**B Positive: A Database for Blood Banks, Patients and Donors.**

**Group 5**

|  |  |
| --- | --- |
| **Class:** | 3432 Database Systems |
| **Professor:** | Dr. Mehdi Allahyari |
| **Term:** | Spring 2020 |
| **Effort:** | 70+ hours |
| **Student Names:** | Cody Hutton, Isabelle Harbuck, Whitson Sutherland |
| **Group Number:** | 5 |
| **Assignment Name:** | Final Project: B Positive |
| **Date of Submission:** | 5/1/2020 |

**Table of Contents**

Page 3: Group Members and Introduction

Page 4-5: Scope

Page 5: Functional Requirements

Page 6: Schema

Page 7: ER diagram

Page 8-10: ER diagram converted to relations

Page 10: Map ER diagram to tables

Page 11: Code for the Tables

Page 12: Referential Constraints

Page 12-14: SQL queries

Page 14: UML

Page 15: Web-based application scope

Page 15-18 : Initial web-app mock up

Page 19: Redesigned Application Features

Page 20-28 : Updated GUI

Page 29: Executive summary

**B Positive**

**Group Members:**

Cody Hutton - Made ER diagram/relation, created database, helped on html and final report.

Isabelle Harbuck- Wrote most of the report, contributed to the queries, created the presentation.

Whit Sutherland - Wrote the code for the web applet (html & php) and created the demonstration video.

**Introduction:**

B Positive is a database system available for patients, blood banks, donors and hospitals so they can have an easy way to display patient, donor, and blood bank data and to find matches for a blood transfusion. The purpose of this database is to make it much easier not only for the patients but also for hospitals to display, delete, update, insert and query the database. Our web-interface will allow new patients to fill in their information or to login using a password-like system. If the patient or donor is logging in they will use their name as a username and their SSN as a login. Blood banks will store donors information after they have given blood. If the patient is new we will have them fill in their information on the website and the database will store this information into the database. The patient will also have a choice to type in their blood type to find a match for a blood transfusion. The website will take their blood type and find a match in the database. We will then check to make sure that specific Donor’s blood is not diseased. Once we check that we will inform the patient that we found a match. We will also allow patients and donors to look at their own information stored in the system as well as allowing them to delete their information. The website will also display information about the blood bank itself and give the user an option to update the blood bank’s phone number. For this project we used MySQL Workbench and terminal for the database, Creatively for the ER diagram, Mockaroo.com for the dummy data and Xampp for the website’s code.

**Scope:**

Project details:

For this project, we attempted to create a blood donation database which stores data about patients, donors, blood banks and the donated blood. Once the donor logins using their name and SSN as a password the website will display all of the donor’s information which is ID, name, phone number, address, SSN, birthday and blood type. Patients will also have to login the same way as donors to have their information displayed. Once a patient is logged in it will display their ID, name, phone number, address, SSN, birthday and blood type much like the donor’s page. However, patients will also be given the choice to insert their information onto the database if they are a new patient and look for a match for a blood transfusion by typing in what blood type they have. In other words the patient can query the database to find a compatible donor. On the blood bank’s part of the database we will store the name of the blood bank, its phone number, address, and the names of donors who have donated their blood to that bank. On the web-based application the user can look at all of this information as well as update the phone number by typing in the name of the blood bank which you would like to change as well as the new phone number. The main use of the project is to allow donors and patients to view information regarding themselves as well as patients ability to search for matches.

Problem statement:

When a patient needs a blood transfusion it can be hard for them or the hospital to search for a match as well as an easy way to display patients, donor and blood bank information. It is also hard for hospitals, donors, patients and blood banks to find a way to easily delete, and update their information. This project will solve this issue by being able to match patients with the right blood they need for their transfusion, making it easy for patients and donors to delete their information and for blood banks to update their information.The information regarding this database is too large and complex for an excel file. Queries such as finding compatible donor-patient matches as well as relations including multiple tables through foregin keys would make MySQL more ideal. Therefore we used MySQL Workbench and Terminal for the database and Xampp for the website.

Target user:

The targeted user of this database is individual persons, to third party companies, to full hospitals. Patients and donors (individual persons) will have access to it for information regarding their person as well as their donor-patient relationship. Administers of the database would be blood drive IT leads or hospital IT leads. The hospital or blood drive would maintain the database and information. Individuals will have restricted access to their information when logging in.

A real life example would be for a blood drive, the drive would fill in the database, automatically putting in donor information from physical forms at the end of the drive. With donor information populated in the database, and the blood in bags, patients create accounts through the web app and search for compatible matches.

Data:

Our data will be populated through both external sources and from ourselves. Patients will fill in their own data to be stored into the database and we will fill in the Blood Bank, Donor Data, and Blood data.

