Final System Analysis & Design Project

Group 2: Mason Allen, Daniel Edwards, Horacio Torres, Matthew Bratrsovsky

BACS 487: Systems Analysis and Design

Team

Name	Roles	Responsibilities	
Mason Allen	Project Manager UI/UX Design	Documentation	Work Assignments - Trello
		CSS: Bootstrap	
Daniel Edwards	Network Architect	PythonAnywhere	Python, Django
	Web Developer	CSS: Bootstrap	
Horacio Torres	Business Analyst	Change management	
	Security Engineer	NGINX, Cloudflare	AUTH0
Matthew Bratrsovsky	Database architect	MYSQL	Views
	UI/UX Design	FlutterUI	
	Mobile Apps	Dart	Flutter

Project / System Request

SPONSER

Debi Bratrsovsky, Owner of All About Paws Pet Spa LLC

https://www.fortmorgangrooming.com

Need

The owner of All About Paws Pet Spa (hereby abbreviated AAPPS) requires a new system which allows for quicker integration into modern business technologies. This includes a mobile application and website that directly integrate into a central database. The primary question being answered by this system is how to increase a service-sector's repeat business with smart applications.

Due to increased business and clients a new scheduling application is required, in order to speed up efficiency in the scheduling process. This will allow clients to schedule an appointment themselves with an application, based on factors such as job size and the business owner's availability, rather than the current manual scheduling. Employees or authorized users will also have access to see the full schedules created on the application.

It is also of note that by integrating information technology, valuable statistics and marketing information will be obtained through this application. Whereas the current system is entirely manual, the new system will allow for tracking clients via inputted data and emphasize follow-up services, both complementary and repeat.

Requirements

The functionality of the application for the clients should follow these requirements:

- Maintain records of clients in system
- Maintain accounts in system
- Maintain scheduling available timeslots
- Maintain records for completed appointments
- Allow for client self-registration for appointments
- Allow clients to see scheduling history
- Add appointment requests

Additionally, the functionality of the application for the Employees should:

- View full schedule of all made and past appointments
- Ability to cancel made appointment
- Have access to client accounts
- Add or edit appointment times or schedule
- See appointment requests

Value

It can be both incredibly frustrating and time-consuming to manually schedule many clients. Furthermore, keeping track of every client and uniquely marketing to their needs to drive repeat business is even more difficult. Therefore, introducing information technology to help ease the burden will release these burdens and allow the owners of service-sector businesses to focus on what they do best: delivering an excellent service. This system will not only cut back on busy work, but also increase profits resulting from better team transparency. With this, more time and resources can be put towards the client and their service.

Level of Effort

XL: 40 hours; Large: 20 hours; Medium: 14 hours; Small: 10 hours

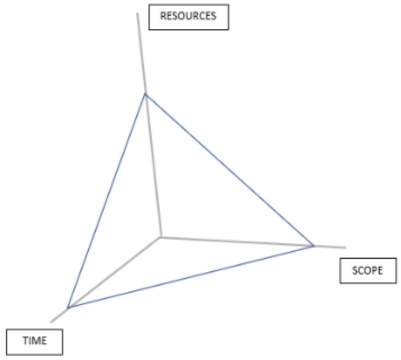
Task	Size	Total	Running Total
Design/Layout			
Customer Facing Pages			**64**
Landing	Medium	16	
Pricing	Small	8	
Login	Medium	16	
Sign up	Large	24	
Database			**216**
Client Table	Medium	16	
Client Table Views	Large	24	
Pet Table	Medium	16	
Pet Table Views	Large	24	
Reference Table	Medium	16	
Reference Table Views	Large	24	
Employee Table	Medium	16	
Employee Table Views	Large	24	
Other Tables	Large	24	
Other Table Views	XL	32	
Forms			***56***
Create Forms			**24**
New client form	Small	8	
New grooming agreement form	Small	8	
Client sheet card form	Small	8	
Rig Forms			**32**
Print all new forms	Medium	16	
Print specific form	Medium	16	

Task	Size	Total	Running Total
Logic			192
Maintain scheduling available timeslots	Small	8	
Web			120
Allow for client self-registration for appointments	Large	24	
Allow clients to see scheduling history	Medium	16	
Request Appointment	Large	24	
View full schedule of all made and past appointments	Medium	16	
Ability to cancel made appointment	Medium	16	
Rig email server	Large	24	
Flutter			64
Ability to cancel made appointment	Small	8	
Ability to close account	Medium	16	
Ability to reopen account	Medium	16	
Ability to auto-send emails	Large	24	
Deploy			92
Deploy Android App	Medium	16	
Deploy IPhone App	Medium	16	
Deploy Web App	Medium	16	
Documentation	Large	24	
Change Management	Large	20	
Total			
Small	6	48	
Medium	16	256	
Large	12	288	
XL	1	8	
Total	**35**	**600**	

600 hours / 12 weeks / 4 group members * 1.25 buffer room = 16 hours / week

742 Hours / 10 Weeks / 4 group members = 18.55 hours per week

Constraints



The majority of the resources will be allocated to time, followed by scope, and finally resources. As android is open source and the website can be hosted by GitHub, there is little cost of ownership and no expected licensing costs.

Feasibility Analysis

Technical

The primary deliverable for this project is an android-based app with various views of the same data; the website is a complementary deliverable should time permit and is not vital to the final product. Although the final deliverable is not expected to elapse the constrained 15-week development period, the team has little development time inside of Android studio and this has the potential to throw a wrench into the mix. The core system's process is not projected to be overly complex, which should ease the technical burden.

Economic

The ROI is measured as follows: (Total Benefits – Total cost) / Total Cost. The ROI can be further illustrated by building a simple cash flow projection as the one shown below. In the end, the net cost for all 3 years can be divided by the cost for all 3 years in order to attain our ROI value.

	2021 (Year 0)	2022 (Year 1)	2023 (Year 2)
Total Benefits			
- *Increased client return rate	(No benefits	$4000	$3000
- *Less employee interaction needed	year 0)	$0	$0
- *Increase in appointment booking		$4500	$3500
Dev Cost & Maintenance			
- Hardware	$2000	$500	$500

- Development Labor - Software (building and defects)	$0 $1000	$500 $500	$500 $500
Net (Benefits – Cost)	-$3000	$7000	$5000
ROI Net (Year 0-2) / Cost (Year 0-2)	$15000/$6000 = 2.5%		

Intangibles

An increase in sales can come because of the returning costumer rate increasing as well as higher client satisfaction because of the newly built system

A new and reliable system that allows clients to book appointments with ease shows the importance of client satisfaction. This can help the business grow (increase market share) and improve its reputation and brand recognition.

Organization

As an organization, we have all the non-financial resources we need to succeed in building our application. While some of the group's technical skills may be lacking inside Android Studio, we have many professors and collages who can help us in times of need. Management prowess is very high, as this project needs to be finished and reliable by the due date. While the management experience is not very high, everyone in the team is passionate about getting this application on the market. This project will get very high attention not just from management, but from the whole team. Once the application is made, clients will be incentivized to schedule appointments as it will improve productivity for both the client and employees.

Project Plan

The project must be completed within 15 weeks (one typical school semester), be relatively stable to prevent missed appointments, and be built on unfamiliar technology. However, the requirements are clear and relatively simple. The most ideal methodology for developing this system is therefore the iterative model, such as rapid application development (RAD).

Analysis

Requirement's Definition Statement

System Objective:

The objective of our scheduling system is to improve the pet grooming business in more ways than one. The new scheduling assistant will increase overall business, will be more efficient than the current scheduling system, and will specialize in meeting the clients' needs. This will also introduce more modern technology into the business.

Interview Questions:

The following questions were asked to gather a better understanding of the system to be built. Two core takeaways from the interview are the necessity of the system not surprising the groomers with unanticipated clients and the necessity for change management in the final stretch of the project to ensure that the owner and staff are fully aware of both the benefits and limitations of the new system.

- With the current system, how does a customer make an appointment?
  - 90% of clients call to make an appointment, and 10% send in text messages. Currently, no clients email to schedule appointments. This will be a primary source of needed change management.
- Should people have an account to schedule?
  - Yes. New clients are registered by phone number online and approved by the secretary according to shop preferences such as pet type, account location, and account blacklist. After being approved, new clients can request future appointments.
- How many appointments do you usually have scheduled at any given time?
  - 6:30 – 1pm. Show booked appointments through 5:00pm. This will be implemented by returning available times which can be scheduled based on the client’s pet characteristics such as size, breed, and weight. Full schedules will not be presented so as to protect the privacy of the shop.

- What are some pros and cons changing from a manual system to an application
  - Con – not enough time to complete new animal (new customers)

    Con- Fear of the unknown, new problems and working could arise
  - Pro – Streamline straight to scheduling

    Pro – Could ease work load, taking away manual scheduling time

Functional Requirements:

1. Scheduling (Client)

   1.1. The application will allow customers to create an account and schedule appointments.

   1.2. The application will determine if the requested timeslot is available.

   1.3. The application will display the available times

   1.4. The application can receive user requests

   1.5. The application can display all current and past appointments

2. Employee

   2.1. The application can display all accounts

   2.2. The application can display all past appointments

   2.3. The application can cancel appointments manually

   2.4. The application can edit appointment times or requests manually

Non-Functional Requirements:

1. Operational

   1.1. The application should run on Android devices used by customers and management.

   1.2. The application should be web-based and run on any browser so that clients can easily access and request appointments.

