Generalitat de Catalunya
Departament d'Ensenyament
Institut Provençana

PROJECT MODULE FILE (IT DEPARTMENT)

M12 Project

GROUP

Cicle formatiu de grau superior (CFGS) Desenvolupament d'Aplicacions Web - perfil professional Bioinformàtica (DAW-BIO) (cicle LOE)

TITLE

STR Matching for Dog Identification

GROUP MEMBERS (The students' names that will develop the project)

Alejandro Asensio Sanchez
Oscar Burgos Pacheco

DESCRIPTION (Write in no more than 15 text lines the objectives of the project)

This project aims to use bioinformatics tools to identify a dog by its STR's pattern from two different biological samples: saliva and droppings, respectively. To accomplish this, we are going to build a web application to handle the actions assigned to each user role.

This would be an example of a whole workflow:

1. Veterinarian registers a new Dog with chip number, owner's DNI and saliva sample into a Kit.
2. Bioinformatics Technician sequences the DNA from the saliva sample, getting the fasta file and the STR's pattern.
3. Street Agent takes a sample of a dog's droppings found in street, introduces the sample into the Kit, and registers the Incident into the app, storing the location, a photo as evidence and the Kit's barcode number.
4. Bioinformatics Technician sequences the DNA from the droppings sample, getting the fasta file and the STR's pattern, which will reveal if it matches with some existing saliva sample. This method allows us to uniquely identify a certain dog. If there is no match, a new "Suspected non-registered Dog" log is automatically stored in our database.
5. A Police Officer validates the evidences: photo + location + STR's matching patterns. If everything is considered as correct, a new InfractionProposal is generated and sent to Town Administration, which will proceed with the corresponding monetary fine to the dog's owner.

The friendly codename, as well as the domain name, of this project is "ocikitos", which in Spanish means little snout (animal nose).

The following sections are written in markdown (.md) nomenclature, so for a better readibility, we recommend to read them on GitHub:

https://github.com/aasensios/ocikitos

MATERIALS REQUIRED AND TECHNICAL SPECIFICATIONS OF THE PROJECT

FUNCTIONAL REQUIREMENTS

(Draft version: it is a first version to specify all functionalities and their results, as well as the profile users who may run each functionality)

Functional Requirements

This document has the purpose of defining the Functional Requirements (FR) of the project. In order to accomplish that, we are going to break them into User Stories (US), which will be implemented afterwards, in the code developing stage.

FR00. Define user roles

As the Product Owner, I want to define the basic user roles for the application, so that I can decide which tasks are performed by which role(s). The roles are:

- Application Administrator, **admin**, who has permission to manage the users, as well as all the actions inside the application; for instance, the massive load of Kits for samples (barcode as id).
- Veterinarian, **vet**, who is able to create or modify a Dog entry, injecting the chip into the dog's skin and taking a sample of dog's saliva into the Kit.
- Bioinformatics Technician, **bio**, who is able to create a Sample entry, which contains its DNA sequence info, such as the fasta and STR's files; as well as determine if two samples are from the same dog (match).
- Street Agent, **agent**, who is able to create an Incident entry, taking the dog droppings sample into the Kit and uploading a photo of them.
- Police Officer, **officer**, who is able to validate, invalidate or ask for more evidences about the droppings street Incident. By validating the Incident,
- Final User, **dog**, who is able to log in _from the dog perspective_, this is, and list the Infractions of its human owner, as well as modify some customizable fields, such as the alias, the password or the profile photo.

In this project we are implementing the CRUD (Create, Read, Update, Delete) system for every object type in our app, plus the Filter function; so we call it __CRUDF__. See [Use Case diagram](/01-delivery1/use-case-diagram.png).

FR00. Database and Backend Set Up

US000. Create Database

As an **admin**, I want to create the MySQL database structure and the needed tables, so that I can store the information of the app.

US001. Create Backend and Main App Navbar

As an **admin**, I want to create the backend project using the Laravel framework and the main navbar in blade syntax, so that I can start developing and placing the functionlaities on the app.

FR01. User Management

US011. List Users
As an **admin**, I want to list all the users, so that I can check their info and know the total amount of users in the app.
US012. Filter Users
As an **admin**, I want to filter the users list, so that I can search for an specific user.
US013. Add User
As an **admin**, I want to create a new user with an specific role, so that I can give the correct permissions to use the app.
US014. Modify User
As _any user role_, I want to modify the editable fields of my user, such as my password, so that I can maintain my account updated and secure.
US015. Delete User
As an **admin**, I want to delete a wrong user entry, so that I can maintain clean the users table in database.
US016. Log In
As _any user role_, I want to log in the app, so that I can use its functionalities, depending on my role.
US017. Log Out
As _any user role_, I want to log out from the app, so that I can leave the site securely.
US018. Register
As a _new user_, I want to register into the app, so that I can be assigned an specific role by the **admin** and start contributing to the app.

