traceability	NFR-08-01, NFR-08-02.

User case name	ViewAnimalOverview
ID	UC-13
Participating actors	Communicates with Client, Staff
Flow of events	1. The Animal Information System search all of animals. 2. The Animal Information System display the list of overviews for all animals, such as id, name, gender, type and age.
Entry condition	The Client or Staff choose to view a list of all animals for adoption.
Exit condition	Animals Information System returns animal list and have completed actor's request.
Quality requirement	Animal list should be sorted alphabetically and by size.
Traceability	NFR-03-03.

User case name	ViewAnimalDetail
ID	UC-14
Participating actors	Communicates with Client, Staff
Flow of events	1. The Animal Information System get the Client or Staff's request. 2. The Animal Information System search specific one animal and display all of information about the animal.
Entry condition	The Client or Staff choose to one speacific animal to view details.
Exit condition	Animals Information System returns the information of animal that the Client or Staff requests.
Quality requirement	The system should be able to display the database of animals in a customizable manner.
Traceability	NFR-03-03.

Detailed use cases refining the ViewClientProfile high-level use case:



User case name	ViewClientProfile
ID	UC-15
Participating actors	Initiated by Staff Communicates with Client, Human Information System
Flow of events	1. Shelter staff can choose to view a list of clients with detailed information. 2. The Human Information System display a list of overviews for all clients, such as id, name, phone number, address (include use case ViewClientOverview). 3. The Staff can view more details of one specific client. 4. The Human Information System display all of details about the one specific client (include use case ViewClientDetail).

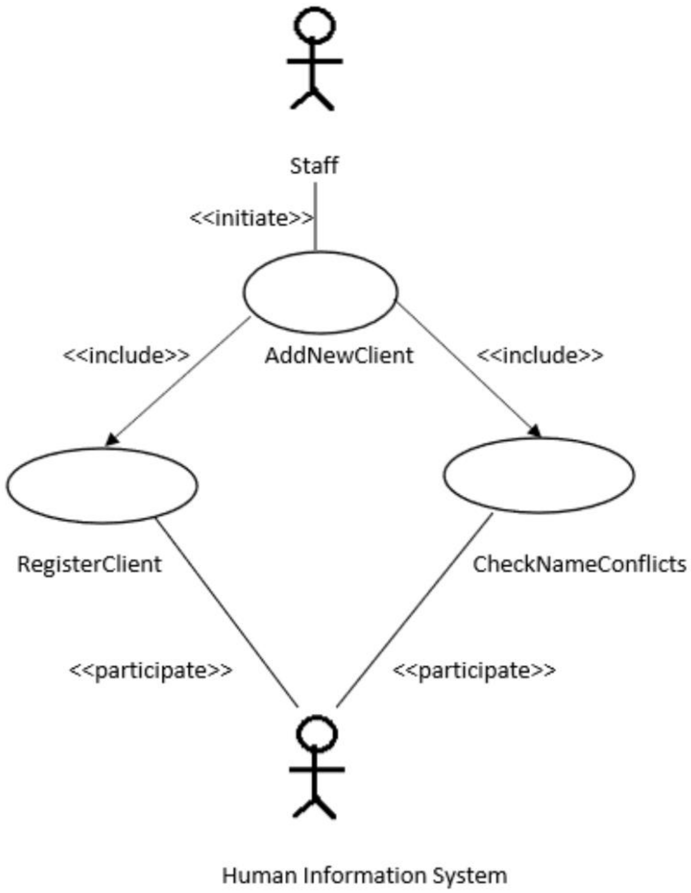
Entry condition	Shelter Staff choose to view a list of all existing clients and the detailed profile of a selected client.
Exit condition	Human Information System returns client profile and have completed actor's (staff) request.
Quality requirement	The system should provide a detailed display of the client's information.
Traceability	FR-05.