   1.3. The application must not crash

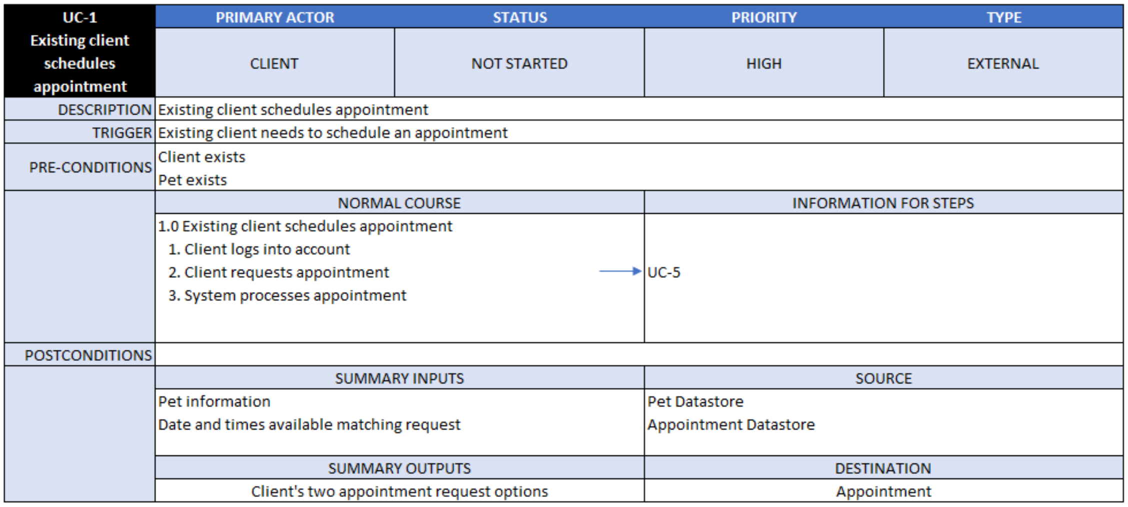
2. Performance
    2.1. The performance should provide response time of less than 5 seconds.
    2.2. The system should update new scheduling appointments every time one is inputted.
    2.3. The system should save account and pet information

3. Security
    3.1. Client accounts and information should be maintained securely
    3.2. Client must make unique password with special characters

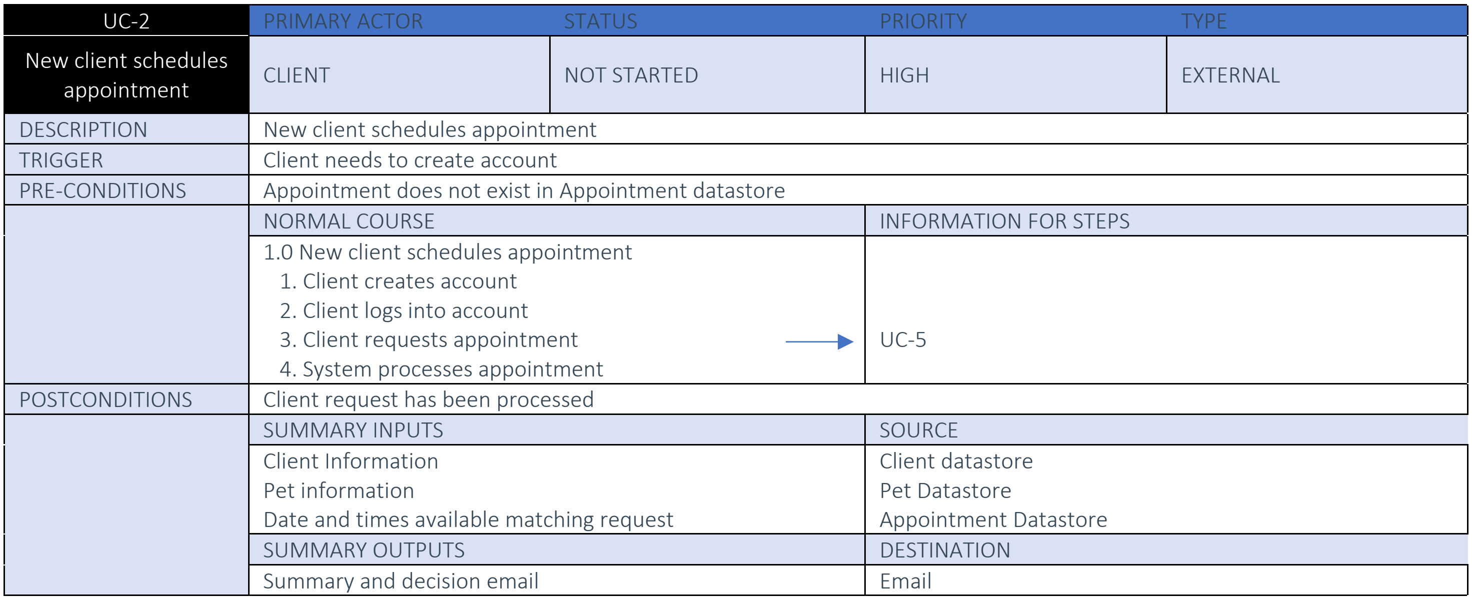
4. Political and Cultural
    4.1. Application is available for Apple and Android products
    4.2. The system must not allow for unanticipated clients or pets to be scheduled and approved.
        4.2.1. Messaging must be clear with both prospective and existing clientel.

Use Cases

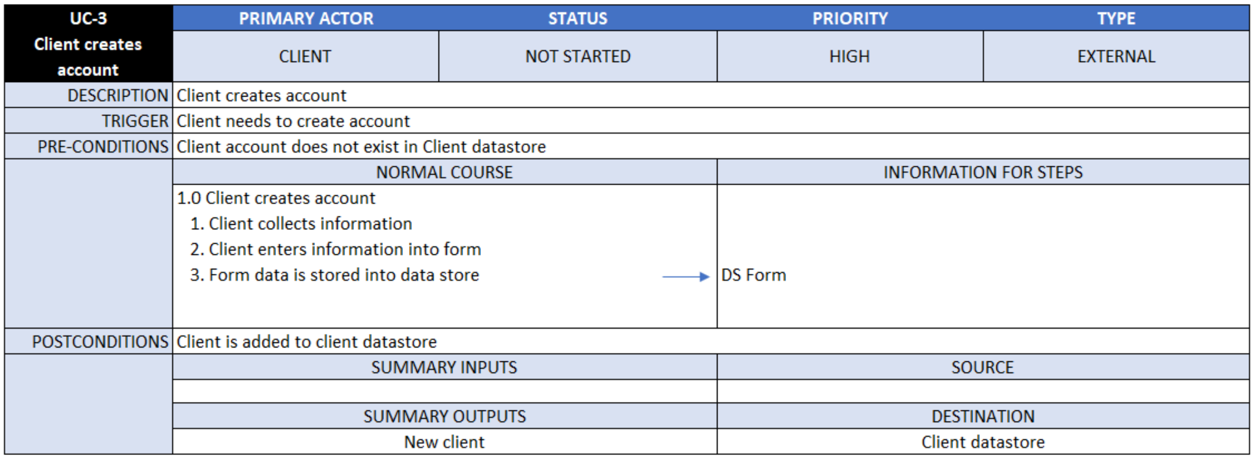
The following use cases were primarily built to stand alone as modular components of the system without extensive pre-requirements. However, UC-5 is the most intensive of the use cases and involves multiple modules.



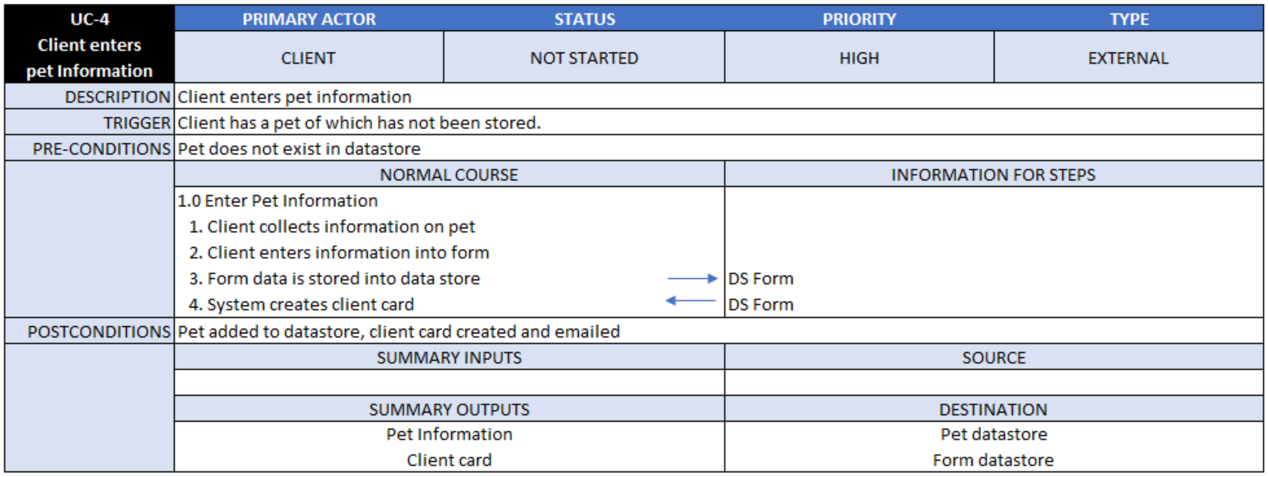
UC-1 Existing client schedules appointment	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	CLIENT	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	Existing client schedules appointment			
TRIGGER	Existing client needs to schedule an appointment			
PRE-CONDITIONS	Client exists Pet exists			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Existing client schedules appointment 1. Client logs into account 2. Client requests appointment → 3. System processes appointment		UC-5	
POSTCONDITIONS				
	SUMMARY INPUTS		SOURCE	
	Pet information Date and times available matching request		Pet Datastore Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Client's two appointment request options		Appointment	