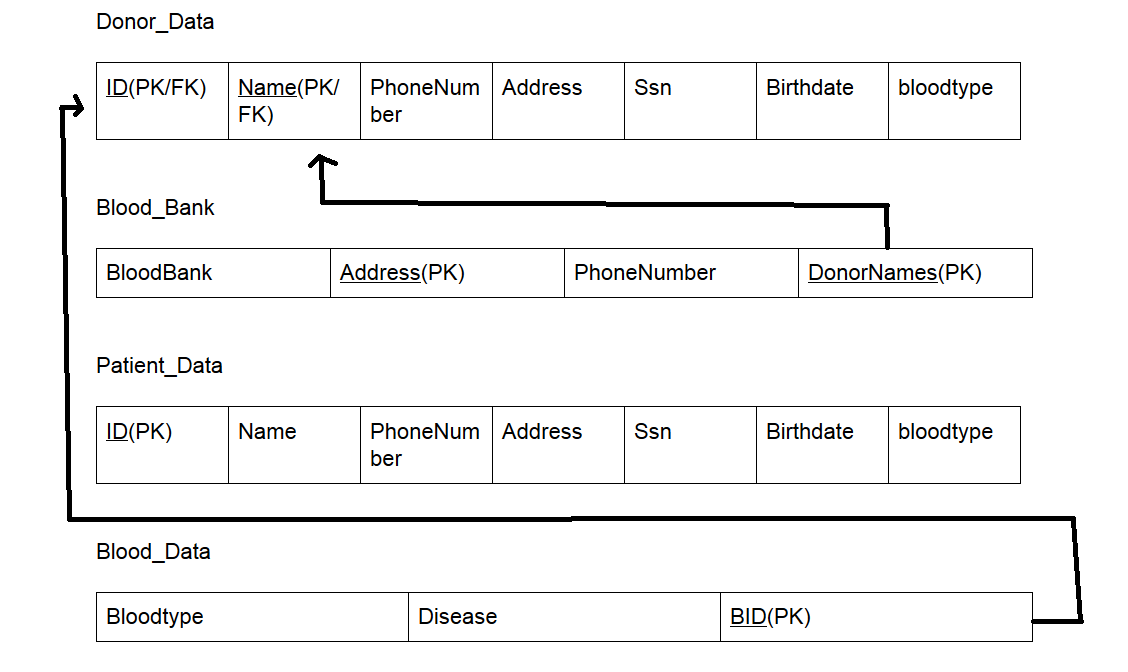
**Functional Requirements:**

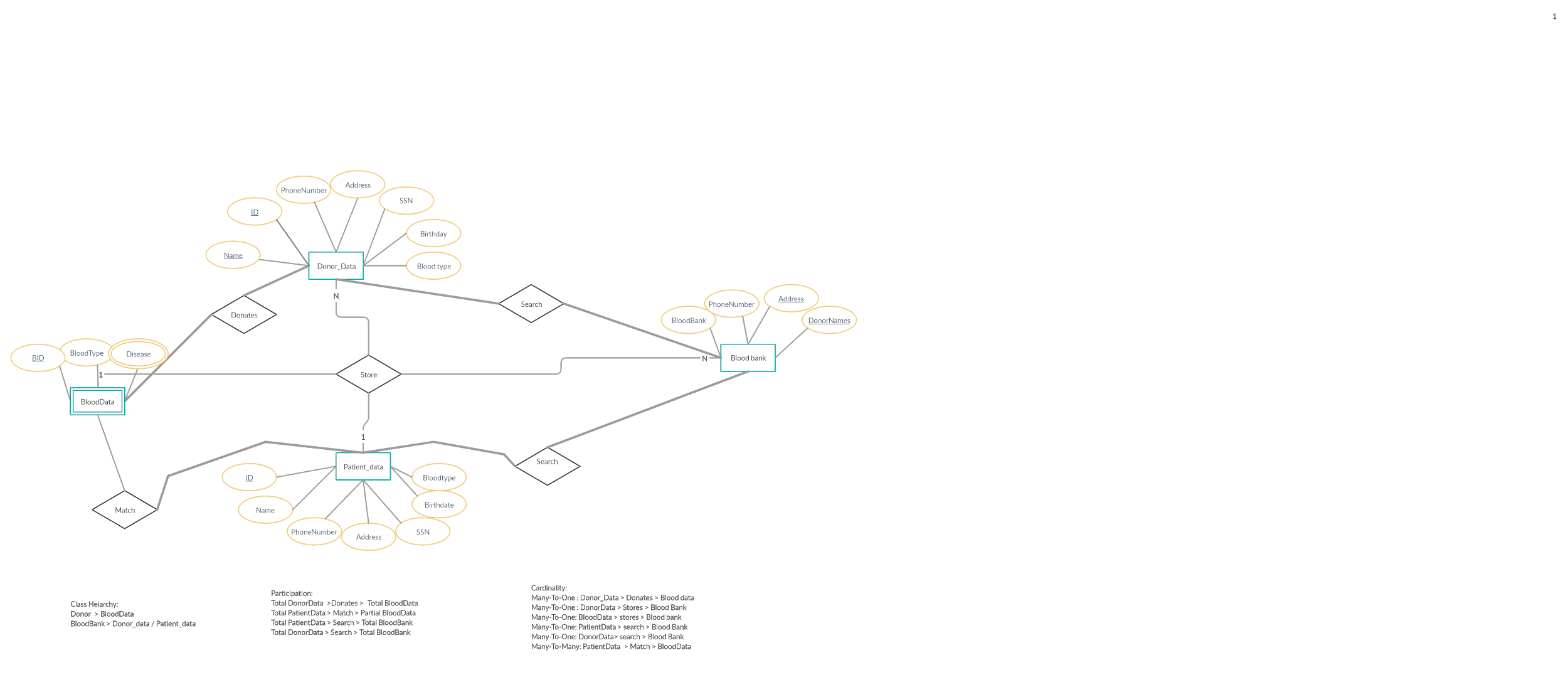
Availability: This application will be used locally. Ideally this application would be able to run simultaneously by all patients and donors but we do not have the resources or knowledge to pull this off. Instead we used it for a smaller amount of users.

