Pet Adoption Database (PAD)

Software Design Document

Prepared by:
Simay Cural, Na'ama Nevo,
James Settles, Walt Jones

Release Date:
September 20, 2022

Executive Summary

Our Pet Rescue program strives to streamline the process of matching adopters and fosterers to pets who need a home. Currently, most pet rescues and shelters manually match pets to potential owners, which can get complicated and time consuming. Our program should accurately and efficiently place pets with people who are looking to either adopt or foster. It takes into account characteristics of the pet such as: Does the pet get along with other pets? Does the pet get along with children? Is it potty trained? Pre-existing health concerns? Additionally, information about the owners and their unique families will be considered too such as allergies, children, and experience. Lastly, this program should provide helpful statistics like how many foster parents end up adopting.

Document Versioning

Date	Owner	Comment
9/13/2022	Walt Jones	Created document outline, updated feature matrix and discussion, Detail Design descriptions
9/13/2022	Na’ama Nevo	Edited summary, description
9/14/2022	Walt Jones	Updated UML, detailed descriptions, feature matrix
9/14/2022	Simay Cural	Edited feature descriptions, data format
9/14/2022	Na’ama Nevo	Edited descriptions in features and components
9/14/2022	James Settles	Updated UML
9/19/2022	Walt Jones	Updated UML
9/20/2022	Na’ama	Updated UML and descriptions
9/20/2022	Walt	Updated UML and Detail Design

Project Description

Hope We Don’t Get A C++ inc. is looking to enter the pet market space by creating a new pet rescue program. It will be a multiplatform program. Additionally, it is a standalone application that does not rely on the internet.

Pet Rescue uses a graphical user interface (GUI) to run the program. This will support a multitude of features. For example, it should streamline the process of adopting and fostering. It takes into account characteristics of the pet such as: Does the pet get along with other pets? Does the pet get along with children? Is it potty trained? Pre-existing health concerns? Additionally, information about the owners and their unique families will be considered. This includes factors such as the presence of allergies, children, and former pet experience. Lastly, this program should provide helpful statistics (like how many foster parents end up adopting) to help expand the agency’s brand.

Features

The feature matrix enumerates the technical features required to support each of the business requirements. The discussion section provides details regarding the constraints and functionality of the feature. The ids are used for traceability. Features that can be removed should strike-through the feature id and have a comment added to identify why this feature can be removed without impacting the BRD requested functionality..

Priority Codes:

H - High, a must have feature for the product to be viable and must be present for launch
M - Medium, a strongly desirable feature but product could launch without
L - Low, a feature that could be dropped if needed

Feature Matrix

ID	Pri	Feature Name	Comment	BRD ID
P.1	H	Language	Code written in Java	s.1
P.2	H	Bundling	Bundling done in JAR file	s.2
L.1	H	Main Menu GUI		ux.2
~~L.2~~	~~L~~	~~Search Pet/Family~~		~~e.5,~~
L.3	M	Create new pet or family profile		e.4
L.4	H	List of Profiles GUI		ux.2, e.6
L.5	H	Statistic GUI		ux.2, e.3
~~T.1~~	~~M~~	~~Error Handling~~		~~ux.2, ux.3~~
C.1	H	Matcher		e.6
C.2	H	Profile		ux.2, e.1, e.2, e.4
C.3	H	DB Adapter		ux.1
C.4	H	Relational Database		ux.1
C.5	H	Statistics		e.3

ID	Pri	Feature Name	Comment	BRD ID
~~C.6~~	~~L~~	~~Pet info generated automatically~~		~~e.4~~

Feature Discussion

P.1 - Language

PAD must be multi-platform and run as a standalone application. Several programming languages could potentially be used; Python, JavaScript, JAVA, etc. While Python is cross-platform, there are challenges in bundling the code and getting consistent GUI behavior. JavaScript is widely available on many platforms but suffers from similar bundling issues and typically runs in a web browser (which is a problem for the target embedded market). JAVA has cross-platform support, includes capabilities for making executable bundles, has consistent GUI behavior across platforms.

P.2 - Bundling

JAVA's built in JAR technology enables bundling of executable code with other assets. For many operating systems, JARs are executable by double clicking on the JAR file. Distributing software as JAR files also has the advantage of adding a step for those trying to improperly modify or reverse-engineer assets.

L.1 - Main Menu GUI

Using the Swing UI toolkit, the main menu GUI should have options to search for profiles, create new profiles, or generate statistics.