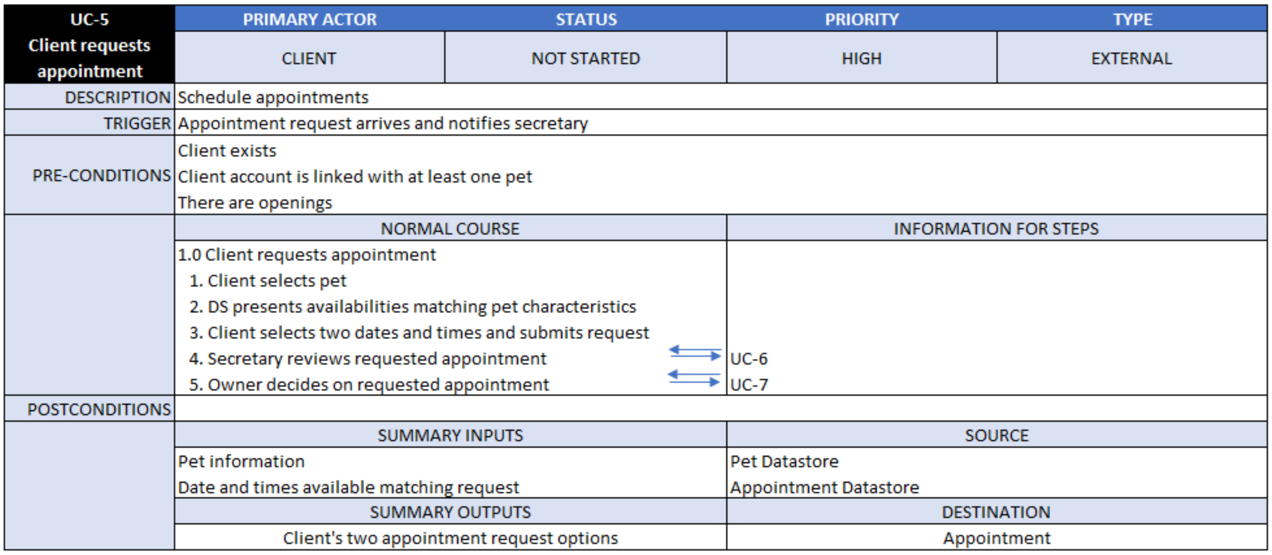
UC-2 New client schedules appointment	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	CLIENT	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	New client schedules appointment			
TRIGGER	Client needs to create account			
PRE-CONDITIONS	Appointment does not exist in Appointment datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 New client schedules appointment 1. Client creates account 2. Client logs into account 3. Client requests appointment → 4. System processes appointment		UC-5	
POSTCONDITIONS	Client request has been processed			
	SUMMARY INPUTS		SOURCE	
	Client Information Pet information Date and times available matching request		Client datastore Pet Datastore Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Summary and decision email		Email	



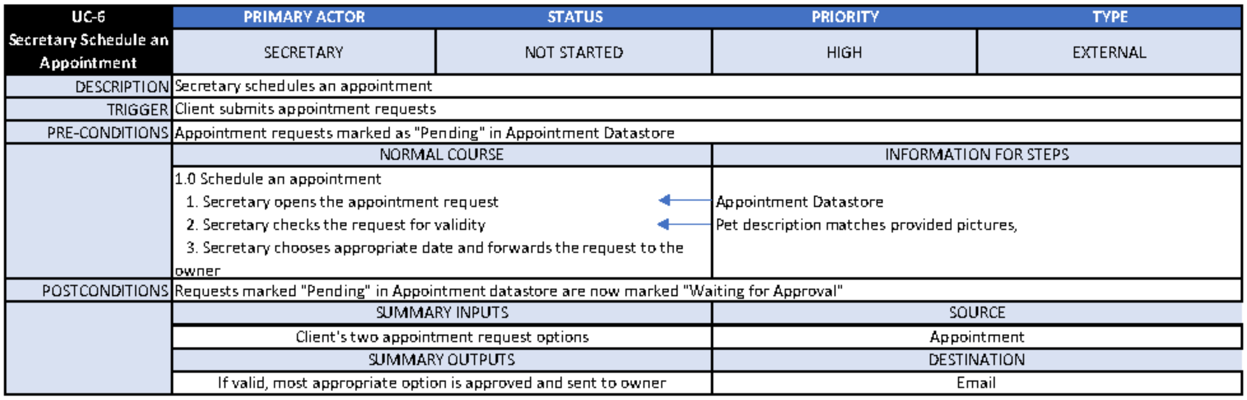
UC-3 Client creates account	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	CLIENT	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	Client creates account			
TRIGGER	Client needs to create account			
PRE-CONDITIONS	Client account does not exist in Client datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Client creates account			
	1. Client collects information			
	2. Client enters information into form			
	3. Form data is stored into data store	→	DS Form	
POSTCONDITIONS	Client is added to client datastore			
	SUMMARY INPUTS		SOURCE	
	SUMMARY OUTPUTS		DESTINATION	
	New client		Client datastore	



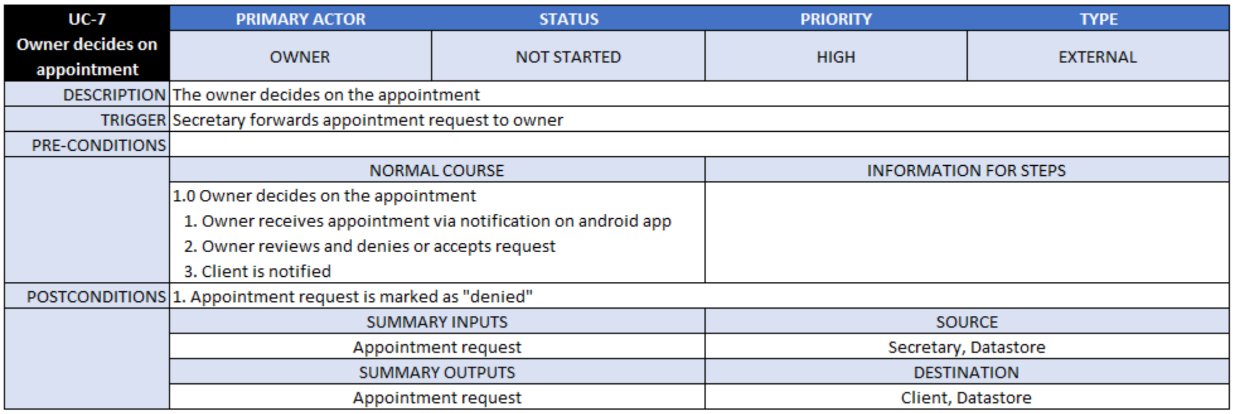
UC-4 Client enters pet Information	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	CLIENT	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	Client enters pet information			
TRIGGER	Client has a pet of which has not been stored.			
PRE-CONDITIONS	Pet does not exist in datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Enter Pet Information			
	1. Client collects information on pet			
	2. Client enters information into form			
	3. Form data is stored into data store	→	DS Form	
	4. System creates client card	←	DS Form	
POSTCONDITIONS	Pet added to datastore, client card created and emailed			
	SUMMARY INPUTS		SOURCE	
	SUMMARY OUTPUTS		DESTINATION	
	Pet Information Client card		Pet datastore Form datastore	



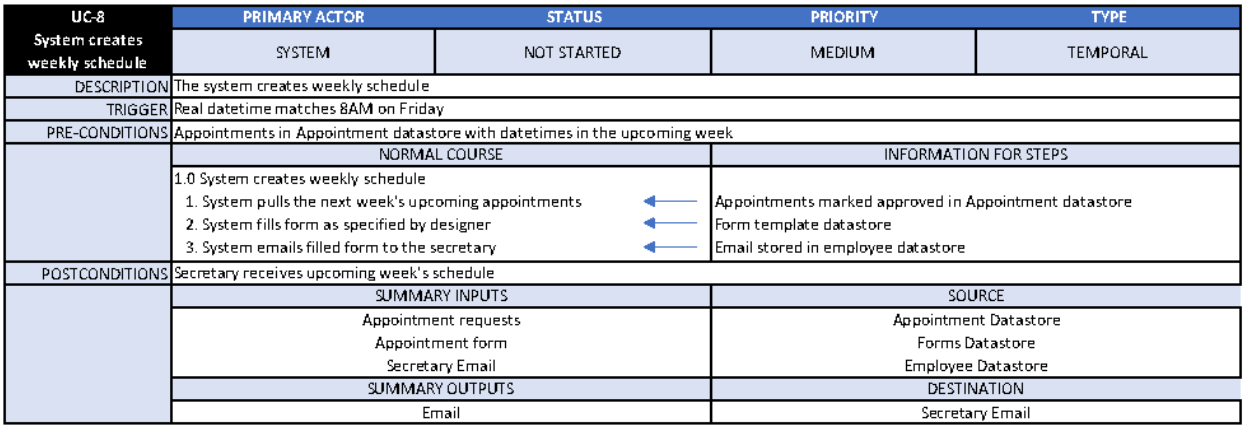
UC-5 Client requests appointment	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	CLIENT	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	Schedule appointments			
TRIGGER	Appointment request arrives and notifies secretary			
PRE-CONDITIONS	Client exists Client account is linked with at least one pet There are openings			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Client requests appointment			
	1. Client selects pet			
	2. DS presents availabilities matching pet characteristics			
	3. Client selects two dates and times and submits request			
	4. Secretary reviews requested appointment	⇄	UC-6	
	5. Owner decides on requested appointment	⇄	UC-7	
POSTCONDITIONS				
	SUMMARY INPUTS		SOURCE	
	Pet information Date and times available matching request		Pet Datastore Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Client's two appointment request options		Appointment	



