EPICS Fall 2014

Hoosier Veterans Assistance Foundation
(HVAF)

Team Lead: Brandon Weber, Class of 2015
Database Design/Programming: Drew Brandon, Class of 2017
GUI Design/Programming: Ethan Miller, Class of 2017

Hoosier Veterans Assistance Foundation
Outreach Form Revitalization and Digitalization

Table of Contents

Appendix

Abstract

The job that our team accepted involving the revitalization and digitalization of the HVAF outreach form involved reorganizing the fields provided to us, in such a way that would not be difficult for an inexperienced end user and would be easy to design as a database. The form provided to us involved the organization of contact, financial, and personal information into a format that makes sense and flows cleanly. Once the data entry and/or lookup has been completed, the final category to manage is the requested services of the potential client. The initial form involved these categories somewhat intermixed on a physical piece of paper, front and back. Our organization of this information involved a several.page program designed in Microsoft Visual Studio, with each category on its own page. There was also a stipulation that clients who received aid within a thirty day period could receive only a specific subset of the possible aid categories. The first page involved entry of the potential client's surname and last four

digits of their Social Security Number, and after this is entered the program will determine what amount of aid they are eligible to receive, notifying the end user with a red or green pop.up box. After this step, the end user is free to enter the remaining portions of the client's data. The client's eligibility of the aid is pulled from a SQL Server database, which keeps track of each visit and what the client received.

Introduction

Problem Statement and Objectives

Create a program that can enter data as seen on Appendix 1.5 into a relational database using a streamlined form.

Requirements

Create a form to enter the data using Visual Studio
Create a database using SQL Server to connect to the form

Motivation

After a tour of the HVAF Headquarters, our team was shown the closet full of file boxes containing these forms dating back into the 1980's. This data had already been begun to be sifted through by a volunteer and work had been done on this project for several months prior. The amount of analog data the HVAF had stored was nearly ludicrous, and the efforts we put forth towards this project now has the potential to save the HVAF hundreds of hours of work.

REQUIREMENTS SPECIFICATION

R1. Database Management

- SR1. Query for existence of veteran form in database
- SR2. Select certain data from the database if there is a pre-existing record
- SR2. Insert form information into a database.

R2. GUI Interaction

- SR1. Fill out veteran information in a simple and logical form.
- SR2. Provide very clear visual notification based on eligibility.

R3. Print Functionality

- SR1. Print filled out form at 1+ printers in the facility.

R4. Non-Functional Requirements

- SR1. Develop agile class structure following standard software development practices.
- SR2. Carefully diagram and document following standard software practices.

FUTURE REQUIREMENTS

FR1. Convert to a web form.

FR2. Optimize for mobile devices.

FR3. Extend use case beyond one form at one location.

