**C868 – Software Capstone Project Summary**

**Task 2 – Section C**

|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | http://www.idevnews.com/views/images/uploads/general/wgu_logo.png  Christie’s Companions |
| **Student Name:** | Michael C Irick |

**Table of Contents**

*Create a professional looking Table of contents that includes your main and subheadings and the related page numbers. Use the automatic TOC generating function of Word or other word processing packages to make the process easier.*

*The headings that follow are only examples of what might be included. You will need to create headings that are appropriate for your application and process.*

***Remember that this needs to be a professionally formatted document with detailed information about your project that is easily accessible.***

Task 2 Part C – C868 Software Development Capstone

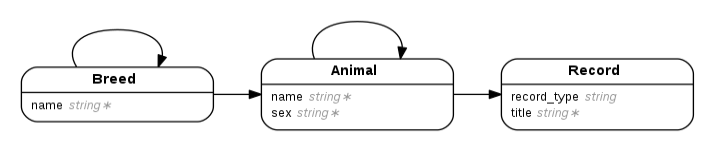
# Application Design and Testing

# Design Document

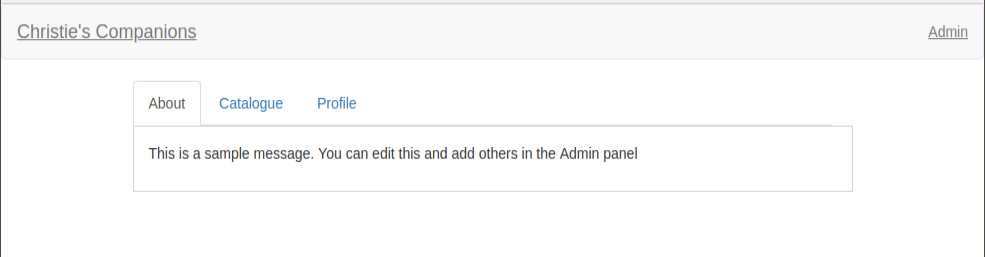
## Class Design

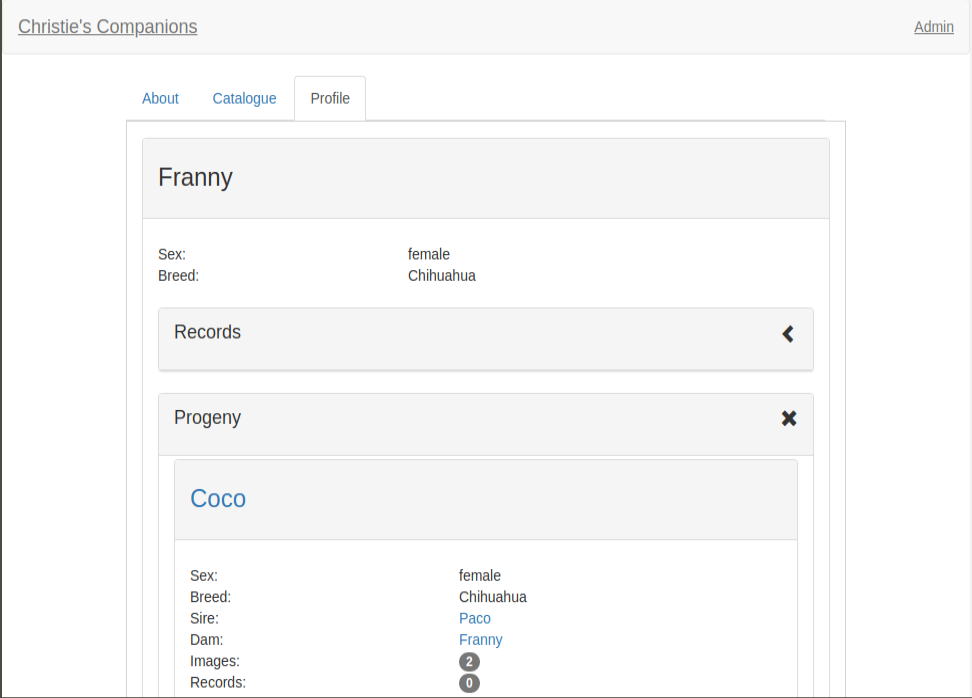
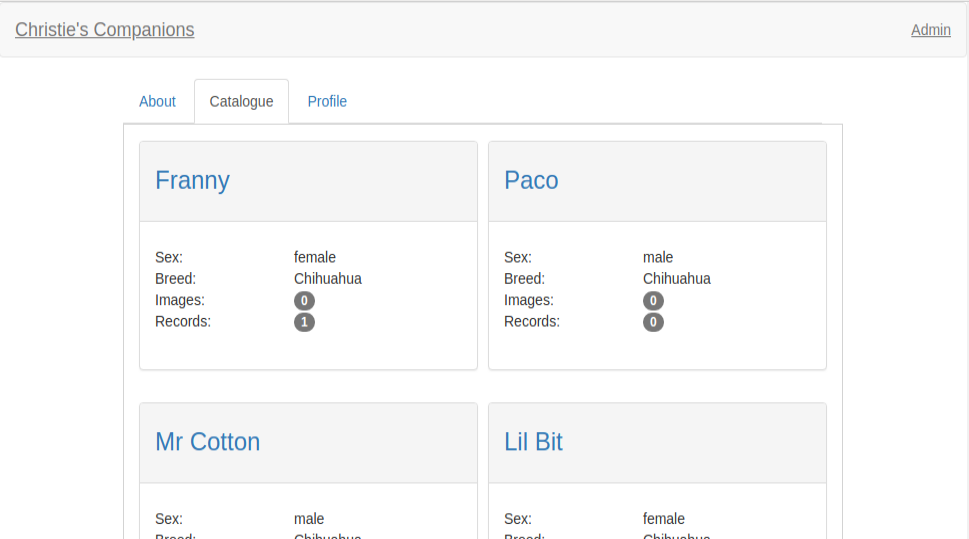
Below is the Entity-Relation Diagram that reflects the class design for the Christie’s Companions application. Due to the fairly standard implementation of MVC (Model/View/Controller) by the Ruby-on-Rails framework used in the project, only the classes involved in the core function of the application as a database will be shown and discussed.