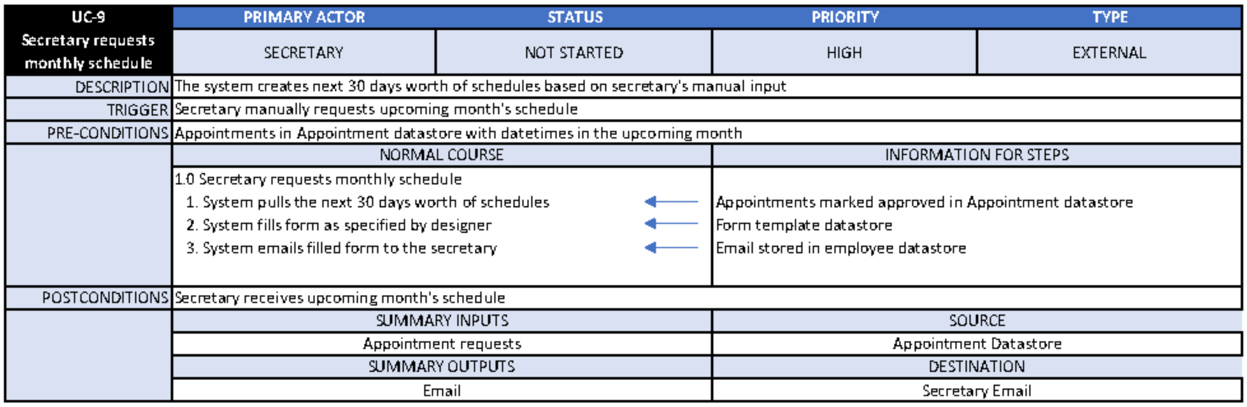
UC-6 Secretary Schedule an Appointment	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	Secretary schedules an appointment			
TRIGGER	Client submits appointment requests			
PRE-CONDITIONS	Appointment requests marked as "Pending" in Appointment Datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Schedule an appointment			
	1. Secretary opens the appointment request ←		Appointment Datastore	
	2. Secretary checks the request for validity ←		Pet description matches provided pictures,	
	3. Secretary chooses appropriate date and forwards the request to the owner			
POSTCONDITIONS	Requests marked "Pending" in Appointment datastore are now marked "Waiting for Approval"			
	SUMMARY INPUTS		SOURCE	
	Client's two appointment request options		Appointment	
	SUMMARY OUTPUTS		DESTINATION	
	If valid, most appropriate option is approved and sent to owner		Email	



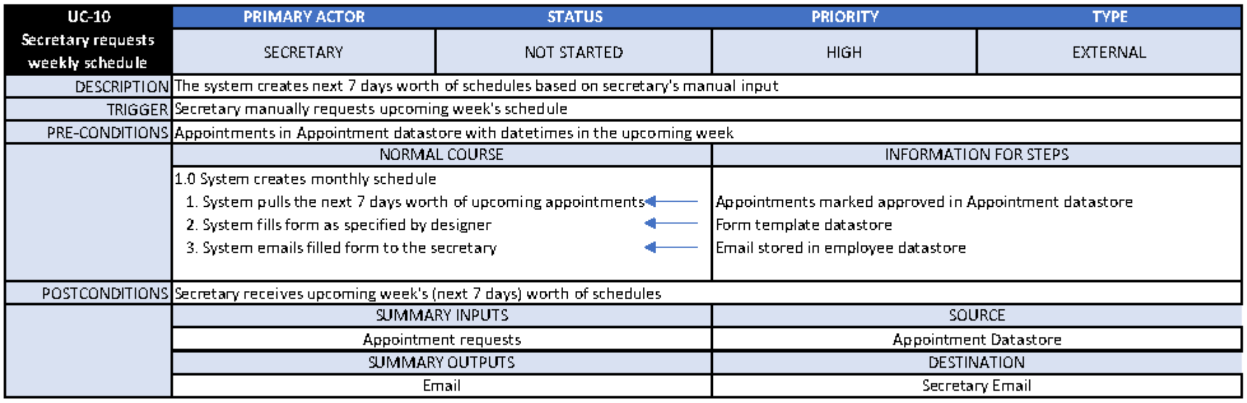
UC-7 Owner decides on appointment	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	OWNER	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	The owner decides on the appointment			
TRIGGER	Secretary forwards appointment request to owner			
PRE-CONDITIONS				
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Owner decides on the appointment 1. Owner receives appointment via notification on android app 2. Owner reviews and denies or accepts request 3. Client is notified			
POSTCONDITIONS	1. Appointment request is marked as "denied"			
	SUMMARY INPUTS		SOURCE	
	Appointment request		Secretary, Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Appointment request		Client, Datastore	



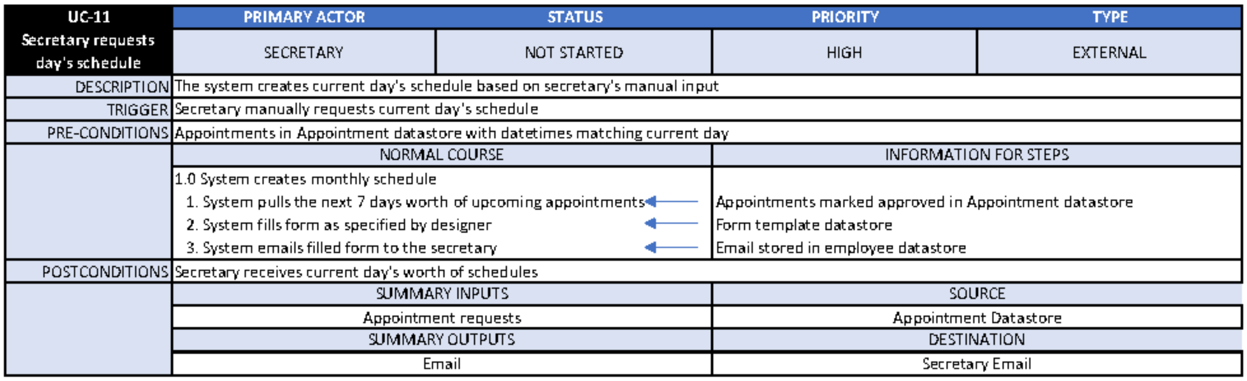
UC-8 System creates weekly schedule	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SYSTEM	NOT STARTED	MEDIUM	TEMPORAL
DESCRIPTION	The system creates weekly schedule			
TRIGGER	Real datetime matches 8AM on Friday			
PRE-CONDITIONS	Appointments in Appointment datastore with datetimes in the upcoming week			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 System creates weekly schedule			
	1. System pulls the next week's upcoming appointments ←		Appointments marked approved in Appointment datastore	
	2. System fills form as specified by designer ←		Form template datastore	
	3. System emails filled form to the secretary ←		Email stored in employee datastore	
POSTCONDITIONS	Secretary receives upcoming week's schedule			
	SUMMARY INPUTS		SOURCE	
	Appointment requests Appointment form Secretary Email		Appointment Datastore Forms Datastore Employee Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Secretary Email	



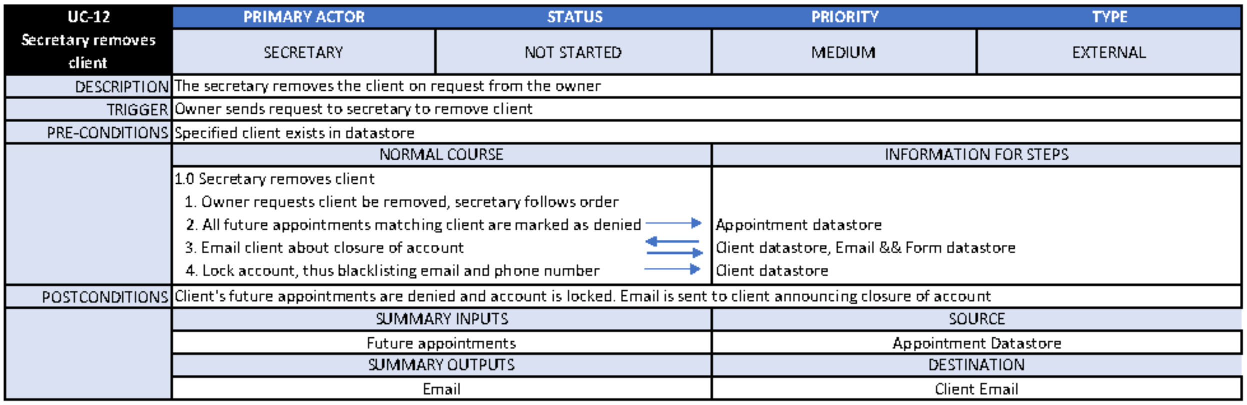
UC-9 Secretary requests monthly schedule	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	The system creates next 30 days worth of schedules based on secretary's manual input			
TRIGGER	Secretary manually requests upcoming month's schedule			
PRE-CONDITIONS	Appointments in Appointment datastore with datetimes in the upcoming month			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Secretary requests monthly schedule			
	1. System pulls the next 30 days worth of schedules ←		Appointments marked approved in Appointment datastore	
	2. System fills form as specified by designer ←		Form template datastore	
	3. System emails filled form to the secretary ←		Email stored in employee datastore	
POSTCONDITIONS	Secretary receives upcoming month's schedule			
	SUMMARY INPUTS		SOURCE	
	Appointment requests		Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Secretary Email	



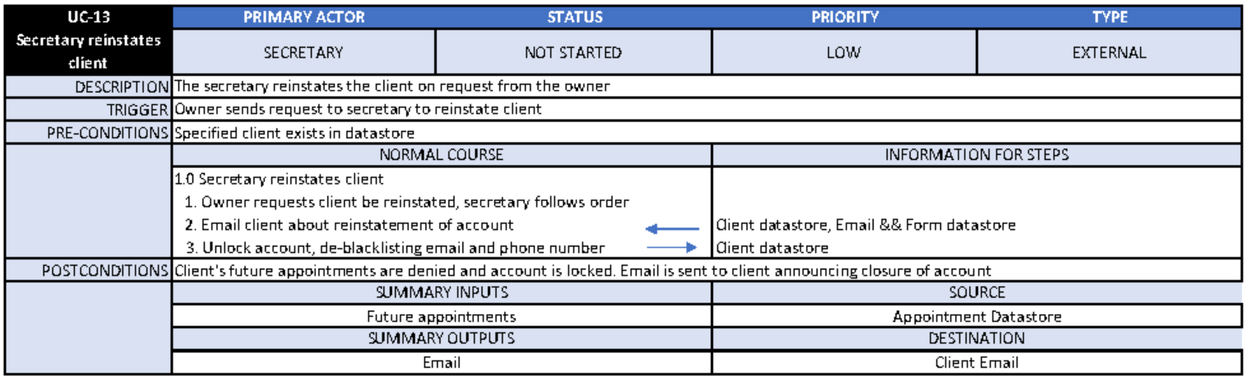
UC-10 Secretary requests weekly schedule	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	The system creates next 7 days worth of schedules based on secretary's manual input			
TRIGGER	Secretary manually requests upcoming week's schedule			
PRE-CONDITIONS	Appointments in Appointment datastore with datetimes in the upcoming week			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 System creates monthly schedule			
	1. System pulls the next 7 days worth of upcoming appointments ←		Appointments marked approved in Appointment datastore	
	2. System fills form as specified by designer ←		Form template datastore	
	3. System emails filled form to the secretary ←		Email stored in employee datastore	
POSTCONDITIONS	Secretary receives upcoming week's (next 7 days) worth of schedules			
	SUMMARY INPUTS		SOURCE	
	Appointment requests		Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Secretary Email	