User case name	ViewClientOverview
ID	UC-16
Participating actors	Communicates with Staff
Flow of events	1. The Human Information System searches for all clients. 2. The Human Information System displays the list of overviews for all clients, such as id, name, phone number, address.
Entry condition	Shelter Staff choose to view a list of all existing clients and the detailed profile of a selected client.
Exit condition	Human Information System returns client profile and have completed the Staff's request.
Traceability	FR-05.

User case name	ViewClientDetail
ID	UC-17
Participating actors	Communicates with Staff
Flow of events	1. The Client Information System get the Staff's request. 2. The Animal Information System search specific one client and display all of information about the client, included the client's preference.
Entry condition	The Staff choose to one specific client to view details.

Exit condition	Human Information System returns client profile and have completed the Staff's request.
Traceability	FR-05.

Detailed use cases refining the AddNewClient high-level use case:



The detailed use case for AddNewClient:

User case name	AddNewClient
ID	UC-18
Participating actors	Initiated by Staff Communicates with HumanInformationSystem

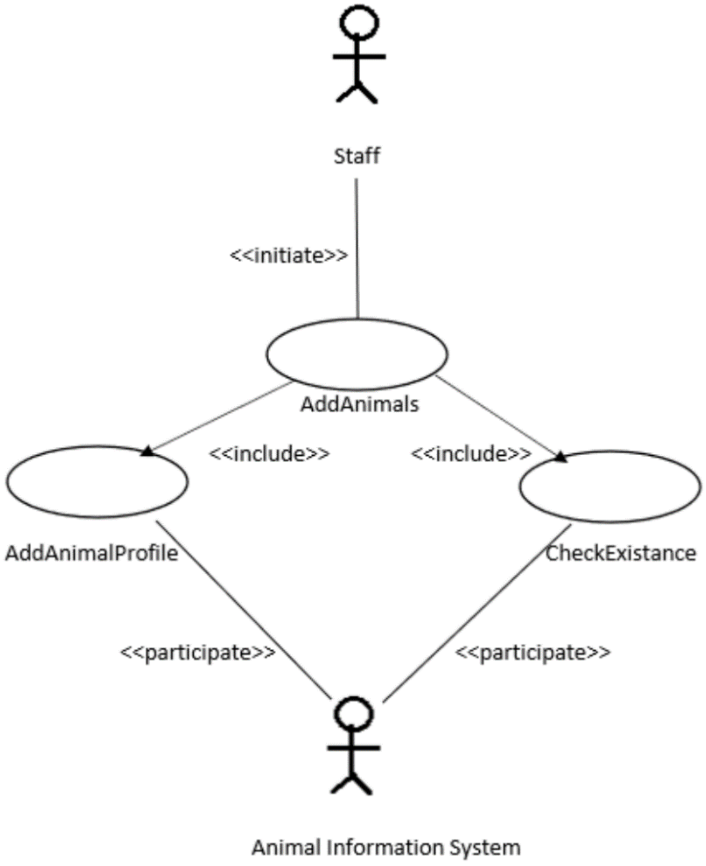
Flow of events	1. The Staff selects the AddClient option. 2. The Staff enters clients' information and confirms the input once done. (included the use case RegisterClient). 3. The Staff receives confirmation of completion. (included the use case CheckNameConflicts).
Entry condition	The Staff must be logged in to cuACS.
Exit condition	The Staff has successfully added the selected client.
Quality requirement	While they system is processing the Staff's request, a wait icon should be displayed.
Traceability	FR-06.

User case name	CheckNameConflicts
ID	UC-19
Participating actors	Initiated by Staff. Communicates with Human Information System.
Flow of events	1. The Staff submits a request to register a new client. 2. The system checks name conflicts. If not, the system presents the Staff with a registration form.
Entry condition	The Staff is logged into cuACS as Staff
Exit condition	The Staff receives confirmation message and is proceeded to the registration page.
Traceability	NFR-04-04.