FR02. Dog Management
US021. List Dogs
As a **vet**, I want to list all the dogs, so that I can check their info and know the total amount of dogs in the app.
US022. Filter Dogs
As a **vet**, I want to filter the dogs list, so that I can search for an specific dog.
US023. Add Dog
As a **vet**, I want to create a new dog entry, so that I can register its new chip number, body features and the Kit's barcode; that Kit is made for taking a saliva sample from dog. This is the START of our workflow.
US024. Modify Dog
As **vet**, I want to modify a dog entry, so that I can update any change what is needed. For example, if a dog dies, we are not deleting it, but modifying its status field.
US025. Delete Dog
As a **vet**, I want to delete a wrong dog entry, so that I can maintain clean the dogs table in database.

FR03. Sample Management

US031. List Samples

As a **bio**, I want to list all the samples, so that I can check their info and know the total amount of samples in the app.

US032. Filter Samples

As a **bio**, I want to filter the samples list, so that I can search for an specific sample.

US033. Add Sample

As a **bio**, I want to create a new sample entry, so that I can register its sequenced DNA as a fasta file and the obtained STR's, which identifies the dog uniquely.

US034. Modify Sample

As a **bio**, I want to modify a sample entry, so that I can update any change what is needed. For example, if the sequentiation process has to be repeated, the fasta file must be overwritten.

US035. Delete Sample

As a **bio**, I want to delete a wrong sample entry, so that I can maintain clean the samples table in database.

US036. Analyze Sample

As a **bio**, I want to analyze a sample, so that I can __sequence__ it, obtain its __STR pattern__ and find a possible __STR match__ among our database. This user story is complex, so it has been split in three sub-tasks.

US0361. Sequence Sample

As a **bio**, I want to sequence a sample, so that I can store it as a FASTA file as an attribute of the sample register.

US0362. Obtain STR Pattern

As a **bio**, I want to obtain the STR pattern of a sample, so that I can identify a unique dog by that pattern of Short Tandem Repeats of DNA.

US0363. Find STR Match

As a **bio**, I want to find a possible STR match, so that I can uniquely identify a certain dog that was previously registered in our database. If the match occurs, an Infraction proposal is automatically emitted to the Police Officers; if no match is found, this anonymous sample is stored and our workflow reaches its END.

FR04. Incident Management

US041. List Incidents

As an **agent**, I want to list all the incidents, so that I can check their info and know the total amount of incidents in the app.

US042. Filter Incidents

As an **agent**, I want to filter the incidents list, so that I can search for an specific incident.

US043. Add Incident

As an **agent**, I want to create a new incident entry, so that I can register the location, an attatched photo of the evidence, along with the Kit barcode, where

the dog droppings sample is taken.

US044. Modify Incident

As an **agent**, I want to modify an incident entry, so that I can update the attatched photo or re-scan the Kit barcode.

US045. Delete Incident

As an **agent**, I want to delete a wrong incident entry, so that I can maintain clean the incidents table in database.

FR05. Infraction Management

US051. List Infractions

As an **officer**, I want to list all the infraction proposals, so that I can check their info and know the total amount of infraction proposals in the app.

US052. Filter Infractions

As an **officer**, I want to filter the infraction proposals list, so that I can search for an specific infraction proposal.

US053. Add Infraction

As an **officer**, I want to add an infraction proposal entry manually.

US054. Modify Infraction

As an **officer**, I want to modify an infraction proposal entry manually. so that I can fix any possible wrong information.

US055. Delete Infraction

As an **officer**, I want to delete a wrong infraction entry, so that I can maintain clean the infractions table in database.

US056. Validate Infraction

As an **officer**, I want to validate an Infraction proposal. In one hand, I may want to __approve__ an existing infraction proposal entry, so that I can confirm the evidences; in this case, the infraction status changes to "approved". In the other hand, I may want to __reject__ an existing infraction proposal entry, so that I can refuse the evidences as non-conclusive (maybe due to a non-accurate droppings photo or an inconsistency between owner address and Incident address); in this case, the infraction status changes to "rejected".

US057. Generate Official Document

As an **officer**, I want to emit an automated generation of an offical document, so I can proof my approval or rejection of an Infraction proposal. That document will be sent to the corresponding Town Administration. At this point, our workflow reaches its END.