~~L.2 - Search for Pets & Families~~

~~Search using the DBAdapter querying our relational database for different attributes of pets or families. Typing in a keyword will search for families or pets that have traits containing or relating to the word and present a filtered list of profiles.~~

L.3 - Create new Pet & Family profile

The Create New Profile button on the main menu GUI allows the user to create new profiles for pets and families. It opens an editable blank profile template in a new window. These profiles are then added into the database with the DBAdapter.

L.4 - List of Profiles GUI

Using the Swing UI toolkit, this GUI displays a scrollable list of profiles excerpts. Clicking on a profile excerpt will open the corresponding Profile GUI. The user will be

able to choose to see a full list of pets, a full list of families, or a filtered list based on a keyword search.

L.5 - Statistic GUI

Uses Java Swing to display statistics about the number of adopted and fostered pets, and the number of adopting and fostering families. This is done querying the database using the DBAdapter and the statistics logic.

~~T.1 - Error Handling~~

~~Using the Swing UI toolkit, GUI displays error messages when user inputs incompatible data types, searches don't return results, the user navigates the program without inputting data, etc. The program has unique exceptions for errors not covered by existing Java errors.~~

C.1 - Matcher

Every pet or family profile has a match button. This button opens a new window using the matcher with a list of pets or families from the database that have compatible traits with the current profile. For example, the pet allergen traits would be matched with family allergy traits.

C.2 - Profile

This class should contain all the relevant information about either families or pets. Each profile class represents some data that is either read from or will be written to the database using the DBAdaptor.

C.3 - DB Adaptor

The DB Adaptor will connect to the database and translate from Java method calls to SQL queries.

C.4 - Relational Database

Database that stores pets and families with unique keys for each. Pets will store: ID, name, breed, size (weight and height), age, previous foster families, training and certification statuses, energy levels, and whether or not they are potty trained, kid friendly, and get along with other animals,
Families will store: ID, name, number of kids, allergies, animal and breed preferences, energy level preferences, past/current fosters, and past/current adoptions.

C.5 - Statistics

Statistics about adoptions will be generated by using the DB Adaptor to query the database for certain stored metrics. The metrics shown will be the number of pets that have been fostered and adopted, and the number of families who have fostered

or adopted. These metrics will have to be updated when each of their triggering conditions have been met. For example, matching a pet and family will update statistics tracking the number of matches.

~~C.6 - Automatically Generated Pet Information~~

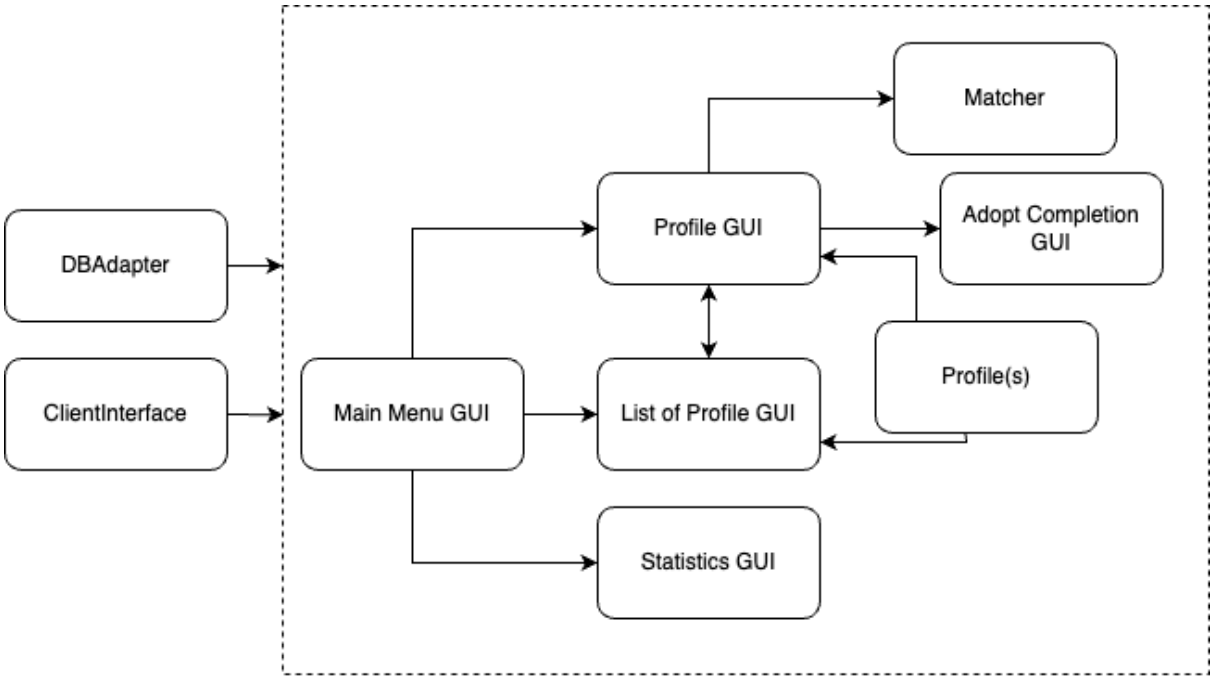
~~This feature is responsible for gathering keywords from the reports written by past fosters and using a combination of topic modeling and sentiment analysis to establish certain characteristics for the pets. These characteristics can be used to create a better matching algorithm between pets and families.~~