Security: Our database is not very secure but we do have a password-like system. This system allows us to assign the patient to the correct information as well as using a password to protect the patient and donors information. This system however is not perfect because it uses SSN as a password and the patient’s/donor’s name as a username.

Reliability: Our system is reliable when used correctly and with not that many people trying to use it at the same time.

**Schema:**



**ER Diagram:**

**ER Diagrams Converted into Relations:**

Donor\_Data:

The attributes for *Donor\_Data* are *ID, Name, PhoneNumber, Address, SSN, Birthday* and *BloodType.* The purpose of the *ID* attribute is to have a unique set of numbers to uniquely identify each donor. *ID* also acts as a primary key and a foreign key. *ID* is the data type of varchar. *ID* must have at least 5 numbers but the database can be up to 100 characters. *ID* must also be not null and unique. The purpose of the *Name* attribute is for the donor to have another form of identification. *Name* is also a primary key and a foreign key. *Name* is the datatype of varchar. *Name* can be up to 100 characters long. *Name* also must be not null and must be unique. The purpose of the *PhoneNumber* attribute is so the blood bank has a way to contact each donor. *PhoneNumber* is the datatype of varchar and needs to be 10 numbers but the database will accept up to 100 characters. *PhoneNumber* can be null and the default is null. The purpose of the *Address* attribute is so the blood bank has a way to contact each donor. *Address* is the datatype of varchar and can be up to 100 characters long. *Address* can be null and the default is null. The purposeof *SSN* the attribute is to have a way to identify the donor and to make sure the donor is who they say they are. *SSN* is the datatype of varchar and it must be 7 numbers long but the database accepts up to 100 characters. *SSN* can be null and the default is null. The purpose of the *Birthdate* attribute is to know the age of the donor. *Birthdate* is the datatype of date. *Birthdate* can be null and the default is null. The purpose of the *Blood*Type attribute is so the blood bank can prematurely know the donor's blood type. *BloodType* is the datatype of varchar and can be up to 50 characters long. *BloodType* can not be null. The primary keys for this relation are *ID* and *Name* and the foreign keys are *ID* and *Name*. We choose the *ID* as the primary key because it will be a unique set of numbers for one donor and only one donor. We can use *ID* to easily identify a specific donor. We chose *Name* as a primary key as another way to identify the donor since their names are also in the *Blood\_Bank* table.  *Name* is a foreign key because *Blood\_Bank* needs to know which donor’s blood specifically is at their bank. *ID* is also a foreign key because it needs to be referenced by *BloodData* so it knows which donor’s blood it is specifically referring to. *BloodData* will refer to a specific donor with an already specified blood type. Both foreign keys are cascade deletes so if they are deleted in this table they will also be deleted in the tables that reference it.