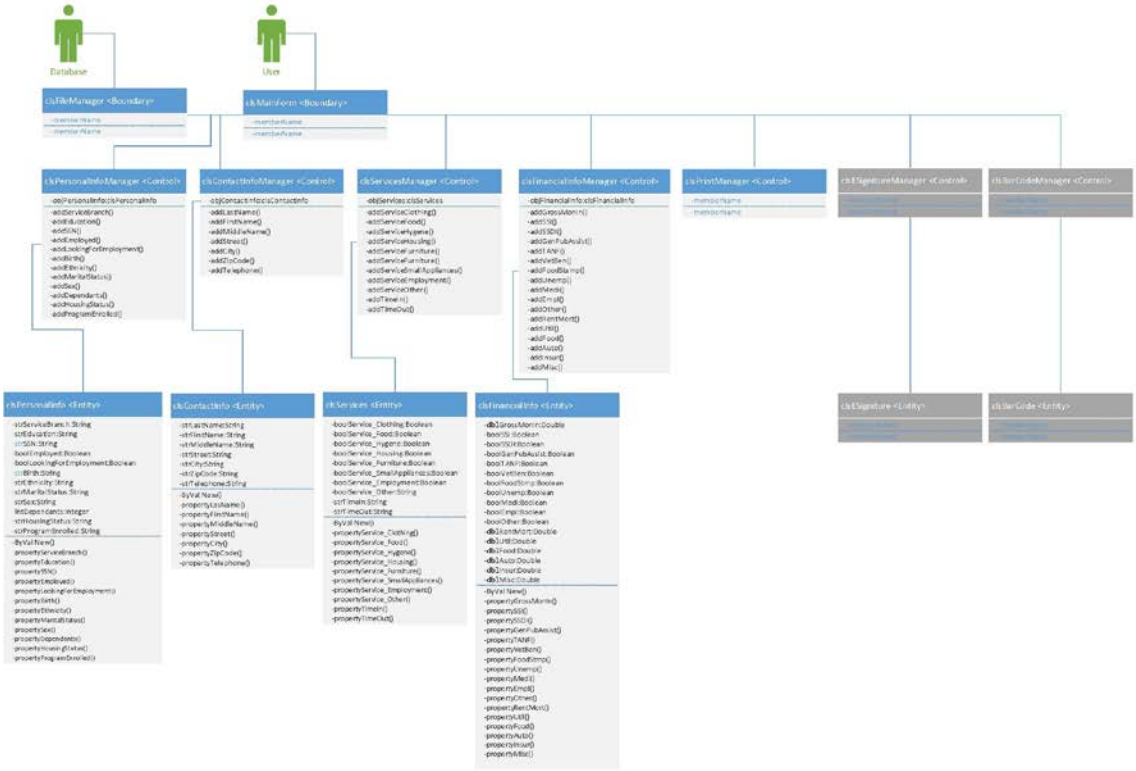
FR4. Develop strong archiving system for quicker data access, but retain robust reporting abilities.

FR5. Develop a live queue that tracks current