FR06. Notification Management

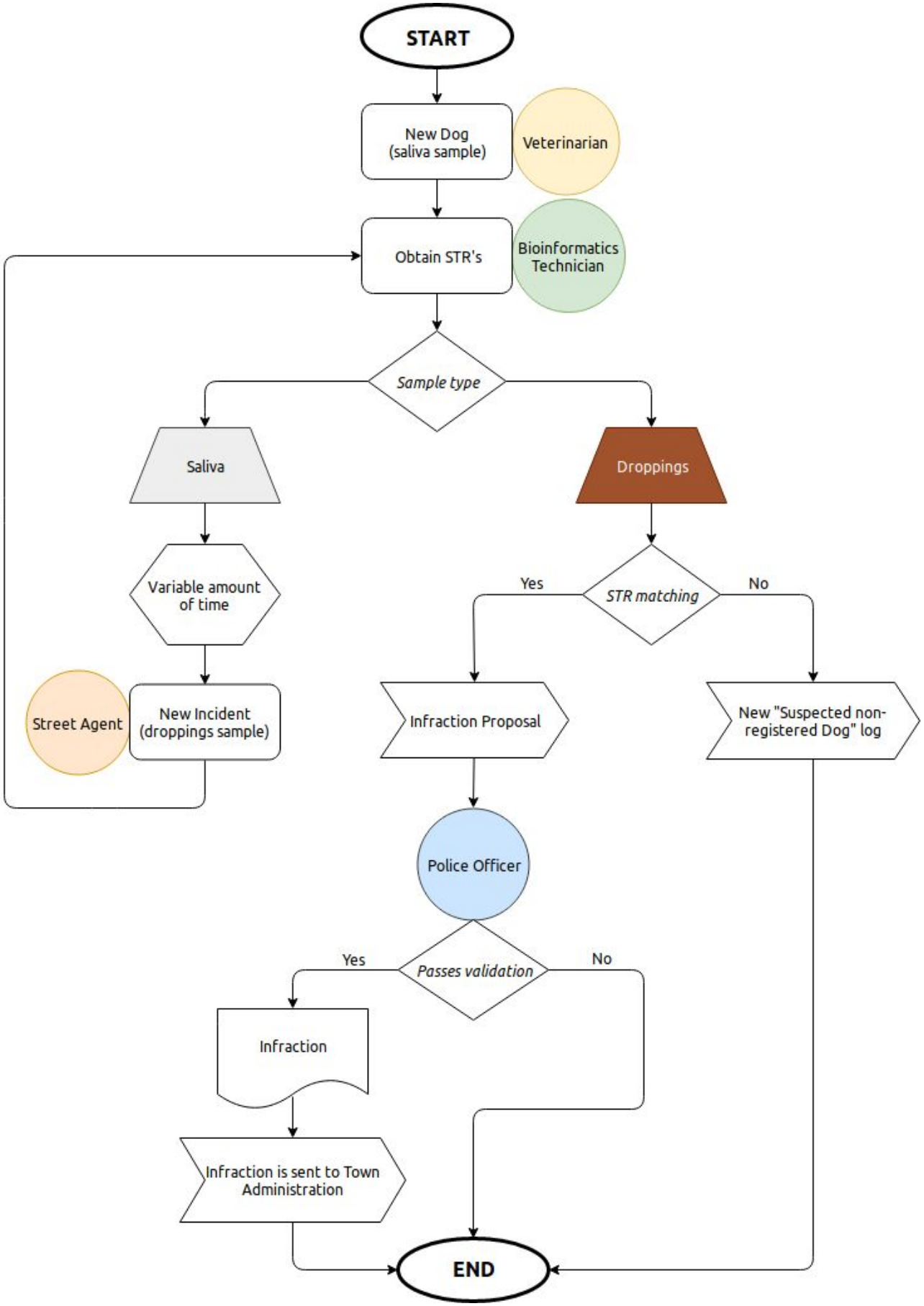
US061. Receive Notifications

As _any user role_, I want to receive a notification when a certain task is pending to be done by my user role, so that I can get my work done on time and when it's needed. For example, when a sample is sent to the lab, the **bio** role will receive a notification to analyze that sample. Other example is: when an infraction proposal is created and is pending to be validated, the **officer**

role will receive a notification to accept or reject that infraction proposal.

(NOTE: as a result of your initial interviews with the clients, you may use whatever you consider necessary in order to clarify what the client precisely wants: diagram, graphic, description, etc)

WORKFLOW DIAGRAM



NON-FUNCTIONAL REQUIREMENTS

Non Functional Requirements

This section has the purpose of defining the Non Functional Requirements (NFR) of the project. In other words, the infrastructure we need to build our application.

NFR01. Hardware

We need a computer for each one of us, with a minimum requirements: a decent processor like an Intel i5/i7 and 8GB RAM.

NFR02. Operating System

We need a Linux-based operating system, such as Ubuntu, with a set of software tools to implement the source code and test the application.