Blood\_Bank

The attributes for *Blood\_Bank*are *BloodBank, Address, PhoneNumber, and DonorNames.* The purpose of the *BloodBank* attribute is to have a name for the specific blood bank. Each blood bank needs a name to make it easier for donors and patients to identify. It is also another form of identification for the database. *BloodBank* is the datatype of varchar and it can be up to 100 characters. *BloodBank* can be null and its default is null. The purpose of the *Address* attribute is for a way for donors and patients to find and contact the blood bank and it is a way to uniquely identify each blood bank since there can only be one location of each bank. This is why *Address* is a primary key. *Address* is of the data type of varchar and can be up to 100 characters long. *Address* can not be null. The purpose of the *PhoneNumber* attribute is so the patients and donor have a way to contact a specific blood bank. *PhoneNumber* is of the data type of varchar and it must be at least 10 numbers long but the database will accept up to 100 characters. *PhoneNumber* can not be null.The purpose of the *DonorNames* attribute is for the blood bank to know what specific donor is donating their blood to the bank. *DonorNames* is of the data type of varchar and can be up to 100 characters long. *DonorNames* cannot be null. The primary keys for this relation are *DonorNames* and *Address*. *DonorNames* is a primary key because it is a way to identify the blood bank and identify the donors who gave blood to this bank. Since there is only one location for each blood bank we can use *Address* to uniquely identify each blood bank. *Blood\_Bank’s* *DonorNames* also refer to *Donor\_Data’s* foreign key *Name.* It refers to *Name* so the *Blood\_Bank* knows what donor specifically is donating blood. When a donor is deleted they will also be deleted from this table since it is set to delete cascade.

Patient\_Data

The attributes for *Patient\_Data*are *ID, Name, PhoneNumber, Address, SSN, Birthdate* and *BloodType.* The purpose of the *ID* attribute is to have a unique set of numbers to uniquely identify each patient. *ID* also acts as a primary key. *ID* is the data type of varchar. *ID* must have at least 5 numbers but the database accepts up to 100 characters. *ID* cannot be null. The purpose of the *Name* attribute is for the patient to have another form of identification. *Name* is the datatype of varchar. *Name* can be up to 100 characters long. *Name* can be null and the default is null. The purpose of the *PhoneNumber* attribute is so the blood bank has a way to contact each patient. *PhoneNumber* is the datatype of varchar and needs 10 numbers but the database accepts up to 100 characters. *PhoneNumber* can be null and the default is null. The purpose of the *Address* attribute is so the blood bank has a way to contact each patient. *Address* is the datatype of varchar and can be up to 100 characters long. *Address* can be null and its default is null. The purposeof *the SSN* attribute is to have a way to identify the patient and to make sure the patient is who they say they are. *SSN* is the datatype of varchar and it must be 7 numbers long but the database accepts up to 100 characters. *SSN* can be null and the default is null. The purpose of the *Birthdate* attribute is to know the age of the patient. *Birthdate* is the datatype date. *Birthdate* can be null and the default is null. The purpose of the *Blood*Type attribute is so the blood bank can prematurely know the patient’s blood type. *BloodType* is the datatype of varchar and can be up to 50 characters long. *BloodType* can be null and its default is null. The primary key for this relation is *ID* and the foriegn key is *ID.* We choose the *ID* as the primary key because it will be a unique set of numbers for one patient and only one patient. We can use *ID* to easily identify a specific patient. There are no foreign keys in this table. When a patient is deleted no action is taken so we can preserve patient history.

BloodData

The attributes for *BloodData*are *BloodType, Disease,and BID.* The purpose of the *BloodType* attribute is for the blood bank to know what blood type the donor has. This will make it easier for patients to find matches. *BloodType* is of the data type varchar and it can be up to 50 characters long. *BloodType* cannot be null.The purpose of the *Disease* attribute is to know if the donor’s blood is usable. Sometimes people have diseases or other problems in their bloodstream making it unsafe to use for transfusion. *Disease* is of the data type varchar and can be up to 100 characters long. *Disease* can not be null. The purpose of the *BID* attribute is to uniquely identify each donor hence why it is the primary key. It also refers to *Donor\_Data’s* foreign key *ID* because it will be referring to a specific donor that already exists in the database. *BID* is a datatype of varchar. It must be at least 5 numbers long but the database can accept up to 100 characters. *BID* cannot be null.The primary key for this relation is *BID*. *BID* is a primary key that refers to *Donor\_Data’s* primary key *ID*. It refers to *ID* because It will be referring to a specific donor that already exists in the database. *BID* is a primary key so it knows what specific donor’s blood the tuple is referring to. *BID* uniquely identifies each donor. When a donor is deleted they will also be deleted from this table since it is set to delete cascade.