User case name	RegisterClient
ID	UC-20
Participating actors	Initiated by Staff Communicates with HumanInformationSystem

Flow of events	1. The Staff requests the creation of a profile for a new client. 2. The system presents the Staff with a form. 3. The Staff specifies a name and date of birth. 4. The system prompts the Staff if more clients need to be added.
Entry condition	The Staff must be logged in to cuACS.
Exit condition	The new client is successfully registered
Traceability	FR-06.

Detailed use case for AddAnimals:

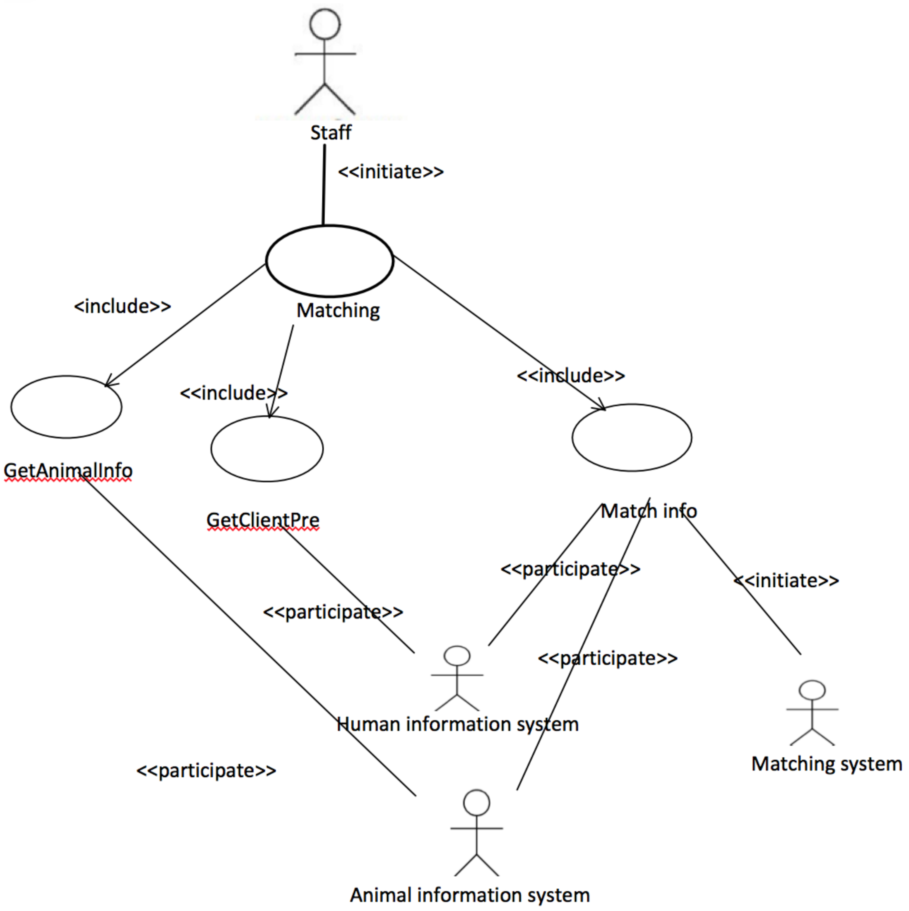


User case name	AddAnimals
ID	UC-21
Participating actors	Initiated by Staff Communicates with AnimalInformationSystem
Flow of events	1. The Staff selects the AddAnimal option. 2. The Staff enters animals' information and confirms the input once done (include use case AddAnimalProfile). 3. The Staff receives confirmation of completion.
Entry condition	The Staff is logged into cuACS as Staff
Exit condition	The Staff has successfully added the selected animal.
Quality requirement	While they system is processing the Staff's request, a wait icon should be displayed.
Traceability	FR-04-02.