System Design

This section describes the system design in detail. An overview of the major system components and their roles will be provided, followed by more detailed discussion of components. Component discussion will reference the technical features the component helps satisfy. The design pattern we use is the adapter design pattern.

Architecture Overview

High-level System Architecture



Major Components

GUI Components

L.1, L.2, L.3, L.4, L.5
The GUI components are responsible for showing the main menu, editing and adding new pet/family profiles, presenting a scrollable list of profiles, getting the statistics of the current database, and portraying the matching algorithm to the user in real time.

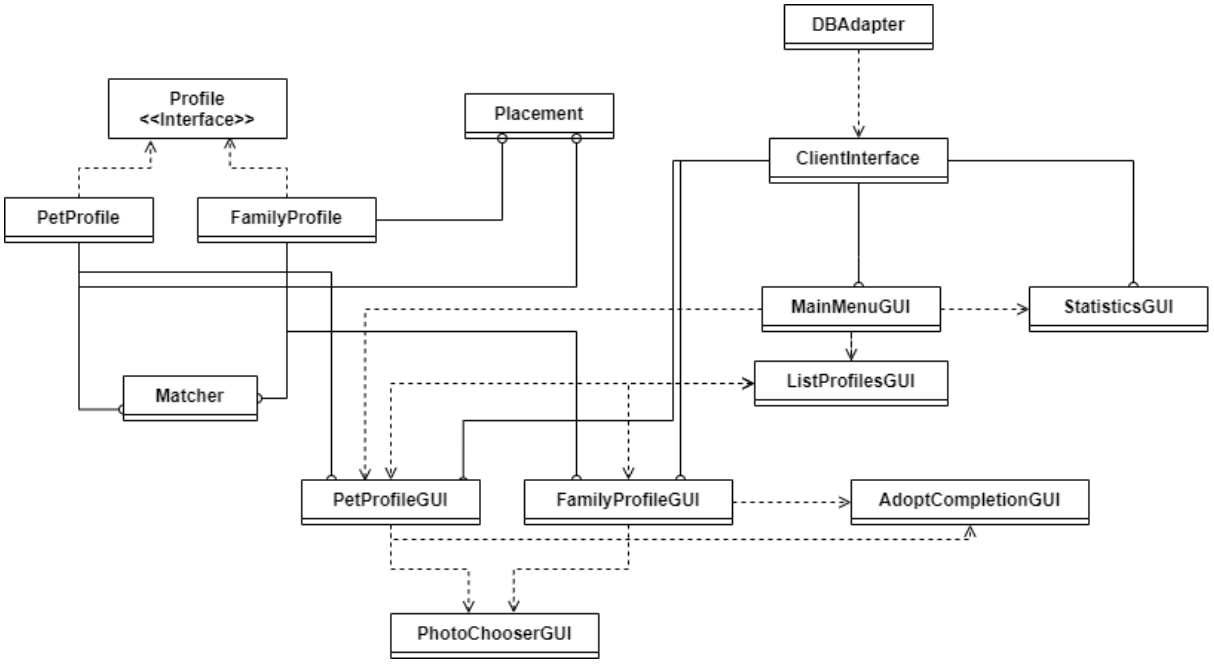
Profile Components

C.2
The profile components should be responsible for having features that collectively belong to both family and pets. These components should be able to get data from the database and set profiles.