**Mapping from the ER diagram to the Relation:**

The entities in our ER diagram consist of “Donor Data,” “Blood Bank,” Patient Data,” and a weak entity “Blood Data.” The relations used are defined as “Search,” “Donates,” “Stores” and “Match.” Donor data is connected to Blood Bank using the relation ”Search.” This is so the information in the blood bank information will have access to the data stored in Donor Data for when the database is searched for corresponding information. The other instance of the relation “Search” is between patient data and blood bank. This is so that the information stored in patient data is accessible to the Blood Bank when performing the action of searching. It is very similar to the prior use between Blood Bank and Donor Data. Another relation defined in the ER diagram is that of “Match,” which goes between the Blood Data and the Patient Data entities. This relation is established so it is possible to check for a match between the blood type of a donor and that of a patient. The relation is essential for being able to match up patients with their correct blood type, but is also very dependent on the next relation “Donates.” Donates is a relation that goes between the Donor Data entity, and the Blood data. The reason this exists is so the blood data information is tied to specific entries in Donor Data (*BID and Bloodtype*). This allows for accurate comparison between the data stored about the donor and about the patient, allowing for an accurate match to be made. This is also why *BID* and *Bloodtype* are used as foriegn keys. The last relation used in the diagram is “Stores''. Stores are used by all entities. Each entity stores information from each other.

**Code for the Tables:**

create table Donor\_Data (

ID int not null unique, #Donor ID

Name varchar(100) not null unique,

PhoneNumber varchar(100) default null,

Address varchar(100) default null,

ssn varchar(100) default null,

birthdate date default null,

Bloodtype varchar(50) not null,

primary key(Name,ID)

);

create table Blood\_Bank (

BloodBank varchar(100) default null,

Address varchar(100) not null,

PhoneNumber varchar(100) not null,

DonorNames varchar(100) not null,

primary key(Address,DonorNames),

constraint bloodName foreign key(DonorNames) references Donor\_Data(Name) on delete cascade

);

create table Patient\_Data (

ID int not null, #patient ID

Name varchar(100) default null,

PhoneNumber varchar(100) default null,

Address varchar(100) default null,

ssn varchar(100) default null,

birthdate date default null,

bloodtype varchar(50) default null,

primary key(ID)

);

create table Blood\_Data (

BloodType varchar(50) not null,

Disease varchar(100) not null,

BID int not null,

primary key(BID),

constraint BloodID foreign key(BID) references Donor\_Data(ID) on delete cascade

);

**Code for the Referential Integrity Constraints:**

constraint BloodID foreign key(BID) references Donor\_Data(ID) on delete cascade

constraint bloodName foreign key(DonorNames) references Donor\_Data(Name) on delete cascade

**SQL Queries**

--Query 1

--For each patient retrieve their names

SELECT Name

FROM Patient\_Data;

--Query 2

--For each donor retrieve their names

SELECT Name

FROM Donor\_Data;

--Query 3

--For each patient retrieve their ssn and birthdate

SELECT name,ssn,birthdate

FROM patient\_data;

--Query 4

--Retrieve all blood bank names

SELECT distinct BloodBank

FROM Blood\_Bank;

--Query 5

--Retrieve Donor name and SSN for those that donated to Thompson LLC

SELECT name, ssn

FROM donor\_data, blood\_bank

WHERE name = DonorNames

And Bloodbank like ‘Thompson LLC’;

--Query 6

--Retrieve all patients whose name start with J

SELECT name

FROM Patient\_Data

WHERE name like ‘j%’;

--Query 7

--Retrieve age of all the donors

SELECT name, year(curdate()) - year(birthdate) as age

From donor\_data;

--Query 8

--Retrieve all Donors with blood type of o+

SELECT name, bloodtype

FROM donor\_Data

WHERE bloodtype like ‘o+’;

--Query 9

--Retrieve address and phone number for the blood bank named Schoen LLC

SELECT phonenumber, address

FROM Blood\_Bank

WHERE BloodBank = ‘Schoen LLC’;

--Query 10

--Retrieve blood bank names that have the phone number 348-378-3223

SELECT distinct bloodbank

FROM Blood\_Bank

WHERE phonenumber = ‘348-378-3223’;

--Query 11

--Match the donor names to the blood bank they donated to

SELECT d.name, b.bloodbank

FROM donor\_data as d, blood\_bank as b

WHERE d.name = b.DonorNames

ORDER BY bloodbank;

--Query 12

--Retrieve Blood id (BID) with perfect blood

SELECT BID

FROM Blood\_Data

WHERE disease LIKE =’perfect’;

--Query 13

--Retrieve count of non-diseased blood labeled “healthy blood”

SELECT count(\*) as “Healthy Blood”

FROM blood\_data

WHERE disease like ‘Perfect’;

--Query 14

--find the donors who have blood that's not perfect

SELECT name, disease

FROM donor\_data,blood\_data

WHERE BID = ID

AND disease not like ‘Perfect’