User case name	AddAnimalProfile
ID	UC-22
Participating actors	Initiated by Staff. Communicates with Animal Information System.
Flow of events	1. The Staff requests the creation of a profile for the animal. 2. The system presents the Staff with a form. 3. The Staff specifies a name and type 4. The system prompts the Staff if more animals need to be added. 5. The Staff selects the "finish" option. 6. The system updates the animal list.
Entry condition	The Staff is logged into cuACS as Staff
Exit condition	The animal profile is successfully added to the Animal Information System.
Traceability	FR-04-02.

User case name	EditAnimalInfo
ID	UC-23
Participating actors	Initiated by Staff. Communicates with Animal Information System.
Flow of events	1.The Staff requests to edit an existing animal profile 2. The system presents the Staff with a list of all existing profiles. 3. The Staff selects the animal profile that needed to be edited. 4. The system present the Staff with the selected profile. 5. The Staff edits the profile and save once finished. 6. The system saves the edited profile and prompts the Staff.
Entry condition	The Staff is logged into cuACS as Staff
Exit condition	The animal profile is successfully added to the Animal Information System.
Traceability	FR-04-01.

Detailed use case for Matching:



Use case name	Matching
ID	UC-24
Participating actors	Initiated by Staff Communicates with Human information system, Animals information system and Matching system
Flow of events	1. The Staff runs the Matching system to start matching 2. The Human information system put the Client preference into Matching system (include GetClientPre). 3. The Animal information system put the Animal profile into Matching system (include GetAnimalInfo). 4. The Matching system estimate the score of matching condition between Animal profile and Client preference (MatchInfo). 5. The Matching system put the list of matched Animal profile which is sorted score by descending order (SortAnimal).
Entry condition	The Staff matches animal information and the Client's preference.
Exit conditions	The Animals information returns the sorted and matched animals along with their information.
Quality requirement	The system should be able to support Animal profile in Animals information and client preference in Human Information system.
Traceability	FR-07.

Name	GetAnimalInfo
ID	UC-25
Participating actors	Communicates with Animals information system and Matching system
Flow of events	1. When the staff request a Matching 2. The Animal Information system check whether the animal lists have any available animals. 3. The Animal Information output a list of available animals as return result.
Entry condition	The Staff is logged in to cuACS.

Exit Condition	There are available animals for the client to choose from.
Traceability	NFR-04-01.

Name	GetClientPre
ID	UC-26
Participating actors	Communicates with Personal Information System and Matching system.
Flow of events	1. The staff request a Matching. 2. The staff specifies a specific id and client name into the Human Information System. 3. The system checks if the client exists. 4.The system located the client's preferences as return value.
Entry condition	The Staff is logged in to cuACS.
Exit Condition	The Client is already added in the account.
Traceability	NFR-04-01.

Name	MatchInfo
ID	UC-27
Participating actors	Communicates with Human information system, Animals information system and Matching system.

Flow of events	1. The system already get client's preference from the Human Information system and the list of animals' information form the Animal information system. 2. The system compare each client preference with animal's information. 3. The system ask the staff to select preference to compare. 4 if the system gets matched one preference then plus one score into the score calculator and mark the preference label with the animal. 5.if the client's preference didn't match any animals, return the command. 6.Since the system finished, the system will get the animal's matching score.
Entry condition	The Staff is logged in to cuACS.
Exit Condition	The matching between animals and the client's preference is successful.
Traceability	FR-07.