NFR03. Server

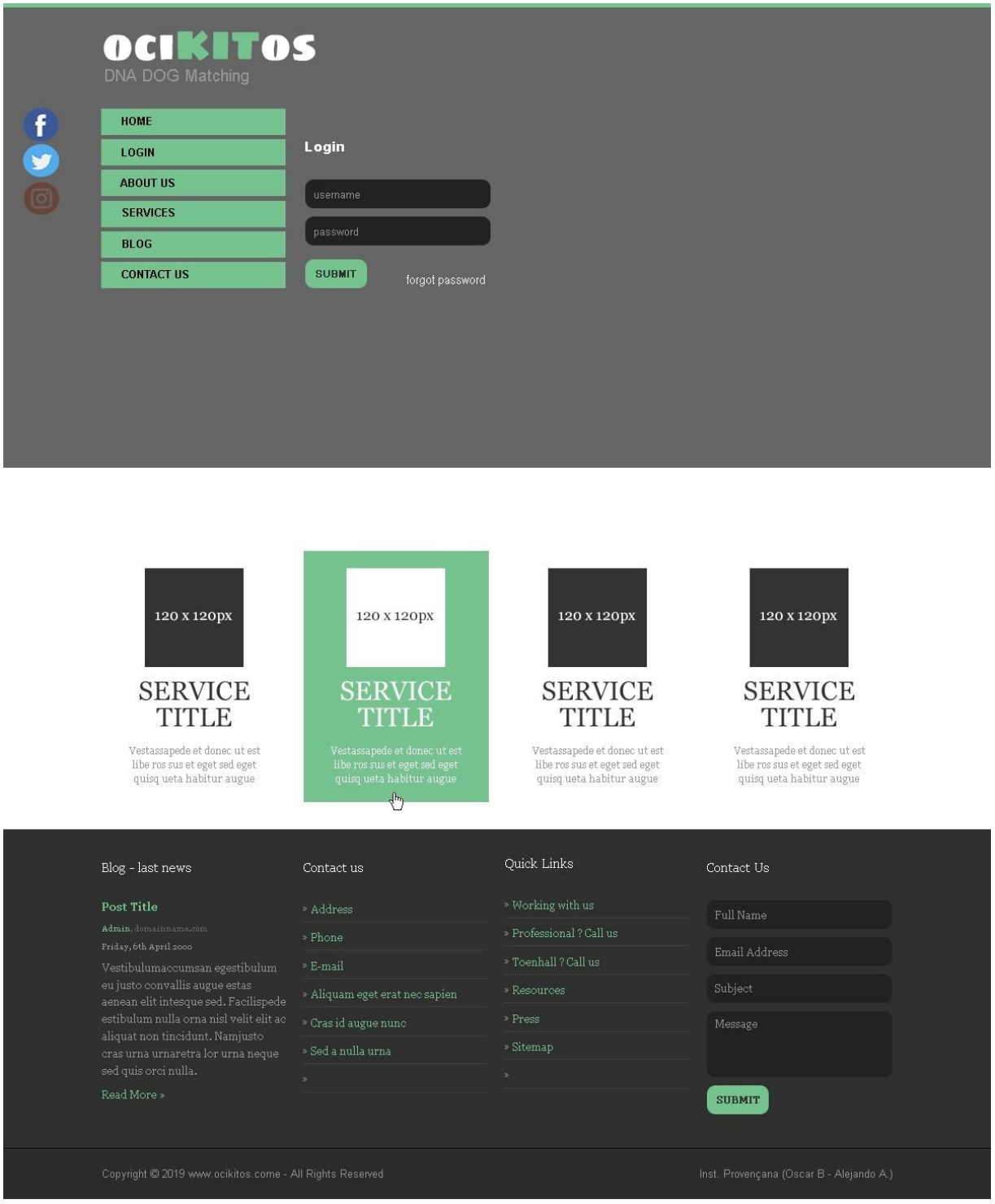
We need a Linux-based server, such as Ubuntu Server, with a set of software tools to deploy the application and perform tests in a real environment.

NFR04. Software

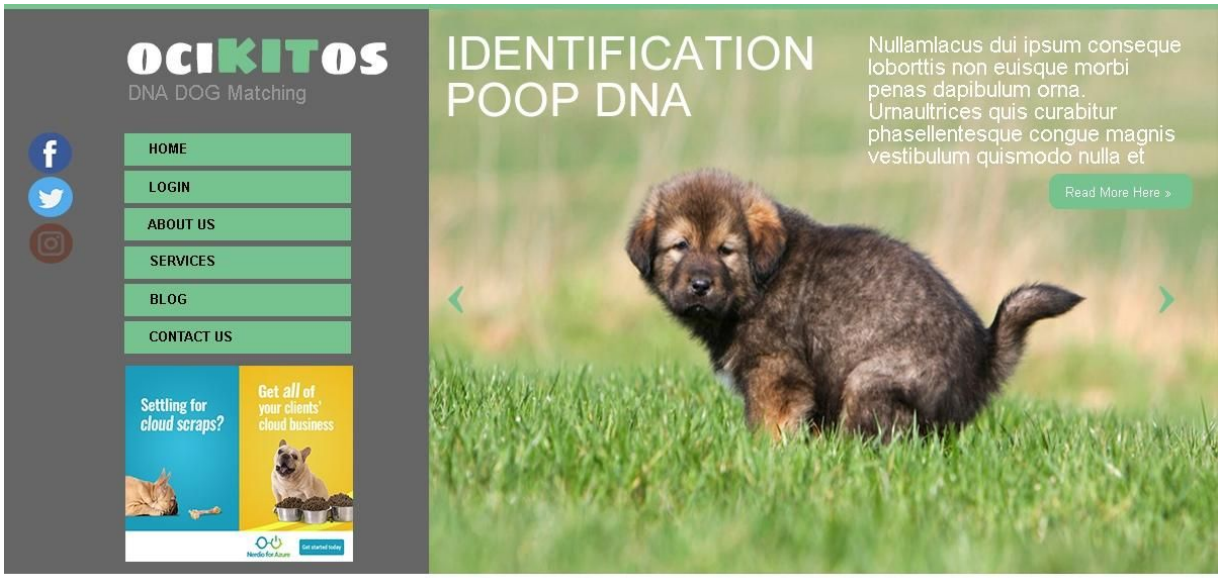
We need certain packages to write and test our code, such as programming language compilers and/or interpreters (PHP, Python), frameworks (Laravel), IDE (Netbeans, Visual Studio Code), web browsers (Firefox, Chromium) and version control system (Git, GitHub).