UC-11 Secretary requests day's schedule	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	HIGH	EXTERNAL
DESCRIPTION	The system creates current day's schedule based on secretary's manual input			
TRIGGER	Secretary manually requests current day's schedule			
PRE-CONDITIONS	Appointments in Appointment datastore with datetimes matching current day			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 System creates monthly schedule			
	1. System pulls the next 7 days worth of upcoming appointments ←		Appointments marked approved in Appointment datastore	
	2. System fills form as specified by designer ←		Form template datastore	
	3. System emails filled form to the secretary ←		Email stored in employee datastore	
POSTCONDITIONS	Secretary receives current day's worth of schedules			
	SUMMARY INPUTS		SOURCE	
	Appointment requests		Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Secretary Email	



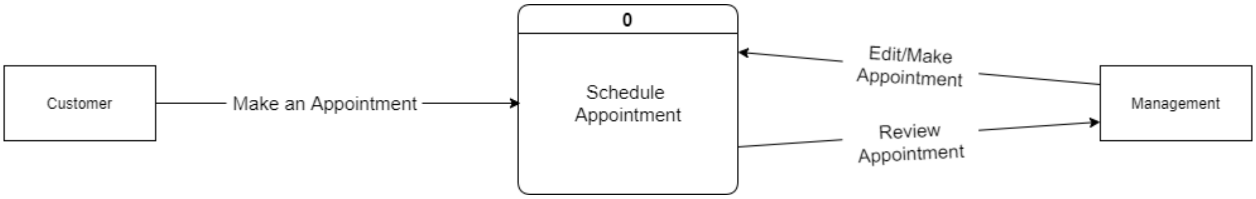
UC-12 Secretary removes client	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	MEDIUM	EXTERNAL
DESCRIPTION	The secretary removes the client on request from the owner			
TRIGGER	Owner sends request to secretary to remove client			
PRE-CONDITIONS	Specified client exists in datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Secretary removes client			
	1. Owner requests client be removed, secretary follows order			
	2. All future appointments matching client are marked as denied →		Appointment datastore	
	3. Email client about closure of account ⇄		Client datastore, Email && Form datastore	
	4. Lock account, thus blacklisting email and phone number →		Client datastore	
POSTCONDITIONS	Client's future appointments are denied and account is locked. Email is sent to client announcing closure of account			
	SUMMARY INPUTS		SOURCE	
	Future appointments		Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Client Email	



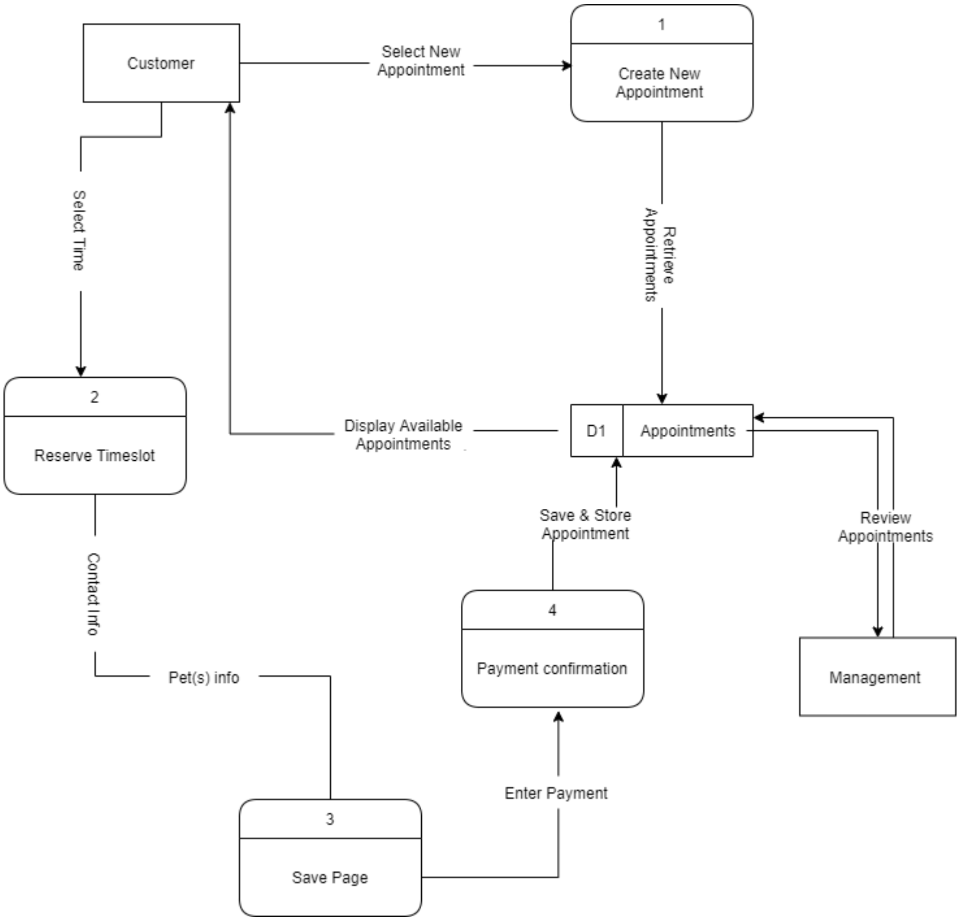
UC-13 Secretary reinstates client	PRIMARY ACTOR	STATUS	PRIORITY	TYPE
	SECRETARY	NOT STARTED	LOW	EXTERNAL
DESCRIPTION	The secretary reinstates the client on request from the owner			
TRIGGER	Owner sends request to secretary to reinstate client			
PRE-CONDITIONS	Specified client exists in datastore			
	NORMAL COURSE		INFORMATION FOR STEPS	
	1.0 Secretary reinstates client			
	1. Owner requests client be reinstated, secretary follows order			
	2. Email client about reinstatement of account ←		Client datastore, Email && Form datastore	
	3. Unlock account, de-blacklisting email and phone number →		Client datastore	
POSTCONDITIONS	Client's future appointments are denied and account is locked. Email is sent to client announcing closure of account			
	SUMMARY INPUTS		SOURCE	
	Future appointments		Appointment Datastore	
	SUMMARY OUTPUTS		DESTINATION	
	Email		Client Email	

Process Models

Level 0 DFD:

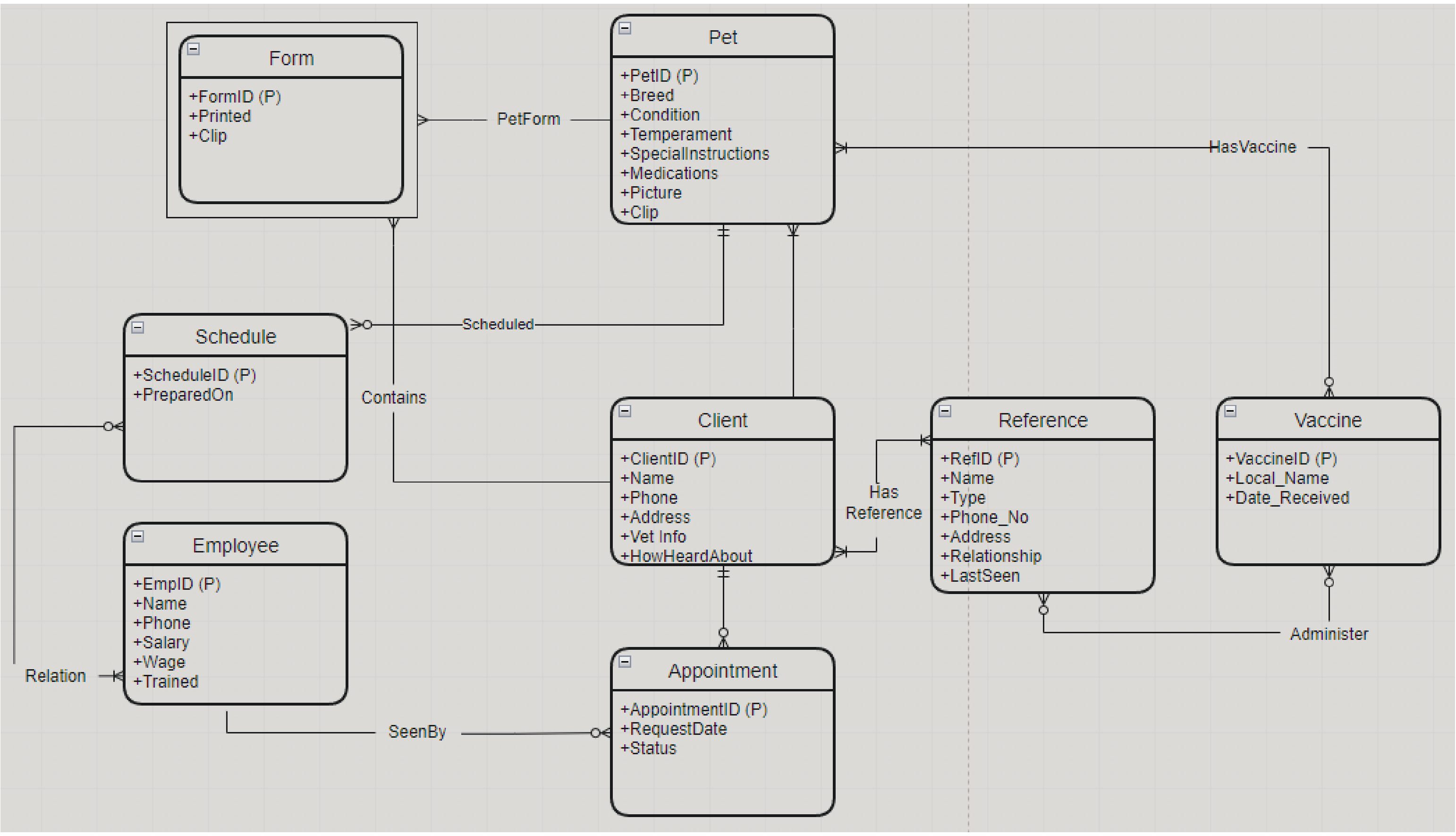


Level 1 DFD:



Logical Data Model

This data model demonstrates both the simplicity and the ability to extend both the mobile and web applications. For example, one could imagine that the system could be adapted to other use cases following the presented structural model, replacing pet with other service-items such as vehicles and vaccines with other materials related to the reference. This structure could potentially add



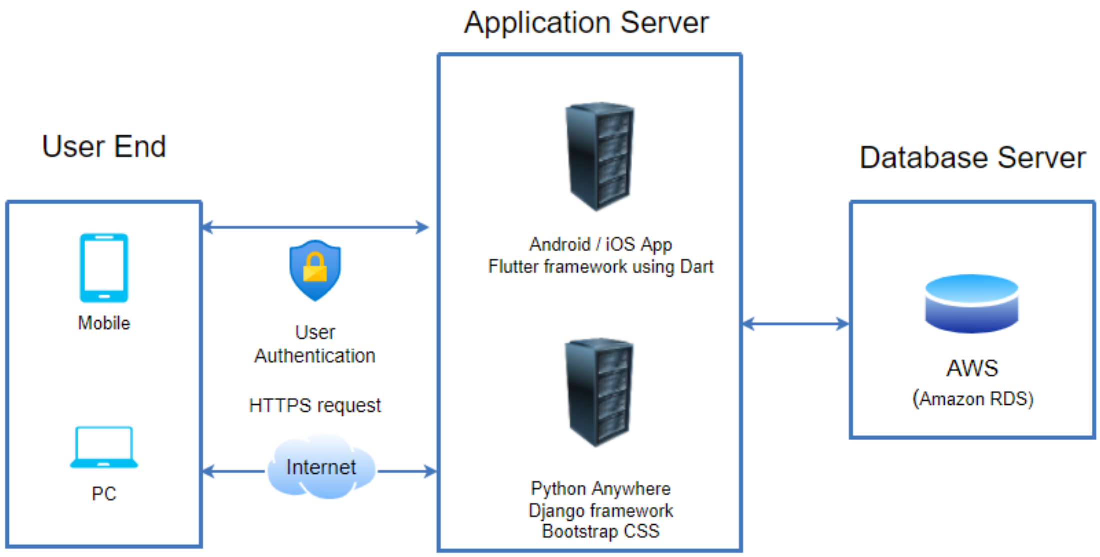
Design

The application to be created is a scheduling assistant for a pet grooming service. The main language running the backend of the web server is Python, and Django along with Visual Studio Code as the IDE will create the web application. Bootstrap will tie in with CSS to create a more professional presentation. To create a more workable environment, GitHub will allow for easier sharing as well as file rollback capabilities. The website will host a user creation process, and users will then enter their pets before they will schedule an appointment. All information will be stored in a MariaDB database using AWS's Amazon RDS functionality. Hosting the database in a web environment will enable the functionality for a user to view and update their account, pets, and other relevant information. New clients will be required to personally meet with the owner and the secretary in order to ensure that the salon has the ability and the pet has the temperament to be groomed. To schedule an appointment, the user will select a pet that they have already added to their account, and the web app will query the database for timeslots available for that type of pet and display them to the user. The user will select two timeslots that they would like for an appointment and submit the appointment request. Once that request is approved, the user will receive an email confirming the appointment with the most appropriate timeslot as chosen by the secretary and approved by the owner.

The web portion of the system will have a view for creating an account, a view for managing the account (editing details, adding pets, changing information), and a view for scheduling an appointment. If we were to really implement this, we would probably integrate it into the customer's existing website, so we would just use the customer's existing home page and whatever else they had on their site previously. Since we are not currently planning on an actual implementation, the homepage will be basic as a sample placeholder and provide functionality to navigate through the views.

The secretary will have the most access to the database itself. For example, the secretary will have the functionality to schedule custom appointments for customers. The secretary will have direct access to the client appointment requests, with the most appropriate requests being sent to the owner for approval. When the request is then sent to the owner, they have the option to approve or deny the scheduling request. If approved, the appointment time is then set in the database and appointments table with its date and time. The client will then be notified of an approved appointment. If the appointment is denied by the owner, the client will also receive an alert. The secretary can also pull up a full schedule for the next 30 days, 7 days, and daily schedules. The appointments marked "approved" in the Appointment table will be shown on the future schedules. The schedule will then be filled out with the form template table that we design. These schedules will be emailed to the secretary, whose email we have stored in the employee table. Accounts and clients can also be removed. With this request from the owner, the secretary can remove the account desired. All future appointments requests which have been denied from the owner will be marked as “denied” in the Appointment table. The client will then receive an email that their account had been terminated. All this information will be pulled from the Client table, Email table, and Form table. The secretary can also reinstate an account. The client will receive an email about the reinstatement of the account. This is only allowed if the account has been removed before.

3-Tier Architecture Design

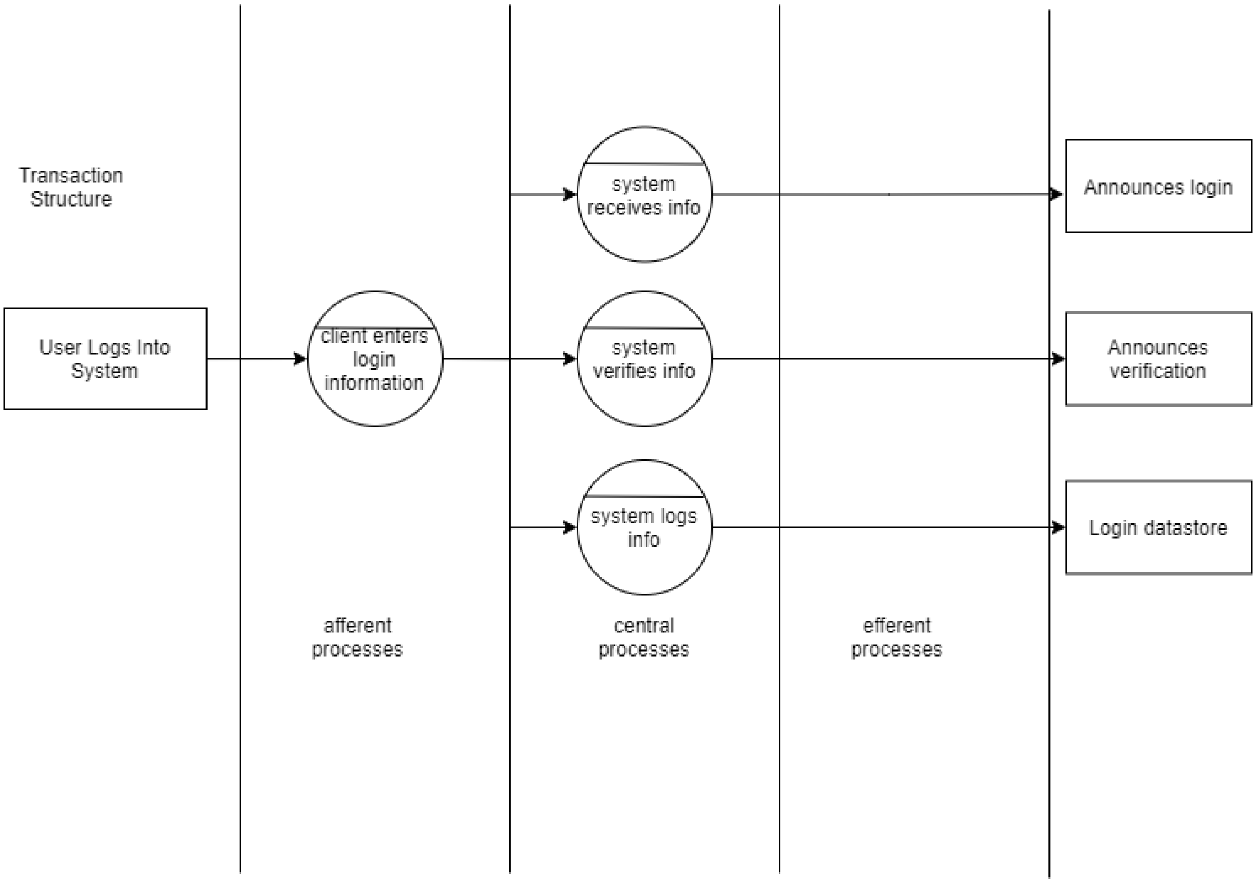


List of technologies:

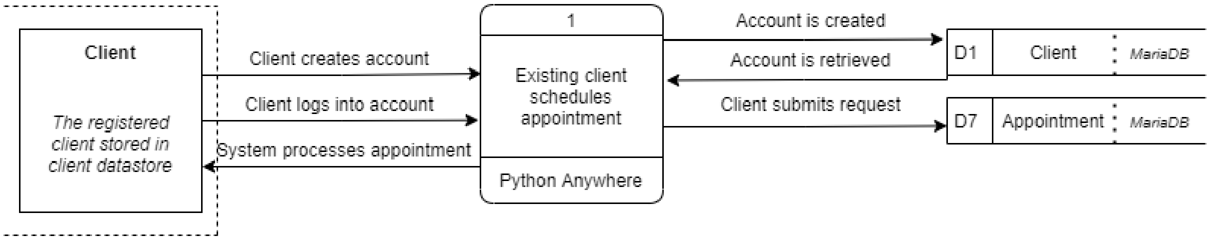
- Web server
  - Language: Python
  - Web framework: Django
  - CSS: Bootstrap
  - Python Anywhere
- Mobile Apps: Dart => Flutter
  - Android
  - Apple
- Security
  - NGINX, Cloudflare: Https requests
  - AUTH0: User Authentication