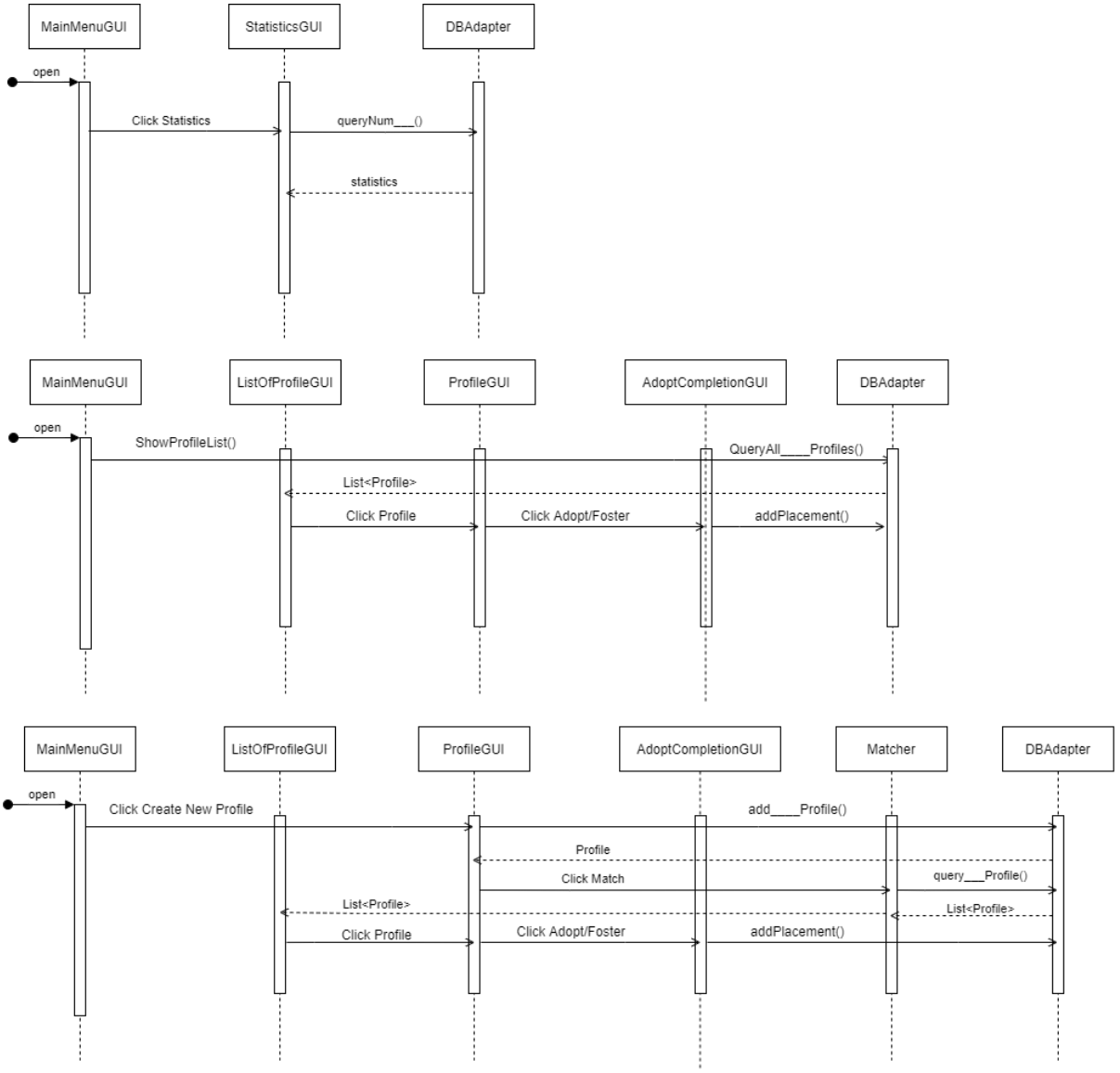
Database

C.6
Relational database containing multiple tables holding all information on existing pets, families, and statistics .