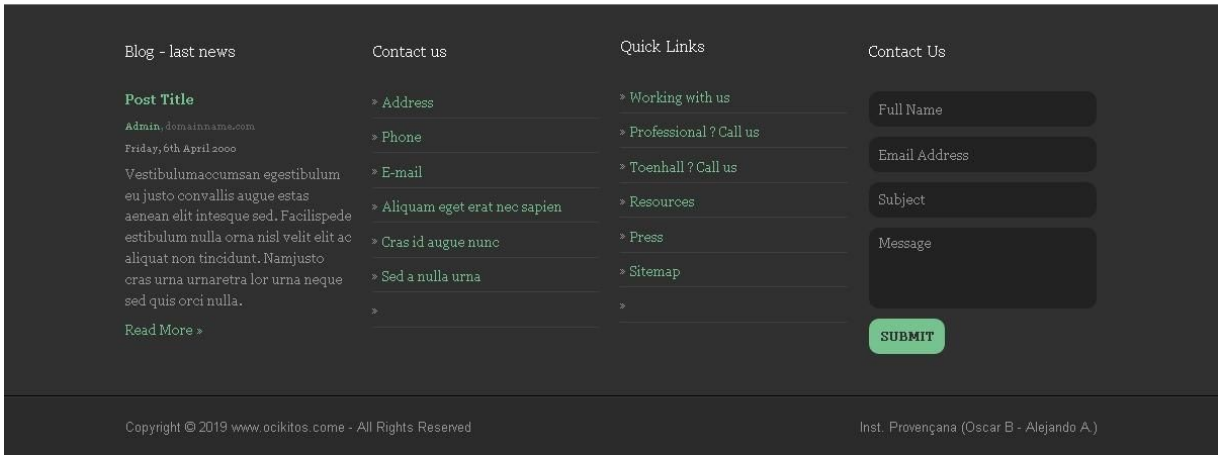
MOCKUPS OR WIREFRAMES

M011 [Login]