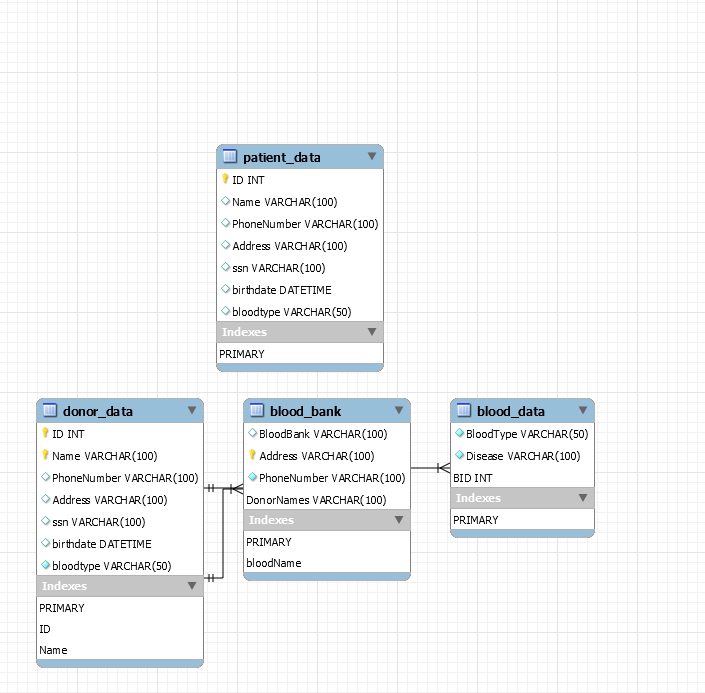
ORDER BY disease;

-Query 15

--Find the count of blood donated from each donor, by type  
Select count(\*), bloodtype

From blood\_data

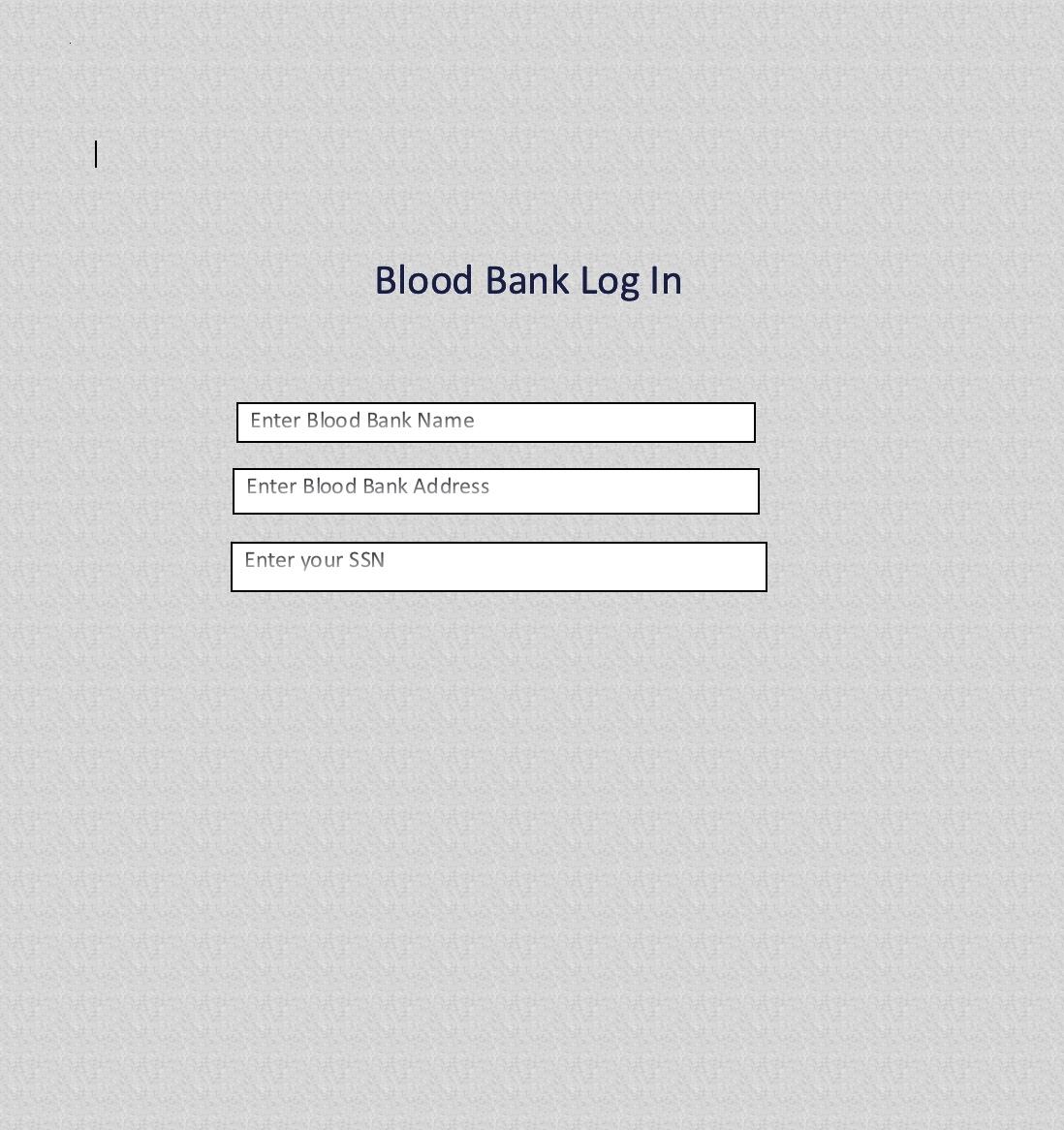
Group by bloodtype;