Detailed Design



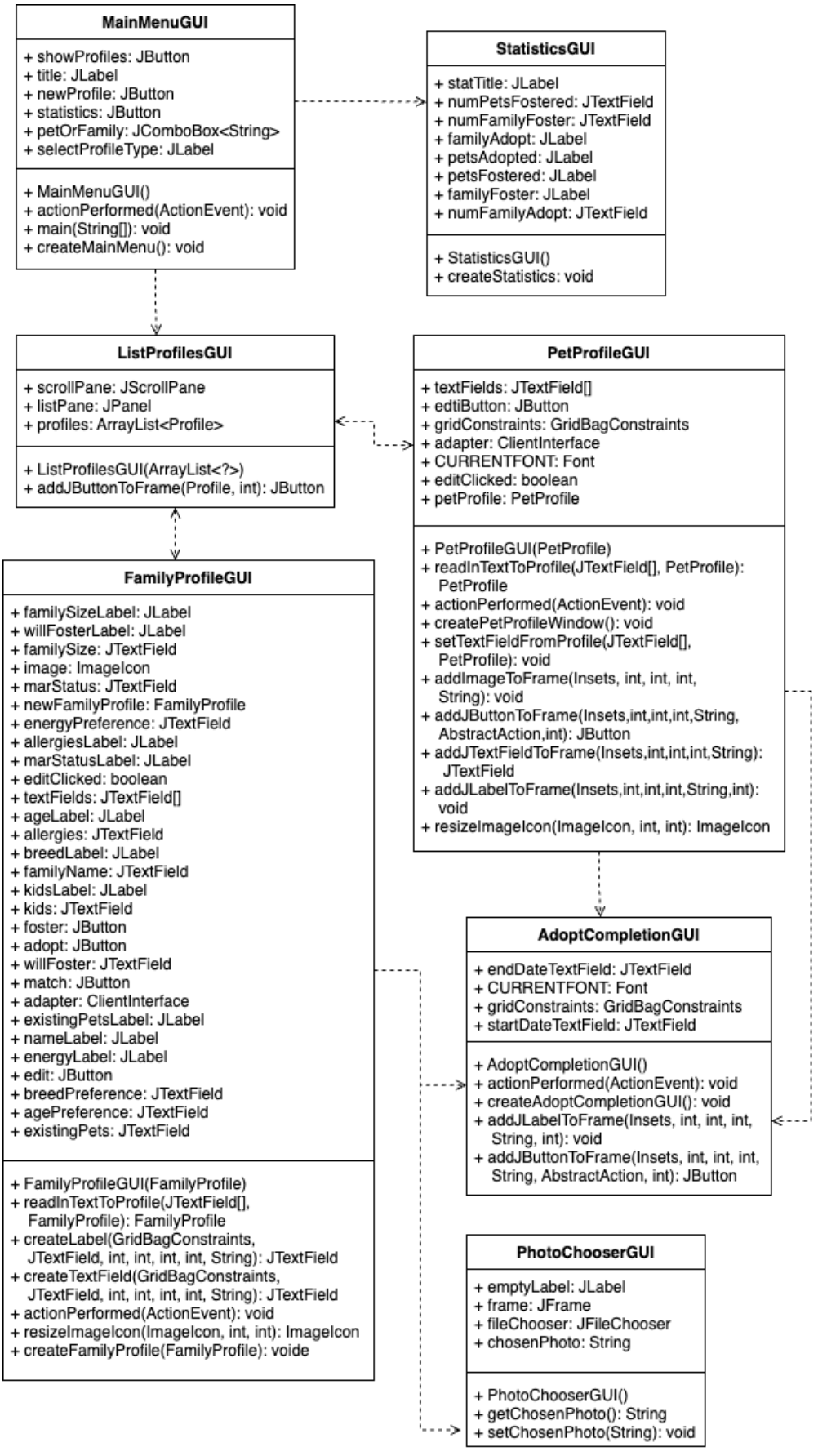
Call Design



The program user may interact with the program in several different ways, including adding profiles, querying lists of profiles, searching for ideal matches, creating records of fosters and adoptions, and querying statistics on total adoptions and fosters.

GUI Components

The user interacts only with the GUI screens when they run the program. The user can search for profiles, edit and create new profiles, and match pets with families through the GUI windows.