- IDE: Visual Studio Code
- Database: AWS ==> Amazon RDS
- Source Control: GitHub

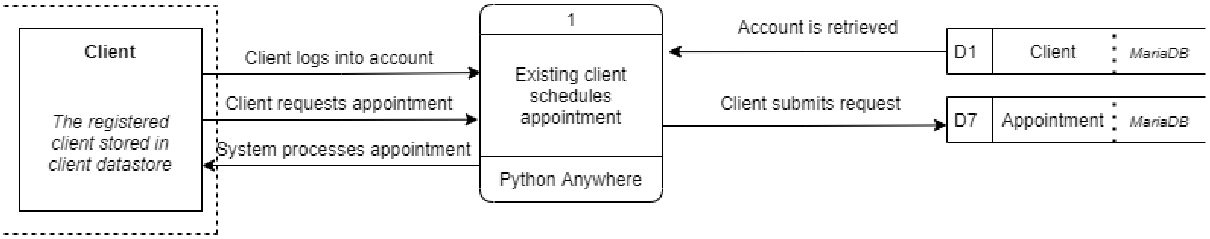
Physical process models



UC-1: EXISTING CLIENT SCHEDULES APPOINTMENT

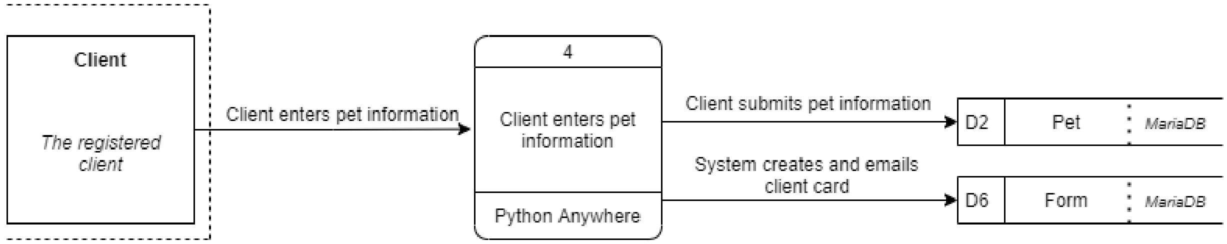


UC-2: NEW CLIENT SCHEDULES APPOINTMENT

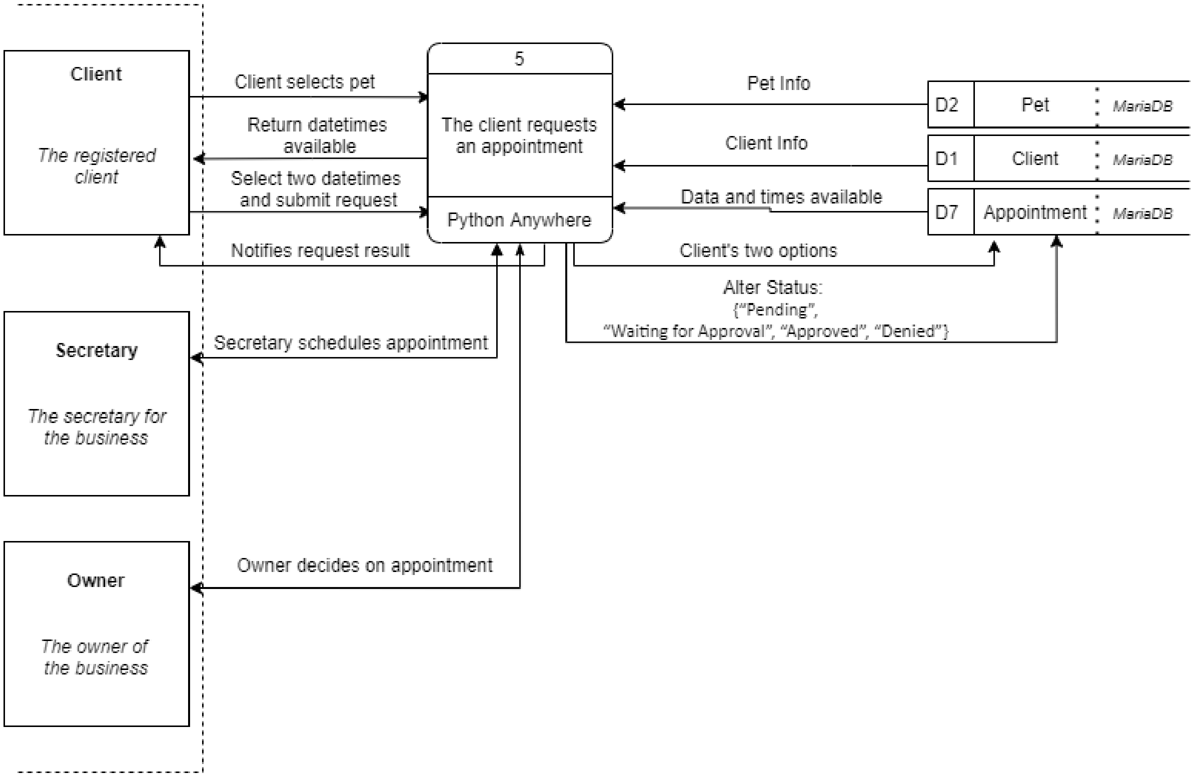

Client
The registered client stored in client datastore
Client logs into account
Client requests appointment
System processes appointment
1
Existing client schedules appointment
Python Anywhere
Account is retrieved
Client submits request
D1
Client
MariaDB
D7
Appointment
MariaDB

UC-4: CLIENT ENTERS PET INFORMATION

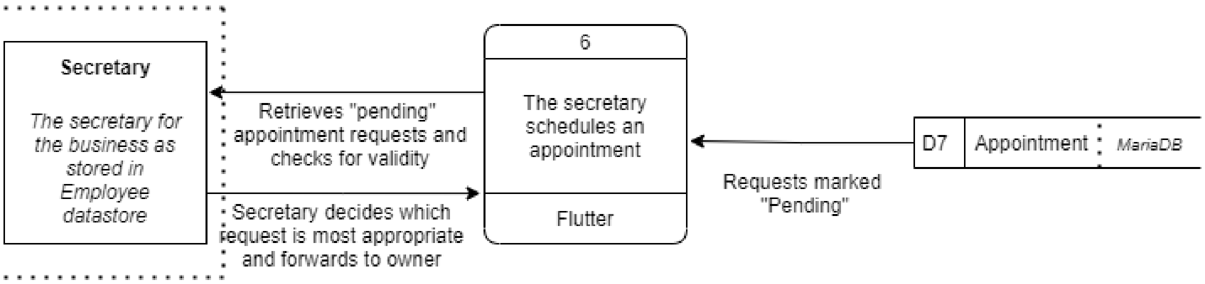
Client enters pet information


Client
The registered client
Client enters pet information
4
Client enters pet information
Python Anywhere
Client submits pet information
System creates and emails client card
D2
Pet
MariaDB
D6
Form
MariaDB

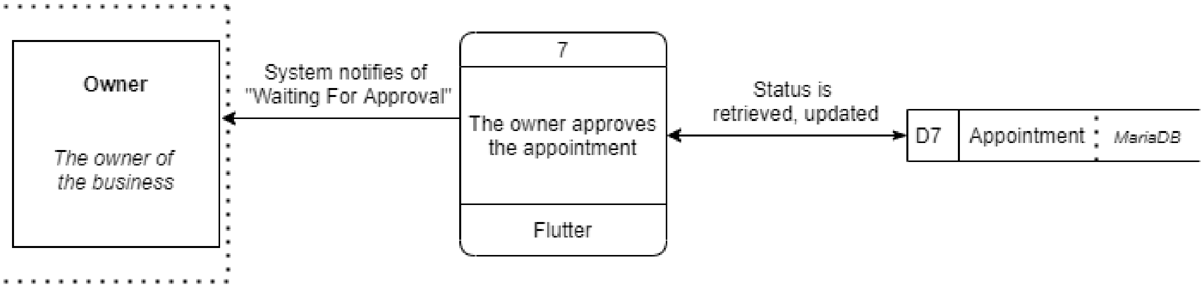
UC-5: CLIENT REQUESTS APPOINTMENT


Client
The registered client
Client selects pet
Return datetimes available
Select two datetimes and submit request
Notifies request result
5
The client requests an appointment
Python Anywhere
Pet Info
Client Info
Data and times available
Client's two options
Alter Status:
{"Pending",
"Waiting for Approval", "Approved", "Denied"}
D2
Pet
MariaDB
D1
Client
MariaDB
D7
Appointment
MariaDB
Secretary
The secretary for the business
Secretary schedules appointment
Owner
The owner of the business
Owner decides on appointment

