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.**

We want to 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.**

We want a pet database that 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.**

We will be 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).

Contributions

**What did each team member do?**

Taryn: UI design for all pages minus the pet details, adoption form, saving adoptee info to a file, updating pet status, submission popup

Hana: UI, Pet classes, refreshing and saving status of pets, pet details, pet database  
**How was the work in the project divided?**

We decided early on that we would each do one part of the project (which we split as the pet portion and the adoption portion) and that we would split the WPF, and that is what we did.

App Snapshots

The main window:

A screenshot of a computer

Description automatically generated

pet page:

A screenshot of a computer

Description automatically generated

pet details page: (NO GUI YET)

A screenshot of a computer

Description automatically generated

Adoption form: A screenshot of a computer

Description automatically generated

Submission popup:

A screenshot of a computer

Description automatically generated

Future Work

**\*\*Important, this is some work we have not finished yet\*\*** Implement adoption fees related to the type of pet being adopted. We want to utilize an interface called IAdoptable to hold the adoption fees and the status of the pet + have classes for each Pet type that will inherit from the Pet class. The pet class will contain members to be overridden such as where they live, what they eat, etc. Finishing the validation of the adoption forms.

Additional work: We could allow the user to put a pet up for adoption, dual adoption (where pairs of pets have bonded and would prefer to be adopted together), maybe add a mini pet shop where the user can select some items to buy alongside their adoption. Maybe having an adoptee of the month (the person with the most adopted pets) or a total of all the money given to the adoption center / num of pets adopted.

Appendix A: Team Contract

Submitted team contract goes here.

Appendix B: UML Class Diagram

* 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.