forms (‘Tickets’) to improve the process.

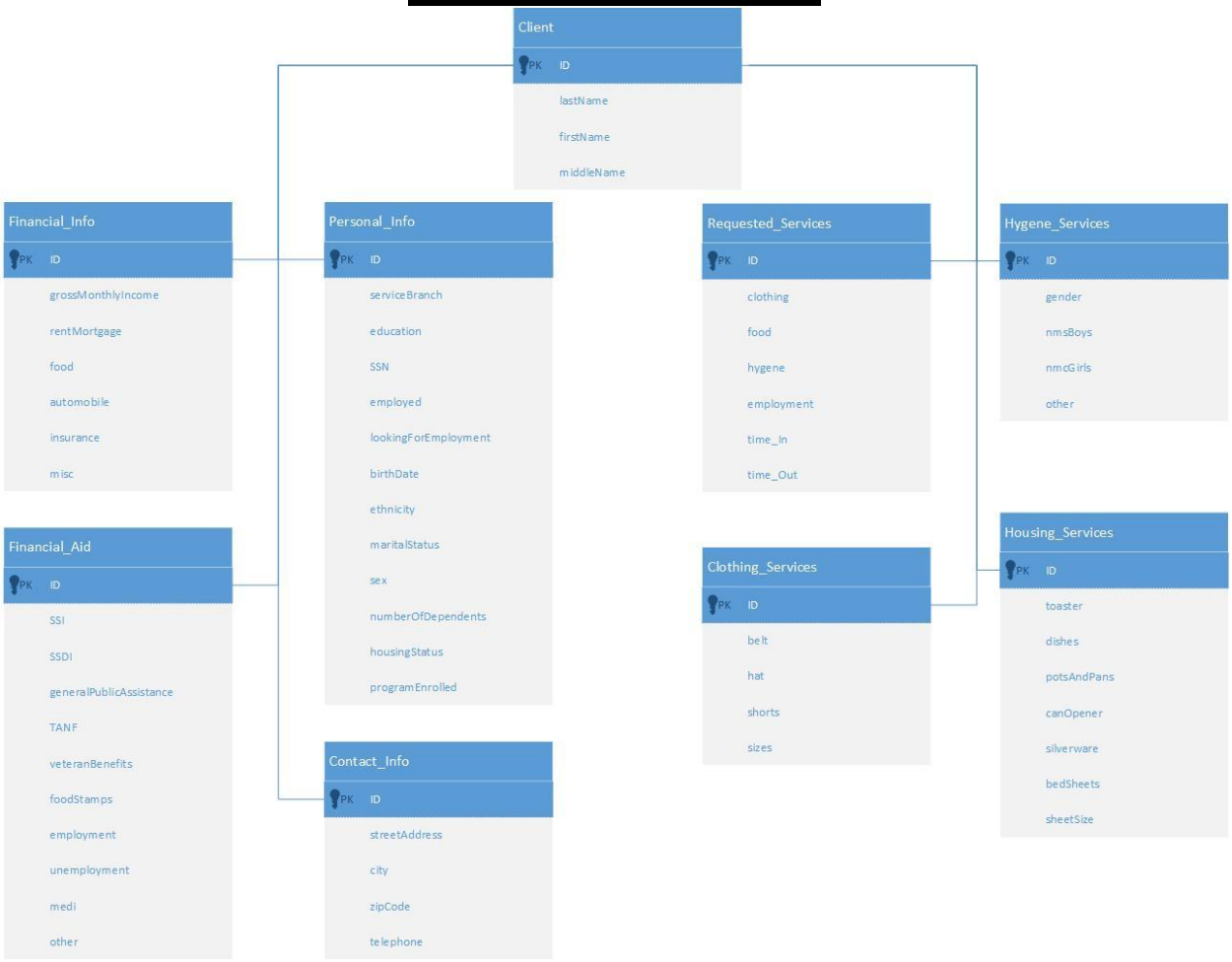
FR6. Implement E-Signature to confirm receipt of goods and services, and encrypt this signature.