MainMenuGUI
+ showProfiles: JButton
+ title: JLabel
+ newProfile: JButton
+ statistics: JButton
+ petOrFamily: JComboBox<String>
+ selectProfileType: JLabel
+ MainMenuGUI()
+ actionPerformed(ActionEvent): void
+ main(String[]): void
+ createMainMenu(): void
StatisticsGUI
+ statTitle: JLabel
+ numPetsFostered: JTextField
+ numFamilyFoster: JTextField
+ familyAdopt: JLabel
+ petsAdopted: JLabel
+ petsFostered: JLabel
+ familyFoster: JLabel
+ numFamilyAdopt: JTextField
+ StatisticsGUI()
+ createStatistics: void
ListProfilesGUI
+ scrollPane: JScrollPane
+ listPane: JPanel
+ profiles: ArrayList<Profile>
+ ListProfilesGUI(ArrayList<?>)
+ addJButtonToFrame(Profile, int): JButton
PetProfileGUI
+ textFields: JTextField[]
+ edtiButton: JButton
+ gridConstraints: GridBagConstraints
+ adapter: ClientInterface
+ CURRENTFONT: Font
+ editClicked: boolean
+ petProfile: PetProfile
+ PetProfileGUI(PetProfile)
+ readInTextToProfile(JTextField[], PetProfile): PetProfile
+ actionPerformed(ActionEvent): void
+ createPetProfileWindow(): void
+ setTextFieldFromProfile(JTextField[], PetProfile): void
+ addImageToFrame(Insets, int, int, int, String): void
+ addJButtonToFrame(Insets,int,int,int,String, AbstractAction,int): JButton
+ addJTextFieldToFrame(Insets,int,int,int,String): JTextField
+ addJLabelToFrame(Insets,int,int,int,String,int): void
+ resizeImageIcon(ImageIcon, int, int): ImageIcon
FamilyProfileGUI
+ familySizeLabel: JLabel
+ willFosterLabel: JLabel
+ familySize: JTextField
+ image: ImageIcon
+ marStatus: JTextField
+ newFamilyProfile: FamilyProfile
+ energyPreference: JTextField
+ allergiesLabel: JLabel
+ marStatusLabel: JLabel
+ editClicked: boolean
+ textFields: JTextField[]
+ ageLabel: JLabel
+ allergies: JTextField
+ breedLabel: JLabel
+ familyName: JTextField
+ kidsLabel: JLabel
+ kids: JTextField
+ foster: JButton
+ adopt: JButton
+ willFoster: JTextField
+ match: JButton
+ adapter: ClientInterface
+ existingPetsLabel: JLabel
+ nameLabel: JLabel
+ energyLabel: JLabel
+ edit: JButton
+ breedPreference: JTextField
+ agePreference: JTextField
+ existingPets: JTextField
+ FamilyProfileGUI(FamilyProfile)
+ readInTextToProfile(JTextField[], FamilyProfile): FamilyProfile
+ createLabel(GridBagConstraints, JTextField, int, int, int, int, String): JTextField
+ createTextField(GridBagConstraints, JTextField, int, int, int, int, String): JTextField
+ actionPerformed(ActionEvent): void
+ resizeImageIcon(ImageIcon, int, int): ImageIcon
+ createFamilyProfile(FamilyProfile): voide
AdoptCompletionGUI
+ endDateTextField: JTextField
+ CURRENTFONT: Font
+ gridConstraints: GridBagConstraints
+ startDateTextField: JTextField
+ AdoptCompletionGUI()
+ actionPerformed(ActionEvent): void
+ createAdoptCompletionGUI(): void
+ addJLabelToFrame(Insets, int, int, int, String, int): void
+ addJButtonToFrame(Insets, int, int, int, String, AbstractAction, int): JButton
PhotoChooserGUI
+ emptyLabel: JLabel
+ frame: JFrame
+ fileChooser: JFileChooser
+ chosenPhoto: String
+ PhotoChooserGUI()
+ getChosenPhoto(): String
+ setChosenPhoto(String): void

MainMenuGUI

ux.1
It is the Main Menu GUI's job to serve as the directory from which users can navigate to all other GUI windows. The Main GUI will instantiate and open up new GUIs depending on user input. It will have buttons to search for profiles, create new profiles, or show statistics of past pet and family matches. The main menu will have a drop down option to select either pet or family, which will then open the corresponding window when the new profile or list of profile buttons are pressed.

StatisticsGUI

ux.1
This GUI will ask the DB Adaptor to retrieve some statistical information from the database, and then it will display that information to the user. The statistics GUI will have four data points: number of pets fostered, pets adopted, families fostering, and families adopting.

ListProfilesGUI

ux.1
This GUI will display a scrollable list of all the available profiles of a given type. The profile information will be retrieved from the database using the DB Adaptor. Clicking on a profile name will open a new window with the corresponding profile.

PetProfileGUI

ux.1
This GUI displays specific attributes of a pet in a new window. It also has buttons that allow the user to edit the profile or find compatible families for fostering or adopting. The profile also contains buttons to become adopted or fostered, which opens a list of families and updates both the family and the pet in the database once the family is selected. This data is retrieved from the database using the DB Adaptor.