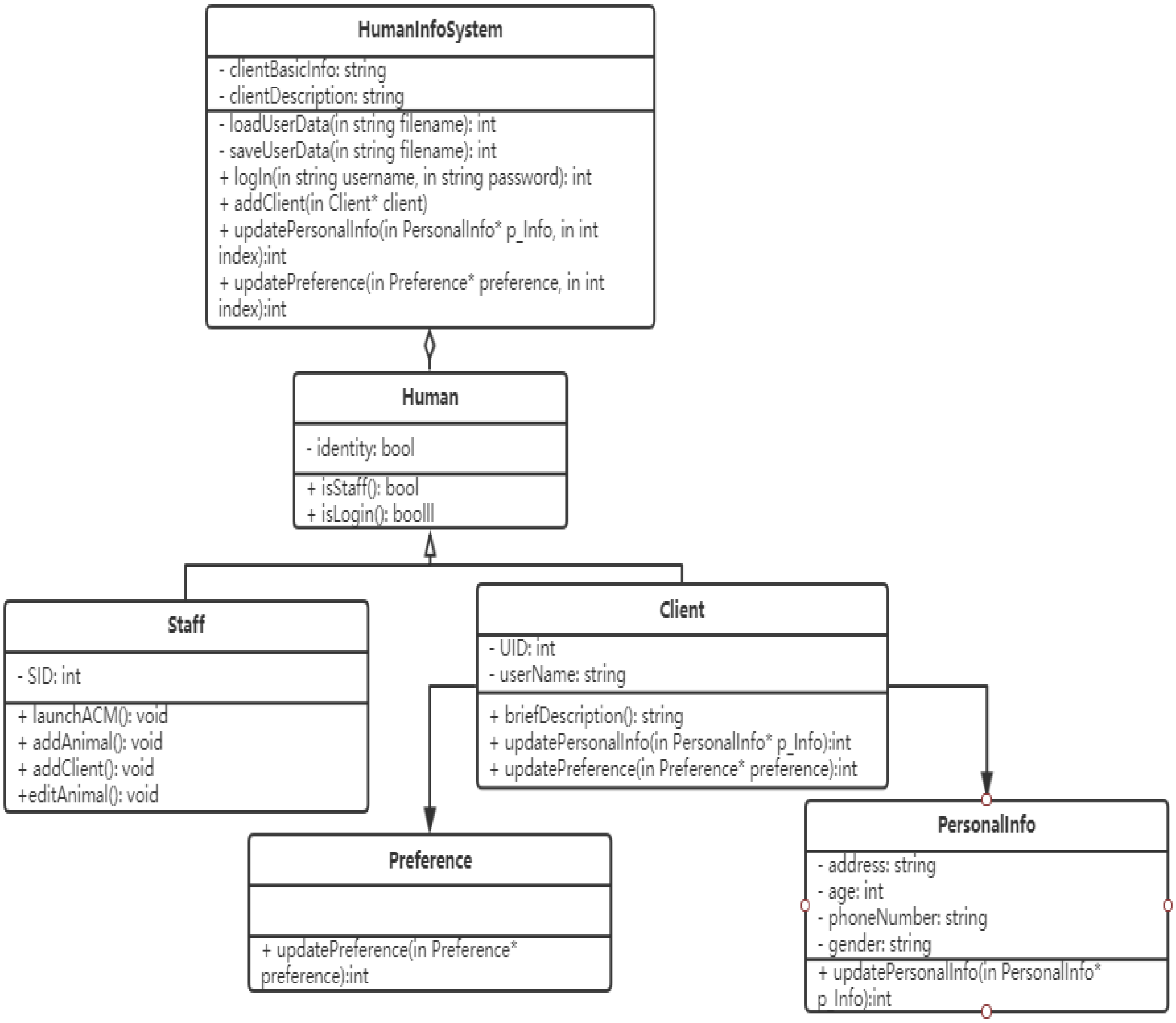
5. Object Model

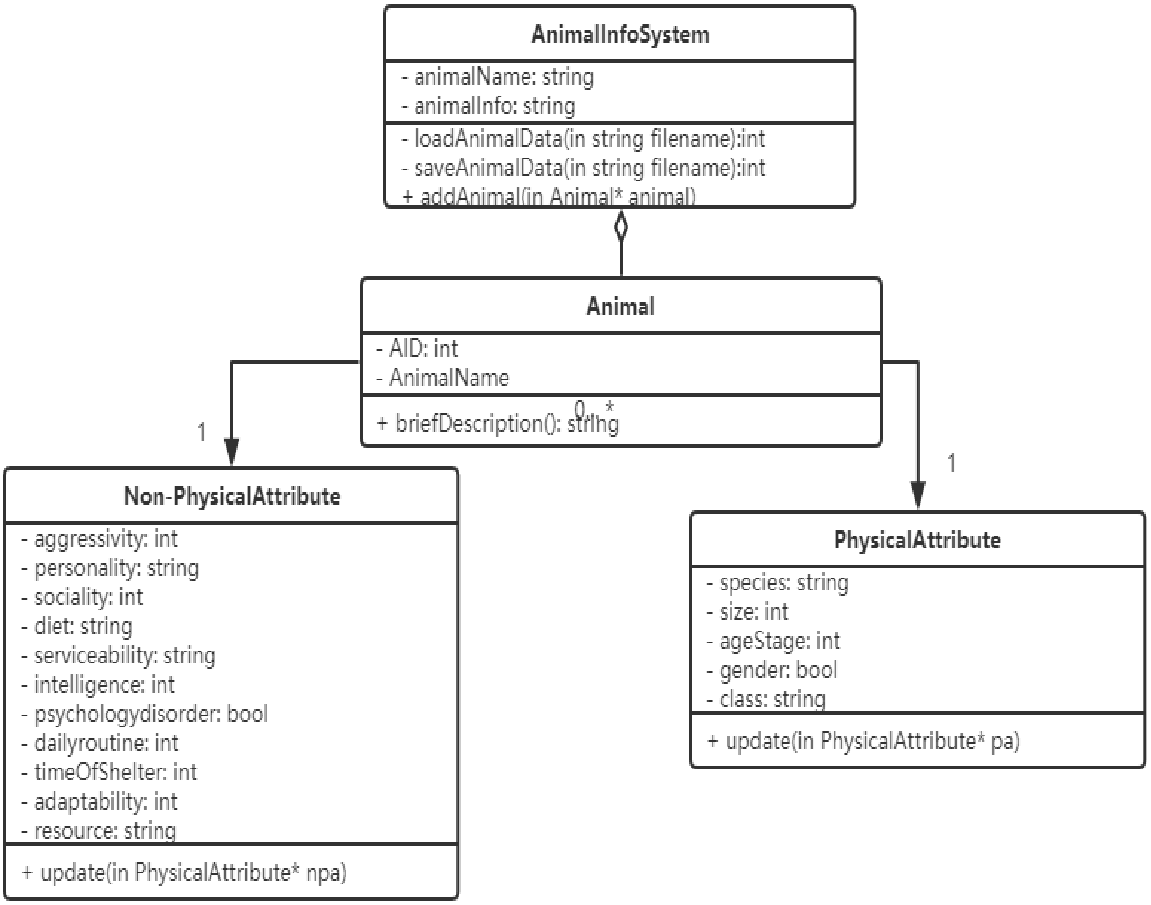
5.1. Data Dictionary

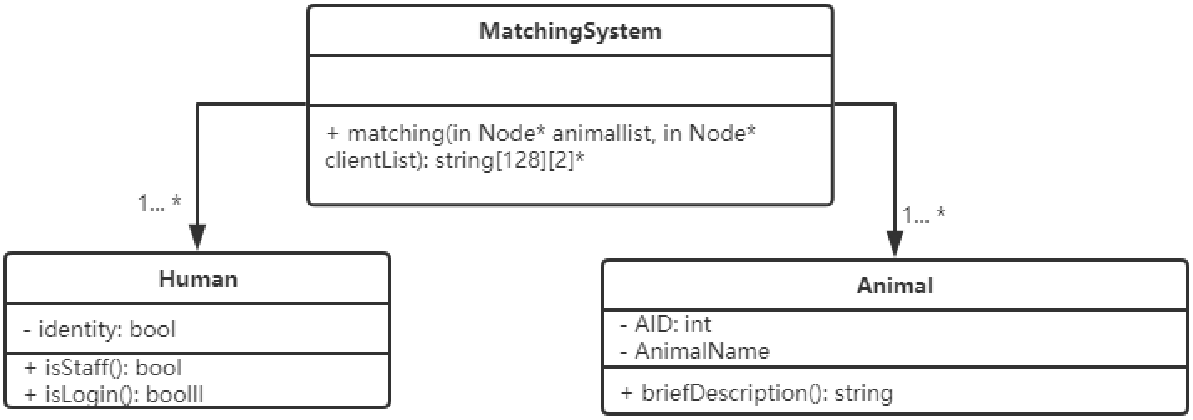
Entity Object	Attributes & Associations	Definition	Use Case Traceability
1. Animal	• animal ID • animal name • physical attribute • non-physical Attribute	An animal is a living organism that feeds on organic matter. In cuACS, the term Animal refers to shelter animals waiting for adoption.	UC-12, 13, and 14
2. AnimalInformation System	• animal profile • list of animals	An instance of the local database management system that holds and manages animals' information.	UC-12, 13,14 and UC-21 to UC-27
3. cuACS		An instantiation of the cuACS system.	