FR7. Implement barcode generation and scanning, via iOS device, to record clock in/out times.

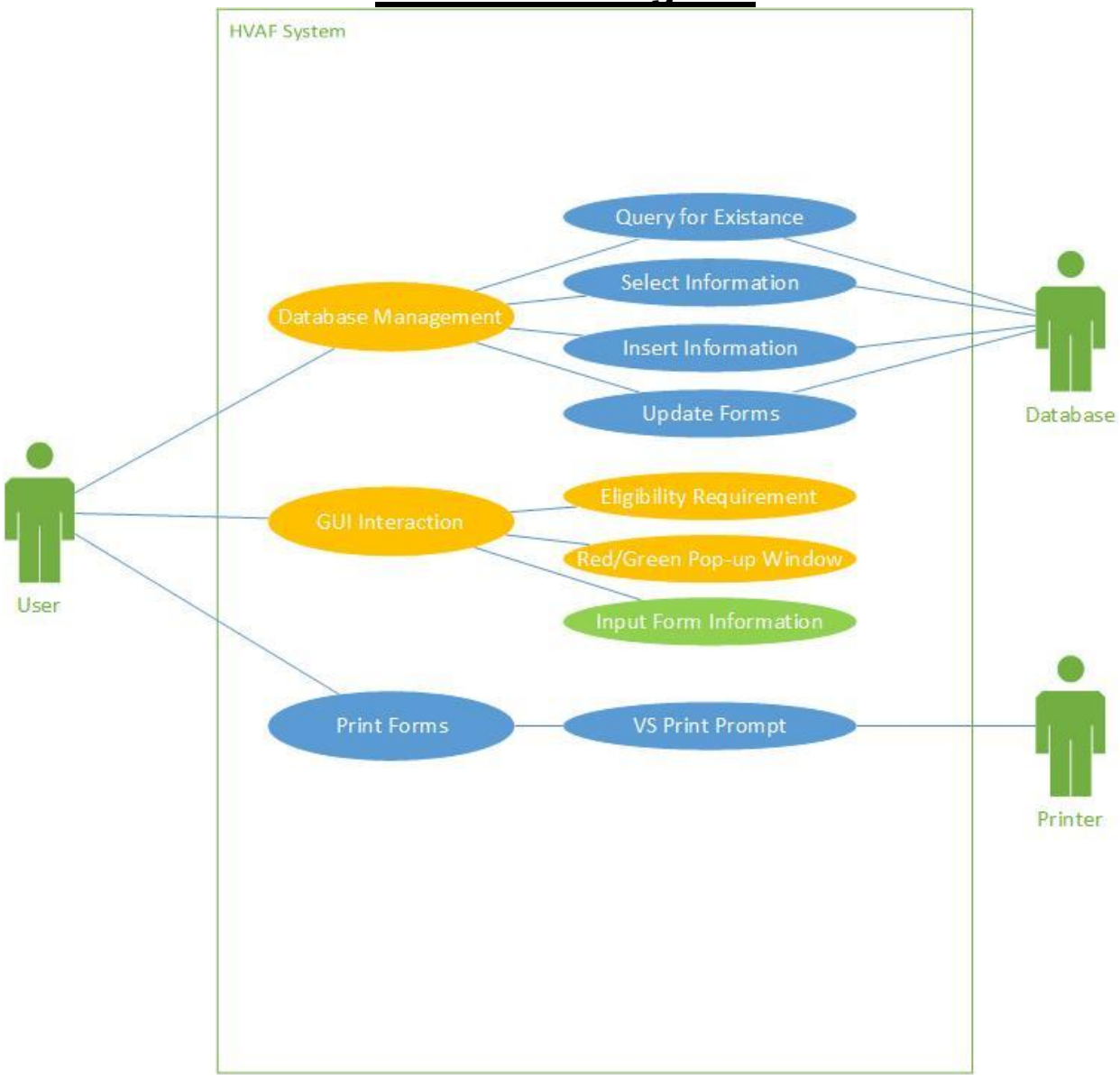
Class Diagram



Database Diagram



Use Case Diagram



Design

Our main goal was to create an interface for the volunteers that is easy to use in order to store information of the clients. Therefore, the design of our program is of incredible importance. The “OutReach Program” that we created is very straightforward and convenient for the volunteers to input client information. We begin with our Client Search page with the option to find an existing client or to create a new client (A.1.1 Client Search). If the user clicks “Search for Client” then a form will bring up that client’s information and show if the client is eligible for any services. If the client is eligible then the program fills multiple information fields, but clears some fields for the user to update. If the person requesting services is not a returning client, then the user has the option to click “New Client.” If this happens, the Personal Information page appears with multiple fields for the user to input the clients information (A.1.2 Personal Information). Once the Personal Information page is completely filled, then the user can go to the Financial Information page (A.1.3 Financial Information). Here, the user will finish completing the form with the client’s information. Once this is done, the user is able to go to the Services Requested page (A.1.4 Services Requested). Multiple separate windows can be opened by checking certain services.

In the main form we also have links on the left side for ease of control. There is also a progress bar at the bottom left side of the form to show the user how far they are with the client and once everything has been filled, then a submit button is visible near the progress bar for the user to submit all of the information up to the database.

In order to progress to the next page of the form, the current page must have all of the information filled out completely. If it is not filled, then there are red markings that show the user which fields still have yet to be completed. Once all fields are complete, the user is able to click “Next” to go to the next page.

Implementation

All of the coding we used for the program is visual basic, as we created the program using *Visual Studio 2013*. The database system, however, is created using *MySQL* thus having a different format. Comments have been added above every function to show exactly what it is doing and how it works. The General User Interface (GUI) was designed by Ethan Miller along with most of the Visual Basic code. The database was designed by Drew Brandon using *MySQL*. The structure and organization of the two were designed by Brandon Weber.