FamilyProfileGUI

ux.1
This GUI displays specific attributes of a family. Similar to the pet profiles, it has options to edit the profile, find a list of matching pets that have compatible characteristics, and adopt or foster a pet. This data is retrieved from the database using the DB Adaptor.

AdoptCompletionGUI

e.1, e.2, e.7
The Adoption Completion GUI facilitates the adopting or fostering matches between pets and families. The window will allow the user to select whether it is an adopt or

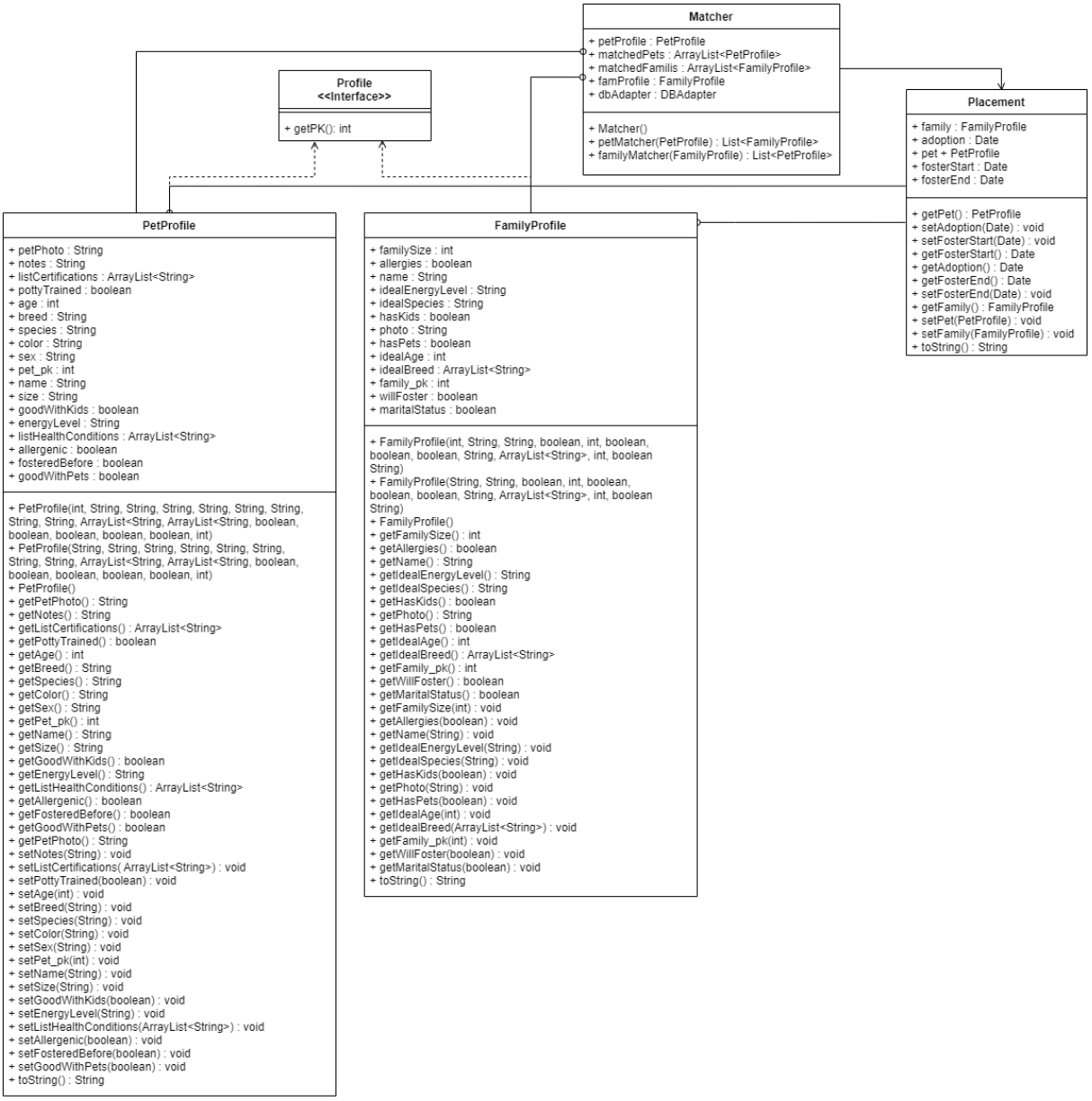
foster match, and then enter the duration of the foster period. This will update the match in the database.