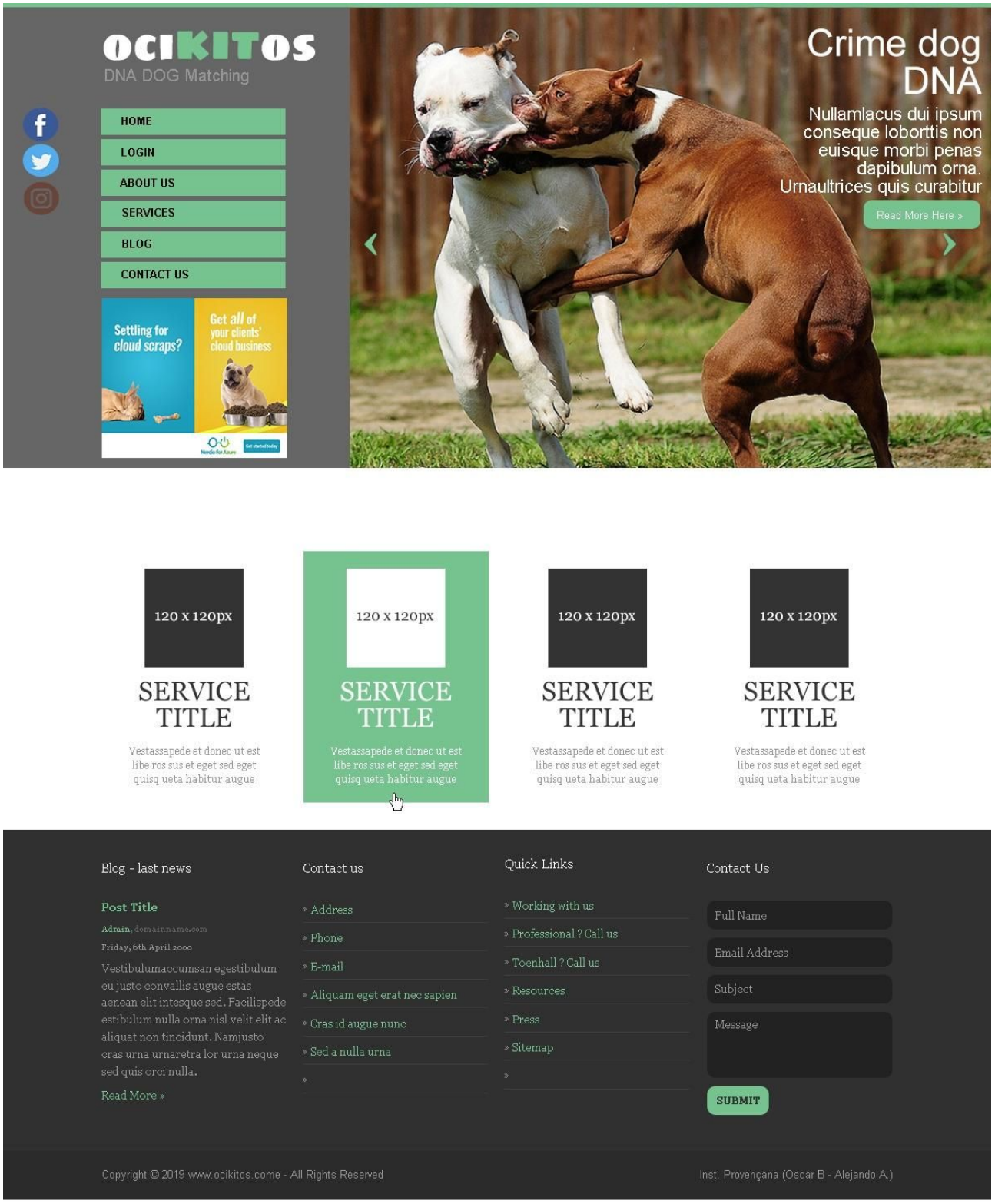


M021 [Poop Dna Service]