Organization of the Code Base

The class structure was designed with the original paper form in mind. We divided the fields on the original form into a number of distinct categories: personal information, contact information, financial information, and services requested/provided. Each of these categories then became an entity class in the application. Above each entity class we have implemented a control class that deals with pulling data from the top level boundary class and pushing it down to the bottom level entity class, and vice versa. Each entity class had exactly one control class, which was referred to as the manager for that entity class. All of the entity and control classes feed up into the main form, which is the primary graphical user interface (GUI). The entity and control classes also feed into the database which has a very similar structure to the main application. We have also included stubs for the future requirements of the project, which include the addition and implementation of a barcode scanner, E.Signature, and printer connectivity.

Quality Assurance & Testing

Most of the testing was done with Visual Studio, making sure that the GUI worked properly. Otherwise, there was also some testing done with the database, making sure that it could collect the data correctly and as efficiently as possible.

Testing the GUI was very easy, as we would simply start the program and go through a large variety of potential information into the fields, testing if the fields would hold the correct information as well as testing if the buttons would respond correctly to specific information in those fields.

Testing the database for the phase of the project that we worked on was as simple as executing the create statements. With the knowledge that they executed correctly and upon checking the database to ensure that the tables exist as we intended them to, we know that the database works perfectly up to this point. The ability to query certain information and connect the database with the GUI is to be implemented in the next semester of this project.

A user manual for this program is to be written and provided upon full completion of this project. As of now, the HVAF Client Search/Entry Program is still in its infancy and has no practical implementation by end users.

Project Organization & Management

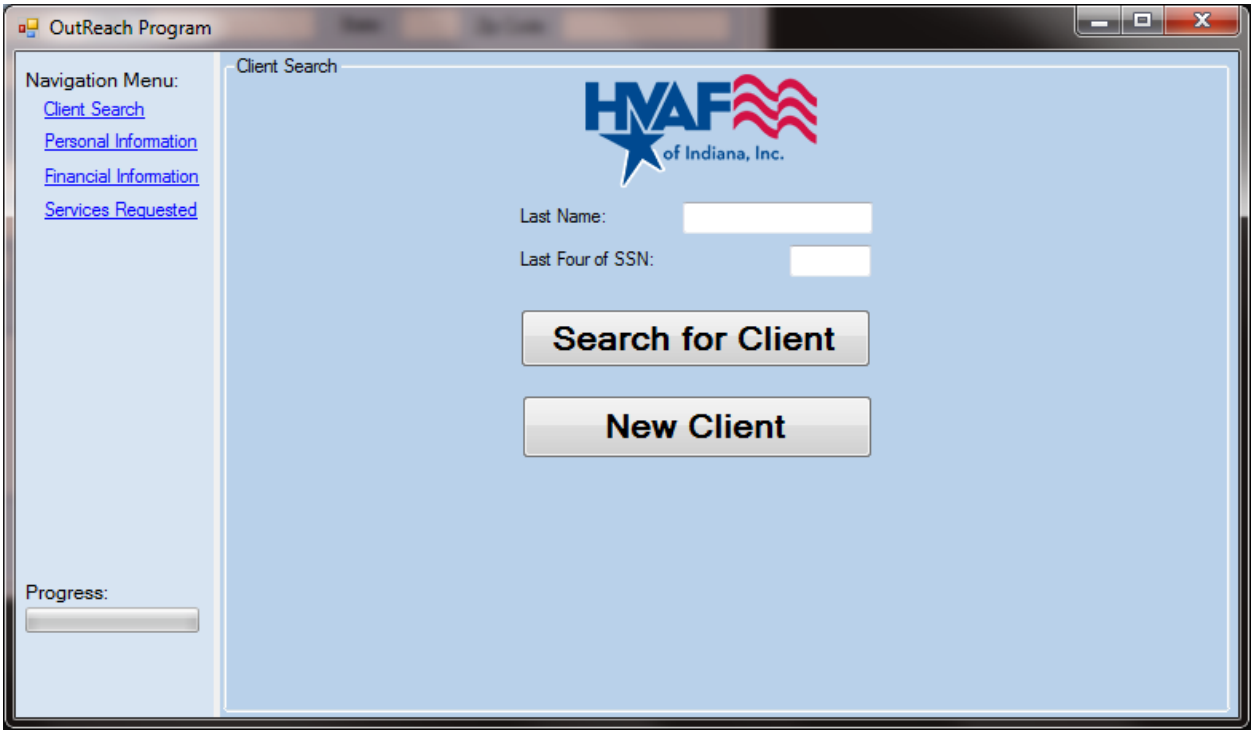
This project was organized under an agile software engineering methodology. One of the cornerstones of this approach is the division of the project into numerous iterations, each of which contains a number of steps. Each iteration began with a requirements analysis that was used to determine what is current completed and what still needs to be done. This was one of the key organizational features of this project because it allowed us to track our progress and keep the current goal directly in front of us. The next step in each iteration was to update and refactor our existing diagrams, which included a use.case diagram, a class diagram, and a database diagram. The structure of the application changed a number of times throughout the semester and it was important to look at our diagrams in order to confirm that we were on the right track. Once we updated our requirements and any relevant diagrams we began work on the code. The division of labor pretty much decided itself, luckily, because Ethan had strong visual studio and user experience skills, Drew had strong SQL database skills, and Brandon worked as the team leader to coordinate work, develop diagrams, and combine the front and back end pieces of the application.