PhotoChooserGUI

C.2, L.3

The Photo Chooser GUI lets the user pick a file from their computer to add as the picture for a new profile. It then reads in the path of the chosen file and adds it to the family or pet in the database.

Profile Components



The profile components are responsible for bridging the gap between the database and the GUI's. The main purpose of these components is to instantiate and get specific profiles to and from the existing database.

Family Profile

C.2

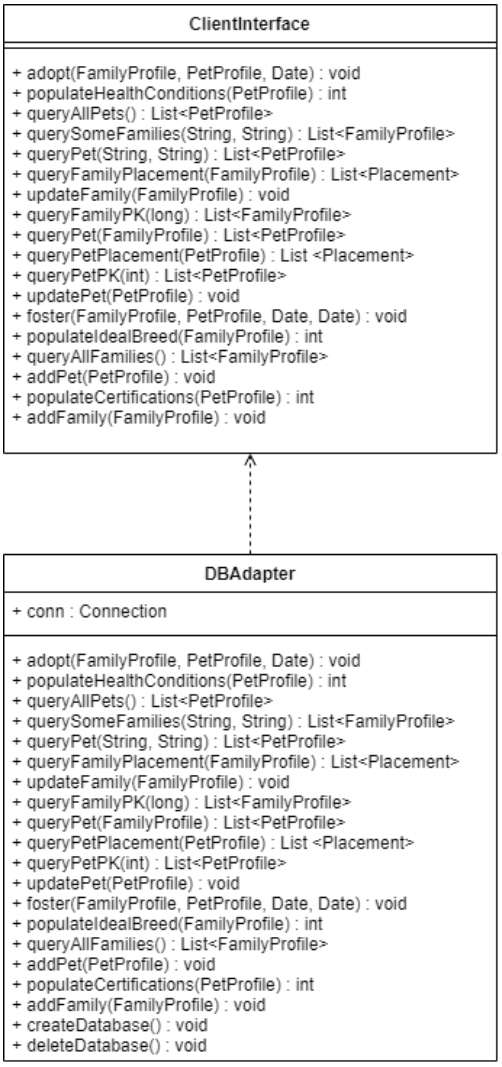
The Family Profile will be a subclass of the Profile class and will call the DB Adaptor to get information from the database about a given family entity, this class will then organize that data for the Profile GUI to display it for the user. It will also pass edited data back into the DB adaptor for writing to the DB.

Pet Profile

C.2

The Pet Profile will be a subclass of the Profile class and will call the DB Adaptor to get information from the database about a given family entity. This class will then organize that data for the Profile GUI to display it for the user. It will also pass edited data back into the DB adaptor for writing to the DB.

Database Components



DB Adaptor

C.3

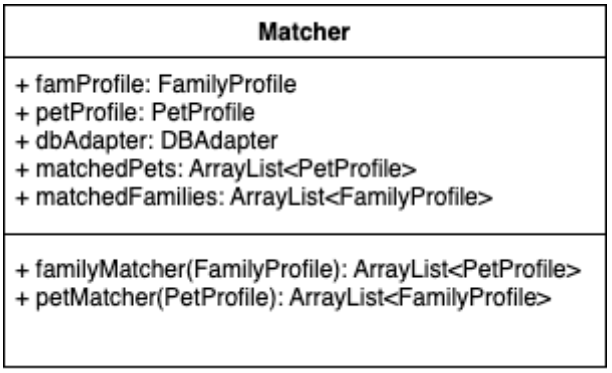
The DB Adaptor and the client facing ClientInterface will make SQL queries to the database and return the results. This class will both read and write data, serving as the primary interface between other classes and the database.

ClientInterface

C.3

The ClientInterface connects the information from the DBAdaptor to the GUI. It takes the information from the database so that it can be displayed in the profiles and list of profiles.

Matcher Components



Matcher

e.6

The matcher should be responsible for getting certain parameters from the user and finding profiles that match the user’s input. The matcher would have different parameters depending on whatever is being searched. This class would have to have access to the existing database to sift through possible matches.

Data Format

The database does not have to exist on any of the user's local networks. For this project all of the data will be instantiated and edited using Structured Query Language (SQL). SQL is a great tool for databases because it already provides mechanisms that enable the engineer to store, retrieve, and manipulate data.