**UML:  
  
**

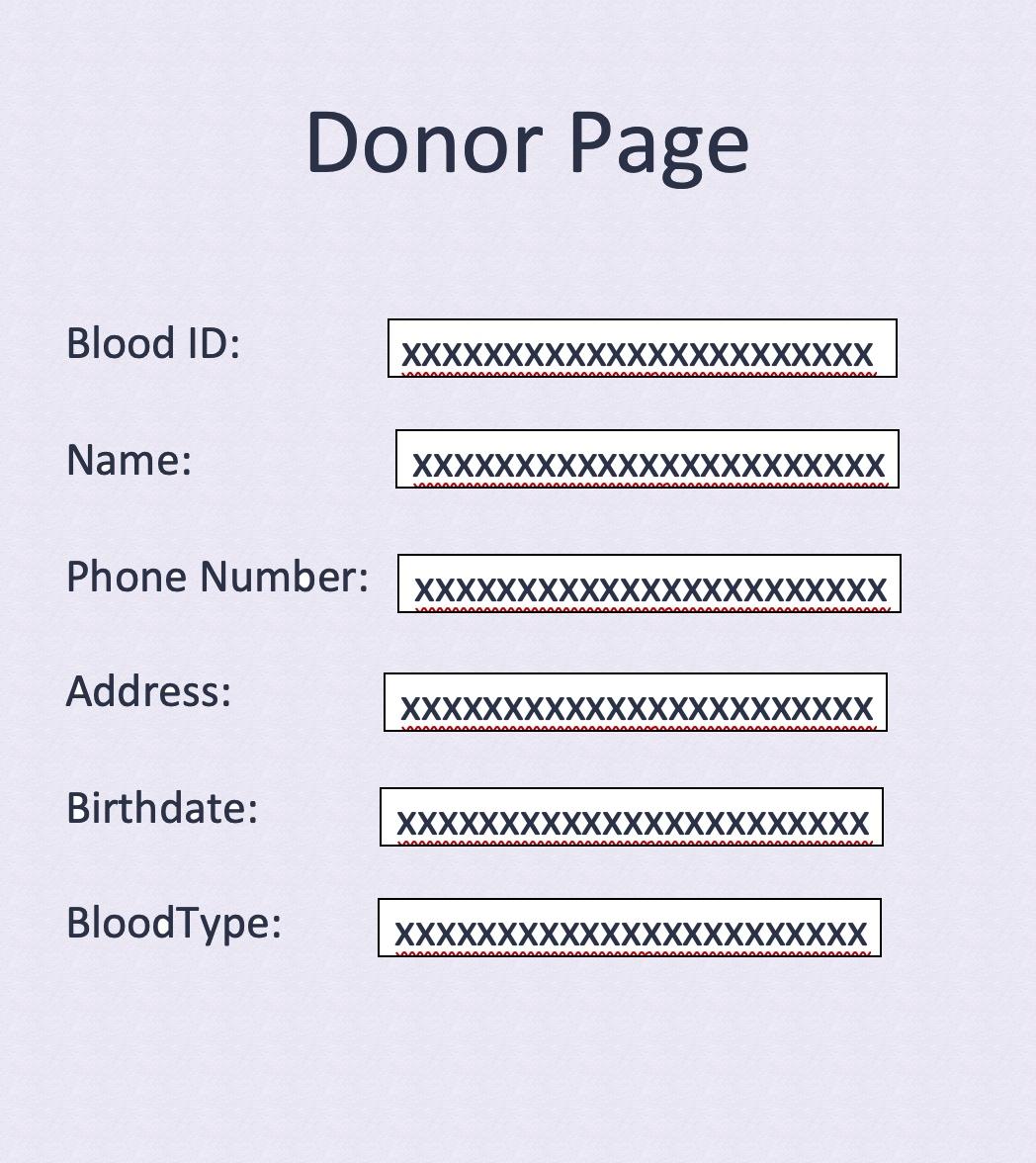
**Web-based Application Scope:**

The purpose of the web-based application is to make it easy on the user to display, delete, update and query the database. We tried our best to make it as easy as possible on the user with fun graphics to make it more visually appealing. The web-based application will allow patients to log in. if the patient is new to the database they will be allowed to put in their information which will be stored into the database and it will automatically create a corresponding account using the patients SSN number and name to find their information. Once the patient logs in they can see their information, have the choice of