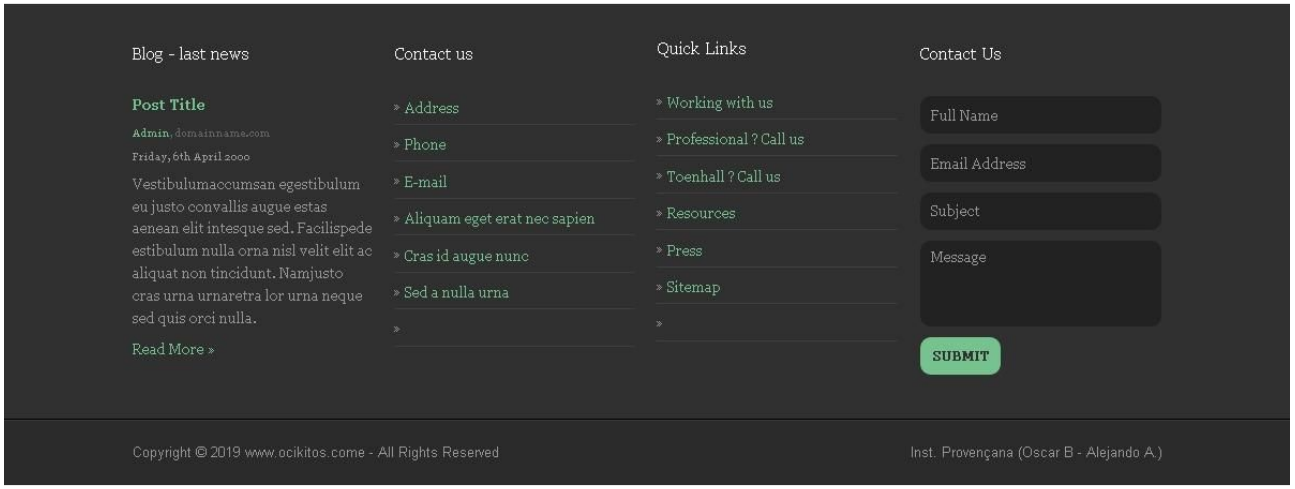




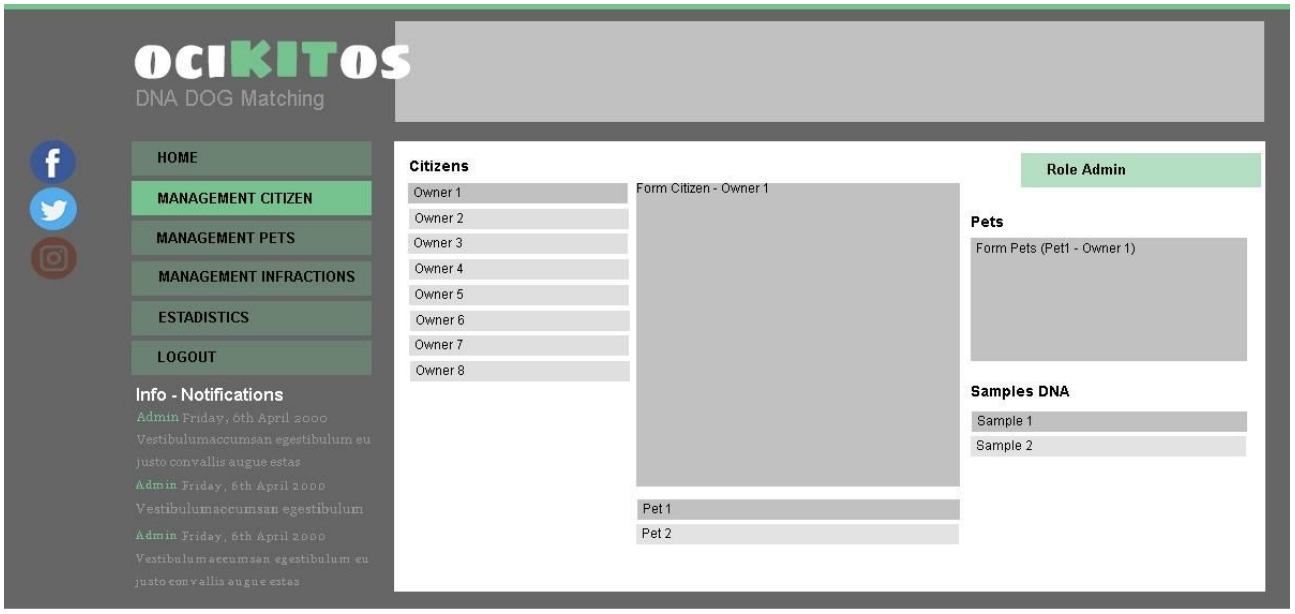
M022 [Crime Dog DNA Service]

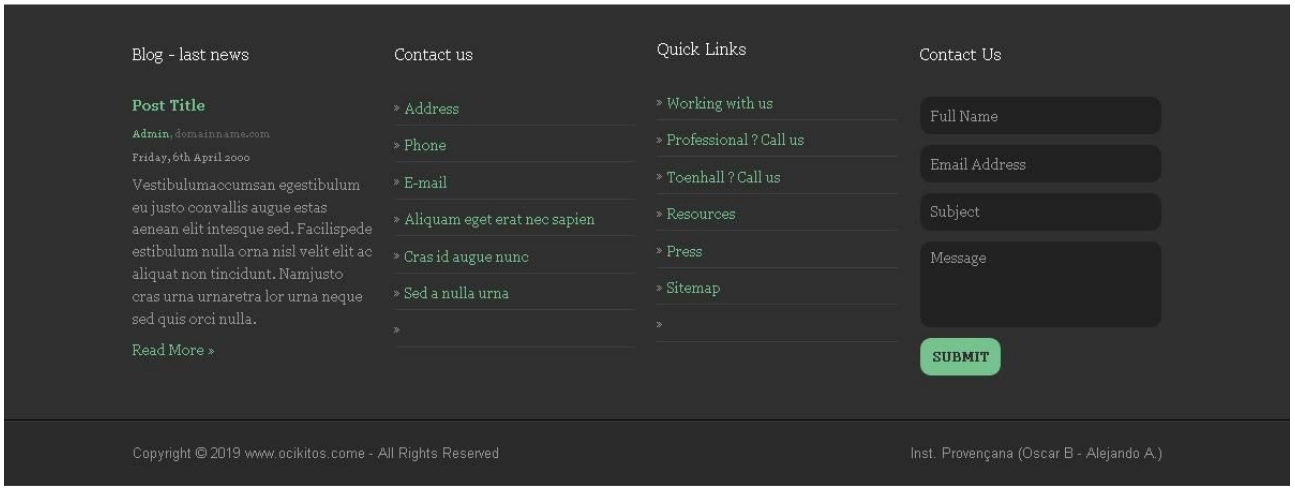


M023 [ID Dog DNA Service]