The primary classes are Animal, Breed, and Record, while DashboardMessage and User are utilized in a secondary manner for presenting customized dashboard messages and restricting access to the administrator panel respectively. Each Breed has a one to many relationship with Animal representing all the animals belonging to that Breed, as well as a one-to-many relationship between Breed and itself representing an optional hierarchy of Breeds, i.e. Dog is the parent of Chihuahua. Animal has the inverse many-to-one relationship with Breed, two optional many-to-one relationships with Animal representing each Animal’s paternal and maternal relationships, and inverse one-to-many relationships with Animal for the Animal’s progeny, though only one of which will contain any relations. Record has a many-to-one relationship with Animal representing the various types of records, e.g. pedigree, vaccinations, that an Animal may have.



## UI Design

Below are the low-fidelity versions of the user interface for the catalogue; high-fidelity user interface is out of scope for this project. Provided are the Dashboard Message, the Catalogue, and the Profile.



# Unit Test Plan

## Introduction

### Purpose

Provide a brief description of the testing method(s) that you used and what the results it yielded. Also, what remediation was required if necessary and how it would be performed.

### Overview

Here you go into more detail about the test(s) and how it related to the overall project. You should include if a similar method was used in other parts of the application or why this was unique for a certain aspect of the code. Then, go into detail about what functions were tested, how the tests were conducted, and how errors were dealt with.

## Test Plan

### Items

What is required to complete the test(s)?

### Features

List the function/features that are part of each test.

### Deliverables

List what the test(s) would produce. For example, documentation or code notations.

### Tasks

List the tasks required to complete the testing and provide the outcomes you identified.

### Needs

Describe what was needed to be running or what support items had to be in place to perform the test? Specify versions if appropriate and other technical requirements. If a testing package and/or library was employed, be sure to identify it/them.

### Pass/Fail Criteria

Describe the criteria you used to determine the success of the test and what the protocol was for a positive result. Also describe what the recourse was if the test failed including remediation strategies and documentation requirements.

## Specifications

Provide sample code that represents what testing code was used. Screenshots are acceptable.



## Procedures

Provide a detailed list of the steps you used to complete the testing process. Be sure to mention if iterations were/are part of the process used and when pass/fail results were provided.

## Results

Here you will describe and provide examples of the testing results. If you were using a testing package include a screenshot of the interface. Screenshot work best.



# C4. Source Code

All source code for the application must be submitted with your documentation. Remember, this must be a fully functional application and the evaluator will need the code and other functional pieces to make the application launch and work as described. For web-based applications, a live website is advantageous and must be accessible to the evaluator, but you still must submit the complete code. Use a separate file for this section and identify the name.

# C5. Link to Live Version

Again, is a live version of the app can be viewed, provide the link or location where it may be found. Include a username and password if applicable.

# User Guide

*Note: This may be included as a separate document if you desire.*

## Introduction

Provide a description of the content you’re providing in the User Guide. This guide will include how to install, log into, sign up, and use all of the functions of the application. The steps need to be clearly defined and fully tested so the process works flawlessly for the evaluator.

## Installation and Using the Application

This procedural information should follow the basic rules of such technical references. While some procedures may provide for personal judgment yours should be clear and concise. Here are other rules to remember:

* Provide step-by-step sequences in the correct order.
* Follow the timing and sequencing of the actual operations.
* Provide visual stepping stones by using bullets or labeling steps.
* Strive to be concise. Avoid lengthy paragraphs but include enough detail so false assumptions are not made.
* Use common terms and jargon appropriate for the audience (someone with basic IT background).
* Explain why steps are completed or what they will yield as well as "How to" instructions.
* Test the instructions to ensure they match the actual product.
* Format the material for ease of reading and use graphic aids to clarify point/steps.
* Write in the present tense and the active voice.

## *Login and Signup (An example*)

1. *Click the "Log in" button in the top right corner of the app.*



1. *If you already have an account, log in with your account name and password. If you need an account, click on the link below that states “Need an account?”*
2. *If you need to create an account, choose a unique username and password. By default, the password requires at least 6 characters. This function could be changed to address new password requirements.*

## *Classes*

### *Create a New Class*

1. *Once logged in, click on the link at the top labeled “Classes”. This will enable you to create a new class of students.*



1. *Click on “+ Add Class”.*



1. *Enter a class name and its description. The class name must be unique.*
2. *Click “Add Class” to add the class, otherwise click “Cancel” or outside of the modal to cancel adding the class.*



## *Reports*

1. *To access the reporting feature, from the Schedule module, click on “Generate Report” near the top right of the page.*



1. *By default, all events are generated and displayed.*