Future Work

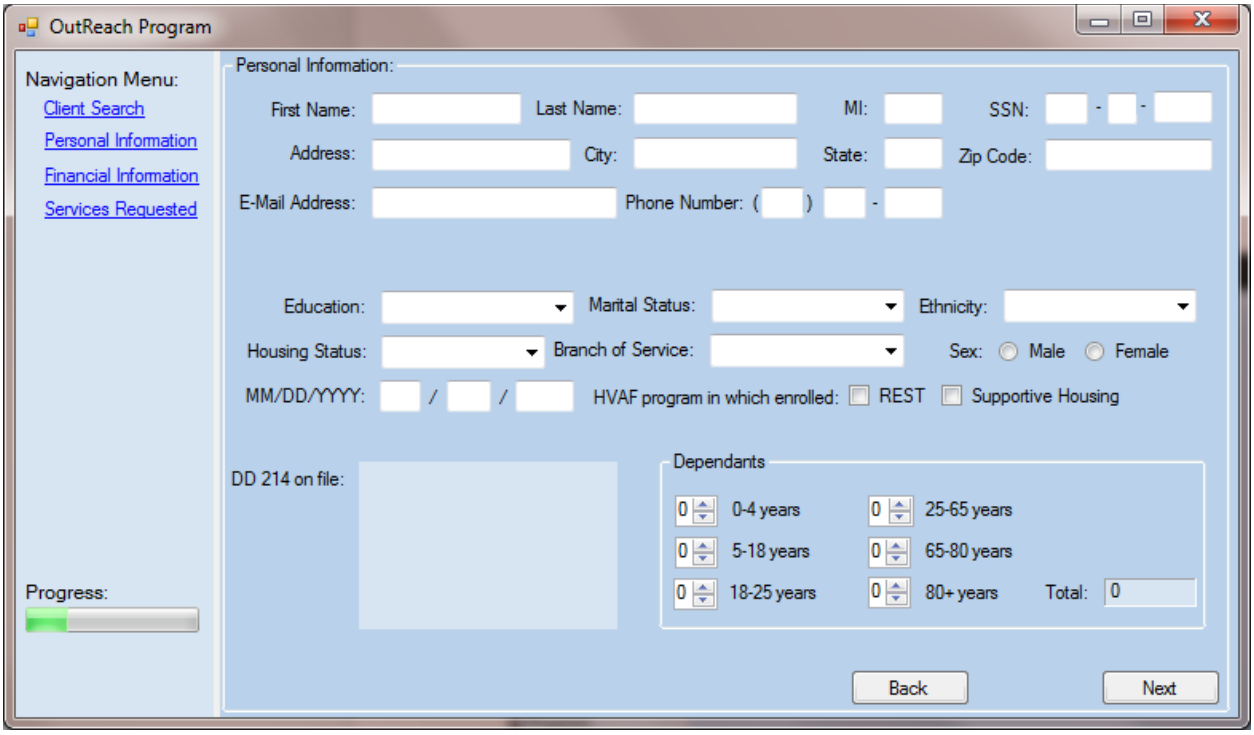
Work that needs be finished in future semesters of this project in the short term includes the connecting of the GUI to the database, writing queries for the database to return the data requested by the HVAF, further organization in the GUI, assistance at the HVAF in setting up a dedicated database server and connection on their network. Long term future work, would include moving this program to a web.based app, implementing the e.signature identity verification method (preferred by the HVAF), and implementing the barcode.scanning capabilities of this program.

