Programming III – Fall 2022

Course Project: Topic

Team Information

|  |  |  |
| --- | --- | --- |
|  | Name | Student Id |
| Member A | Hana Louiza Moussaoui | 2275881 |
| Member B | Taryn Beaupre | 2173710 |

Project Description

Our application allows the user to browse a virtual pet adoption center and to adopt one or multiple pets. The user can refresh the pet display page to see randomly selected pets from the pet database and click on a pet to see details such as age, type, adoption status, and description of the pet. They can then proceed and adopt the pet if they wish, by filling out an adoption form.

Development Approach

1. **Understanding the problem.**

Develop a pet adoption/rescue center that will be able to show pets available in the center and let the user adopt one or many pets.

1. **Formulating the problem.**

A pet database will keep track of which pet has been adopted or not. In the main window, the pets need to be randomly displayed to the user. The user should be able to refresh the page and there should not be duplicate pets on the window. The user should also be able to select a pet to view the pet’s details.

The user can decide to adopt a pet and a new window will appear where they can fill out information. The user can only adopt an available pet and their form details must be valid (details to be decided aka which home type is required, minimum age needed, etc) for the adoption to be completed. Their information will be saved to a file in a folder.

1. **Developing the application \ algorithm.**

Coding in C# for the backend via Visual Studio 2022 as well as WPF for the front end.

1. Create a pet class which holds information about a pet such as name, type, age, adoption status etc.
2. Create a pet database which will hold a list of the pets in the adoption center.
3. Create a main window that will allow the user to see pets from the database 4 at a time.
4. Create a randomizer so 4 random pets from the database are chosen and shown on the main window.
5. Create a pet detail page that will be shown when the user clicks on a pet to show extra information about the pet.
6. Create an adopt button on the pet detail page that will allow the user to navigate to the adoption form.
7. Create an adoption form that will ask for information such as the user’s name, address, date of birth, etc.
8. Update the pet’s status to “adopted”.
9. **Implementing the application \algorithm.**

All code was implemented locally and runs locally.

1. **Testing.**

More rigorous testing to follow, testing by examining cases and trying different actions the user can do.

OOP Design

Talk about the classes you need to create for the application and what is the purpose of each class. Include the UML class diagram in this section. The UML class diagram should include the relations between the created classes. Do not mention the WPF classes (Window, etc.)

**Pet class:** This class contains the backing fields and

**Pet Database class**: This class is to stores Pet objects in an array and acts as a storage for the pets and their information (which is then passed to the Pet class to be utilized).

Adopter class: class contains members needed to validate the adopter’s information and will

**Adopter class:**

**Adopter database class:**

Contributions

**What did each team member do?**

|  |  |
| --- | --- |
| **Member** | **Work** |
| **Taryn** | * WPF design for all pages * adoption form and class * saving adoptee info to a file * updating pet status * submission popup * code improvements across multiple classes * summaries, code organizing, comments * team document * report of all adopted pets in the database |
| **Hana** | * WPF base code design for the pet windows * Pet class * refreshing and saving status of pets * pet details window * pet database class * adopter class * reading from file and showing adopter’s and their adopted pets |

**How was the work in the project divided?**

It was decided early on that the work would be split in half (which was decided as the pet portion and the adoption portion) and the WPF was also split evenly. All pet icons are hand drawnas well.

App Snapshots

**The main window \*\*UPDATE**

Appears on initial launch and is the first window displayed to the user. User can choose to access the pet center when they click on the “Click to see available pets” button

A screenshot of a computer

Description automatically generated

**pet page**

This page appears after the user clicks the first button. It shows the user 4 randomized pets. The user can then select a pet and it will open the Pet Details window. They can also choose to refresh the page to get another 4 random pets.

A screenshot of a computer

Description automatically generated

**pet details page (NO GUI YET)**

This page appears after the user selects a pet from the available pets. A screenshot of a computer

Description automatically generated

**Adoption form:**

A screenshot of a computer

Description automatically generated

**Submission popups:**

A screenshot of a computer

Description automatically generated

Future Work

* Allow the user to put up a rescued pet and register the pet to be available for adoption
* Bonded pets/dual adoption (where pairs of pets have bonded and would prefer to be adopted together)
* Mini pet shop where the user can select some items to buy alongside their adoption/make a donation
* Adoption fees + receipts. Having a total of all the money raised by the adoption center.

Appendix A: Team Contract

Submitted team contract goes here.

Appendix B: UML Class Diagram UPDATE ME

A diagram of a pet and dog

Description automatically generated

* DO NOT PLACE A  LINK TO THE DIAGRAM.
* Do not include WPF created classes in the class diagram.
* The diagram should be placed in the document.