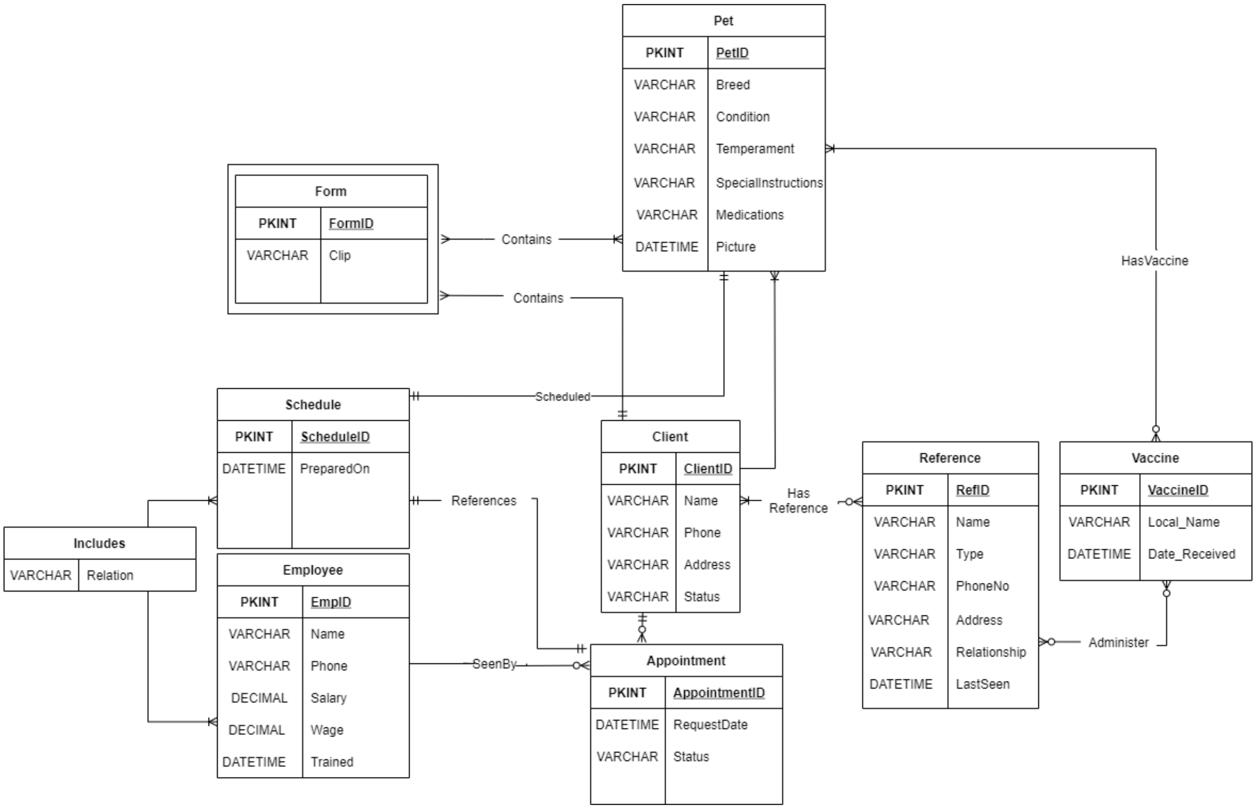
UC-6: SECRETARY SCHEDULES AN APPOINTMENT


Secretary
The secretary for the business as stored in Employee datastore
Retrieves "pending" appointment requests and checks for validity
Secretary decides which request is most appropriate and forwards to owner
6
The secretary schedules an appointment
Flutter
Requests marked "Pending"
D7
Appointment
MariaDB

UC-7: OWNER DECIDES ON THE APPOINTMENT

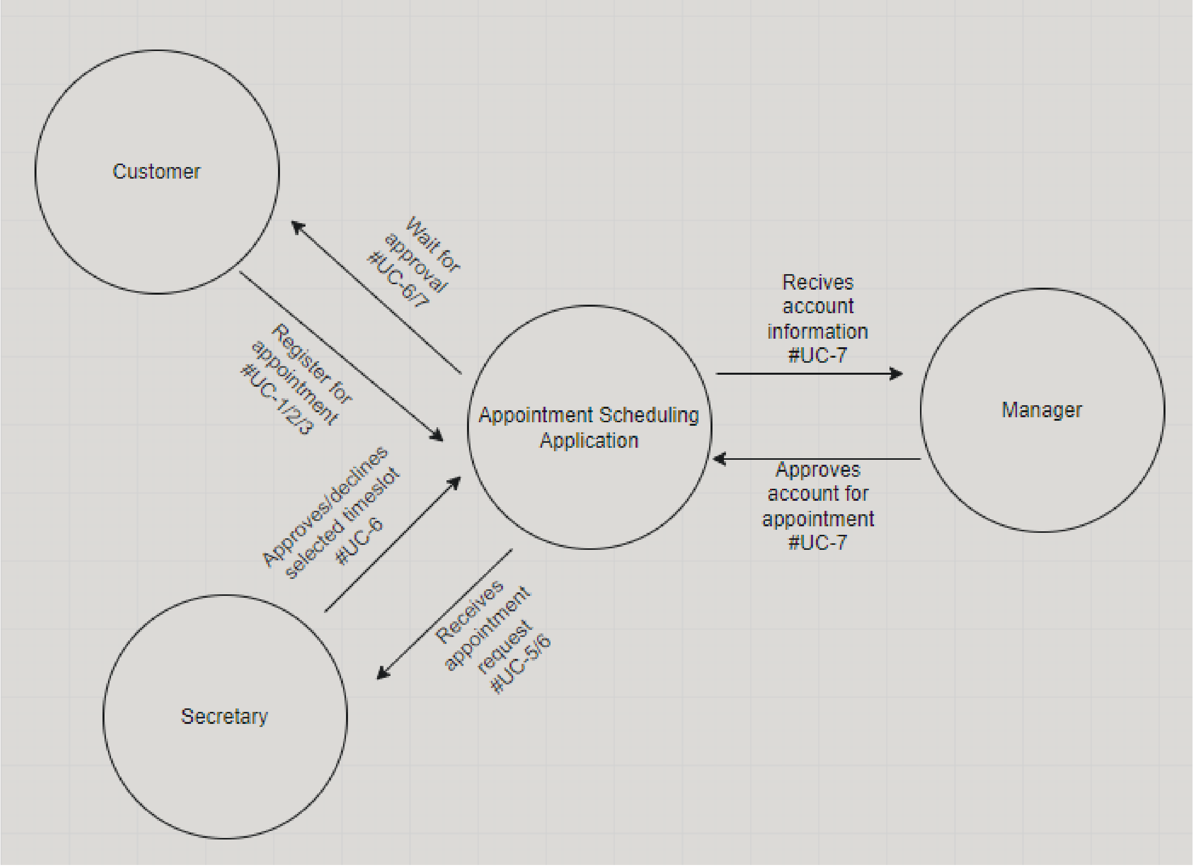

Owner
The owner of the business
System notifies of "Waiting For Approval"
7
The owner approves the appointment
Flutter
Status is retrieved, updated
D7
Appointment
MariaDB

Physical Data Models



-

Context Level DFD


Customer
Wait for approval #UC-6/7
Register for appointment #UC-1/2/3
Recives account information #UC-7
Appointment Scheduling Application
Manager
Approves account for appointment #UC-7
Approves/declines selected timeslot #UC-6
Receives appointment request #UC-5/6
Secretary

Data Dictionary

ATTRIBUTE	Data Type	Constraints	ATTRIBUTE	Data Type	Constraints
Pet			Form		
PetID	INT	PK, Autonumber	FormID	INT	PK, Autonumber
Breed	VARCHAR		Clip	VARCHAR	
Condition	VARCHAR		ClientID	INT	PK, FK, Autonumber
Temperament	VARCHAR		Includes		
SpecialInstructions	VARCHAR		ScheduleID	INT	PKFK, Autonumber
Medications	VARCHAR		EmpID	INT	PKFK, Autonumber
Picture	VARCHAR		Relation	VARCHAR	NOT NULL {“Prepared By”, “Groomer”}
ClientID	INT	FK, Autonumber	Employee		
Schedule			EmpID	INT	PK, Autonumber
ScheduleID	INT	PK, Autonumber	Name	VARCHAR	NOT NULL
PreparedOn	DATETIME	NOT NULL	Phone	VARCHAR	NOT NULL
EmployeeID	INT	FK, Autonumber	Salary	DECIMAL	
PetID	INT	FK, Autonumber	Wage	DECIMAL	
AppointmentID	INT	FK, Autonumber	Trained	DATETIME	
Appointment			Client		
AppointmentID	INT	PK, Autonumber	ClientID	INT	PK, Autonumber
RequestDate	DATETIME	NOT NULL	Name	VARCHAR	NOT NULL
Status	VARCHAR	NOT NULL {“Pending”, “Waiting for Approval”, “Approved”, “Denied”}	Phone	VARCHAR	NOT NULL
EmpID	INT	PK, FK, Autonumber	Address	VARCHAR	

ClientID	INT	PK, FK, Autonumber	Status	VARCHAR	NOT NULL Default “Current” {“Current”, “Closed”}
Reference			Vaccine		
RefID	INT	PK, Autonumber	VaccineID	INT	PK, Autonumber
Name	VARCHAR	NOT NULL	LocalName	VARCHAR	NOT NULL
Type	VARCHAR	NOT NULL {“Personal”, “Vet”, “Emergency”}	DateReceived	DATETIME	NOT NULL
PhoneNo	VARCHAR	NOT NULL	HasVaccine		
Address	VARCHAR		VaccineID	INT	PK, FK, Autonumber
Relationship	VARCHAR	NOT NULL	PetID	INT	PK, FK, AutoNumber
LastSeen	DATETIME		HasReference		
Contains			ClientID	INT	PK, FK, Autonumber
FormID	INT	PK, FK, Autonumber	RefID	INT	PK, FK, AutoNumber
PetID	INT	PK, FK, AutoNumber			
Administer					
VaccineID	INT	PK, FK, Autonumber			
RefID	INT	PK, FK, AutoNumber			

Implementation

GitHub Repo

1. https://github.com/Nauxiliary/Group2 (Links to an external site.)
2. Main

Testing

For testing, will be using all facets of system testing. This includes unit testing which will include black-box tests. Integration testing, which will include user interface and use scenario tests. System testing will include requirements tests, security tests, performance tests, and usability tests. Finally, acceptance testing will include an alpha test and beta test to ensure that the system meets All About Paw's business requirements. With all this meticulous testing, we will ensure our system runs efficiently, correctly, and as planned.

Documentation

As the project manager, Mason will write the documentation of the scheduling system as a complete **user guide** on how the system can be operated. A user guide is a guide or manual to assist the user in learning how to use our system. As the project is being built and steps are completed, Mason will work on the user guide, compared to waiting until the end of the project. This is because writing documentation and a user guide will take longer than one would think.

WORK ASSIGNMENTS

We have filled out a rough estimate schedule to see what we must complete, and how long it will take to complete it. It is also separated into specific sections. While we have this all laid out, we will try and follow this schedule the best to our ability. As the project leader, Mason will oversee assigning work and seeing the progress of the project to make sure everything is on schedule. We will be tracking our work assignments using openproj.

See the following pages for the full work breakdown structure.