4. Client	• client ID • client name • personal info • preference	Actor interested in adopting animals using cuACS software system.	UC-01 to UC-11
5. Human	• identity	An abstraction of the cuACS users, including Staff and Client.	UC-01 to UC-11
6. HumanInformation System	• client profile • list of clients	An instance of the local database management system that holds and manages human clients' information.	UC-01 to UC-11
7. MatchingAlgorithm	• list of animal profiles • list of client preferences	A method that computes the optimal set of animal-client matches.	UC-25 to UC-27
8. MatchingSystem	• list of client profiles • list of animal profiles • matching algorithm	A sub-system of the cuACS that enforces a set of rules to assess animal-client compatibility.	UC-25 to UC-27
9. Staff	• staff identity	The actor responsible for managing animal and clients' profile information, as well as launching the MatchingAlgorithm.	UC-12 to UC-27

5.2. Class Diagram


HumanInfoSystem
- clientBasicInfo: string
- clientDescription: string
- loadUserData(in string filename): int
- saveUserData(in string filename): int
+ logIn(in string username, in string password): int
+ addClient(in Client* client)
+ updatePersonalInfo(in PersonalInfo* p_Info, in int index):int
+ updatePreference(in Preference* preference, in int index):int
Human
- identity: bool
+ isStaff(): bool
+ isLogin(): boolll
Staff
- SID: int
+ launchACM(): void
+ addAnimal(): void
+ addClient(): void
+editAnimal(): void
Client
- UID: int
- userName: string
+ briefDescription(): string
+ updatePersonalInfo(in PersonalInfo* p_Info):int
+ updatePreference(in Preference* preference):int
Preference
+ updatePreference(in Preference* preference):int
PersonalInfo
- address: string
- age: int
- phoneNumber: string
- gender: string
+ updatePersonalInfo(in PersonalInfo* p_Info):int


AnimalInfoSystem
- animalName: string
- animalInfo: string
- loadAnimalData(in string filename):int
- saveAnimalData(in string filename):int
+ addAnimal(in Animal* animal)
Animal
- AID: int
- AnimalName
+ briefDescription(): string
0..*
1
1
Non-PhysicalAttribute
- aggressivity: int
- personality: string
- sociality: int
- diet: string
- serviceability: string
- intelligence: int
- psychologydisorder: bool
- dailyroutine: int
- timeOfShelter: int
- adaptability: int
- resource: string
+ update(in PhysicalAttribute* npa)
PhysicalAttribute
- species: string
- size: int
- ageStage: int
- gender: bool
- class: string
+ update(in PhysicalAttribute* pa)


MatchingSystem
+ matching(in Node* animallist, in Node* clientList): string[128][2]*
1... *
1... *
Human
- identity: bool
+ isStaff(): bool
+ isLogin(): boolll
Animal
- AID: int
- AnimalName
+ briefDescription(): string