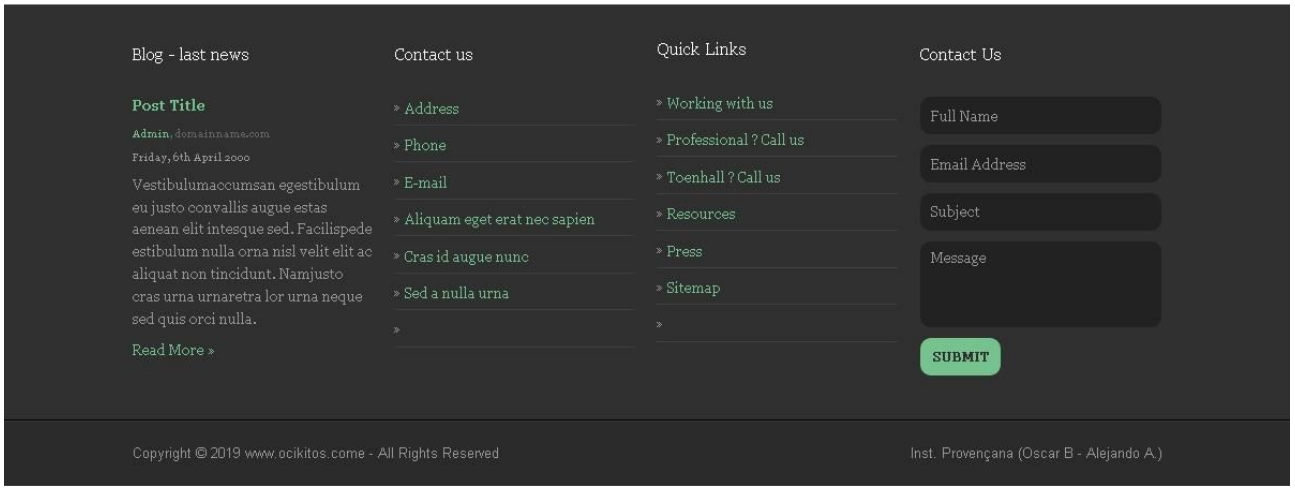


M031 [Citizens Management]

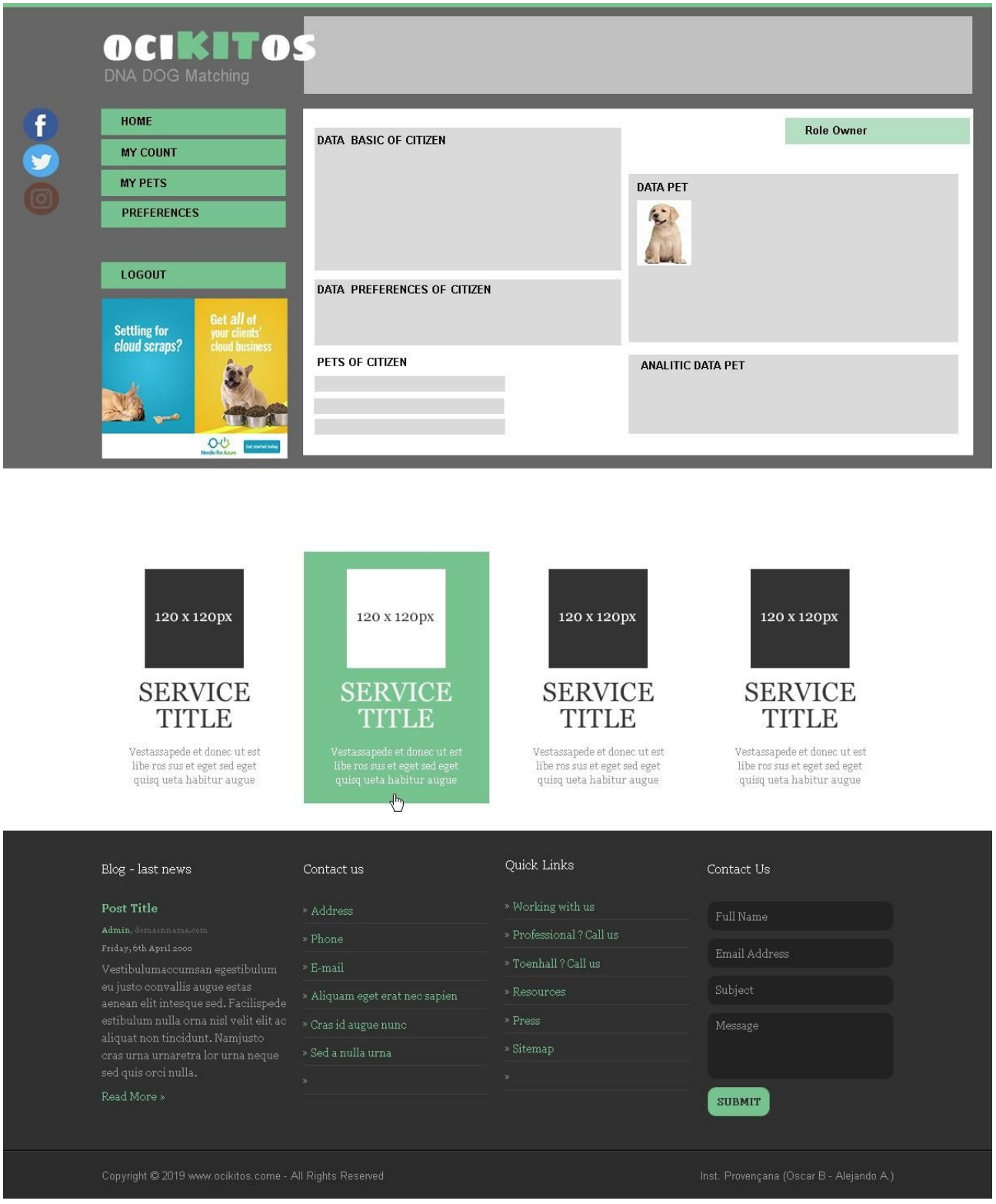




M032 [Dogs Management]



M041 [Dog's Owner Menu]



M051 [Street Agent Form]