Appendix

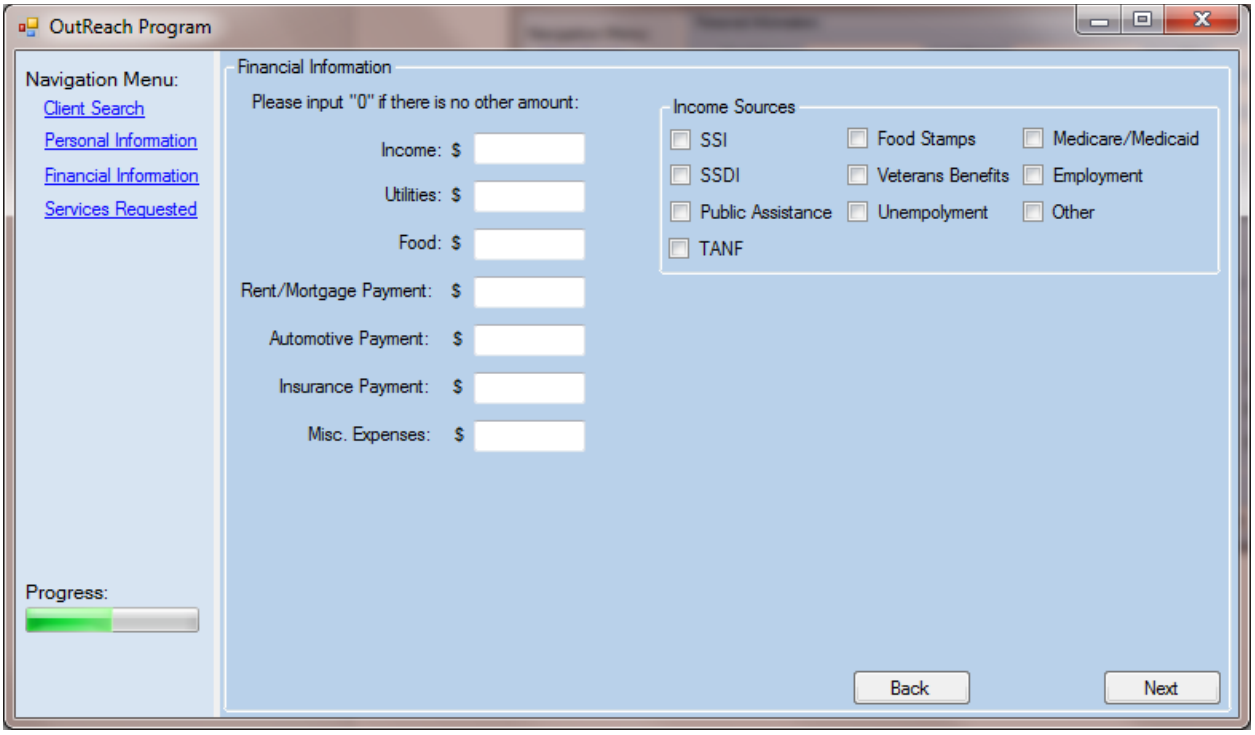
A.1.1 . Client Search Page



A.1.2 . Personal Information Page



A.1.3 . Financial Information Page



A.1.4 Services Requested Page