searching for a match for a blood transfusion or to delete their information. Donors can also put in their SSN and name to login so they can have their information displayed. Donors do not have the choice to put in information to be added to the database like patients because blood banks are supposed to add in the information of donors to the database once they donate blood. Donors also have the choice to delete their information. Blood banks display their information and the names of donors who have previously donated. The *BloodData* table is used during the search of a compatible match to make sure the blood is not diseased.

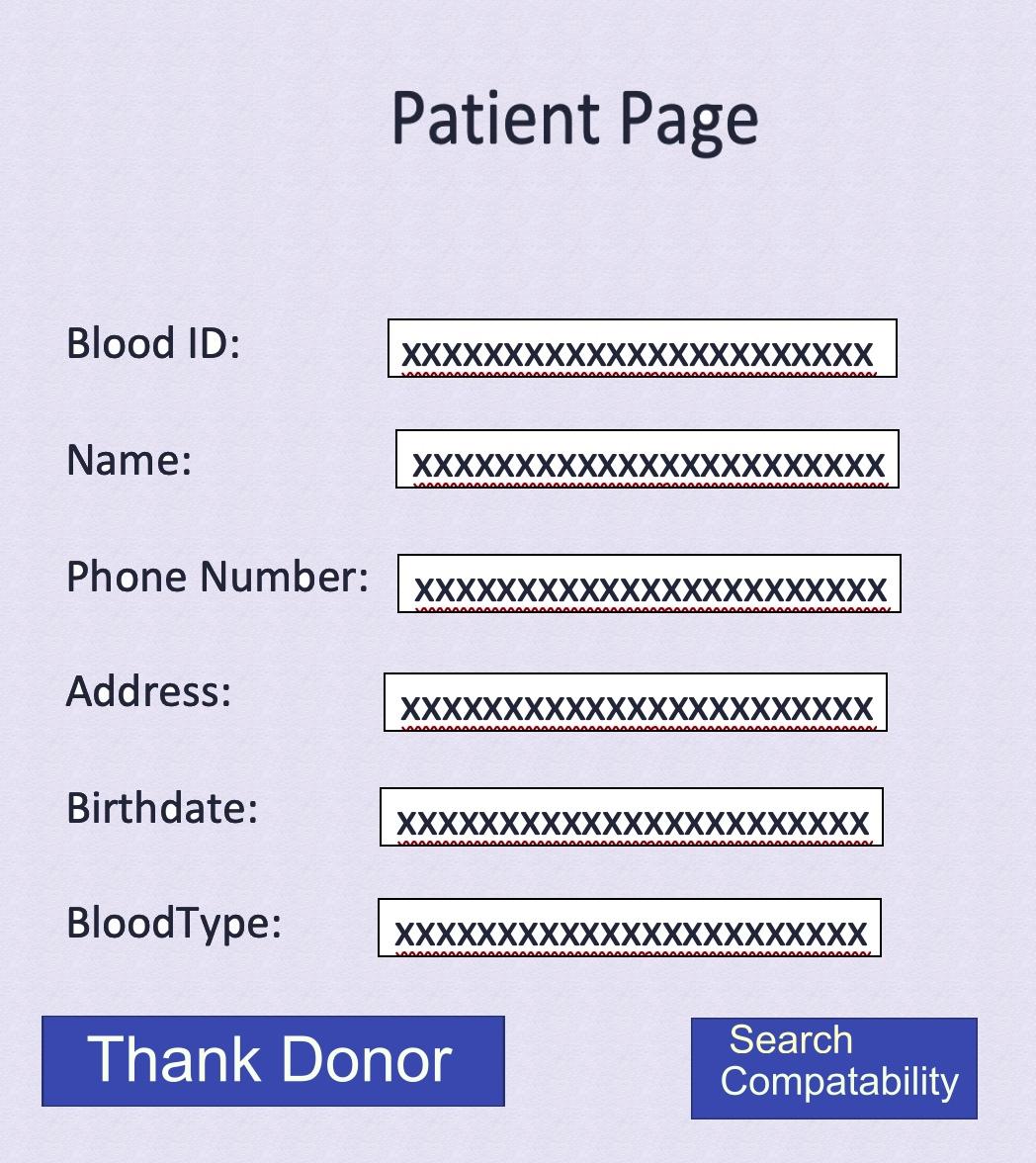
**Initial Web-interface Mock-up:**



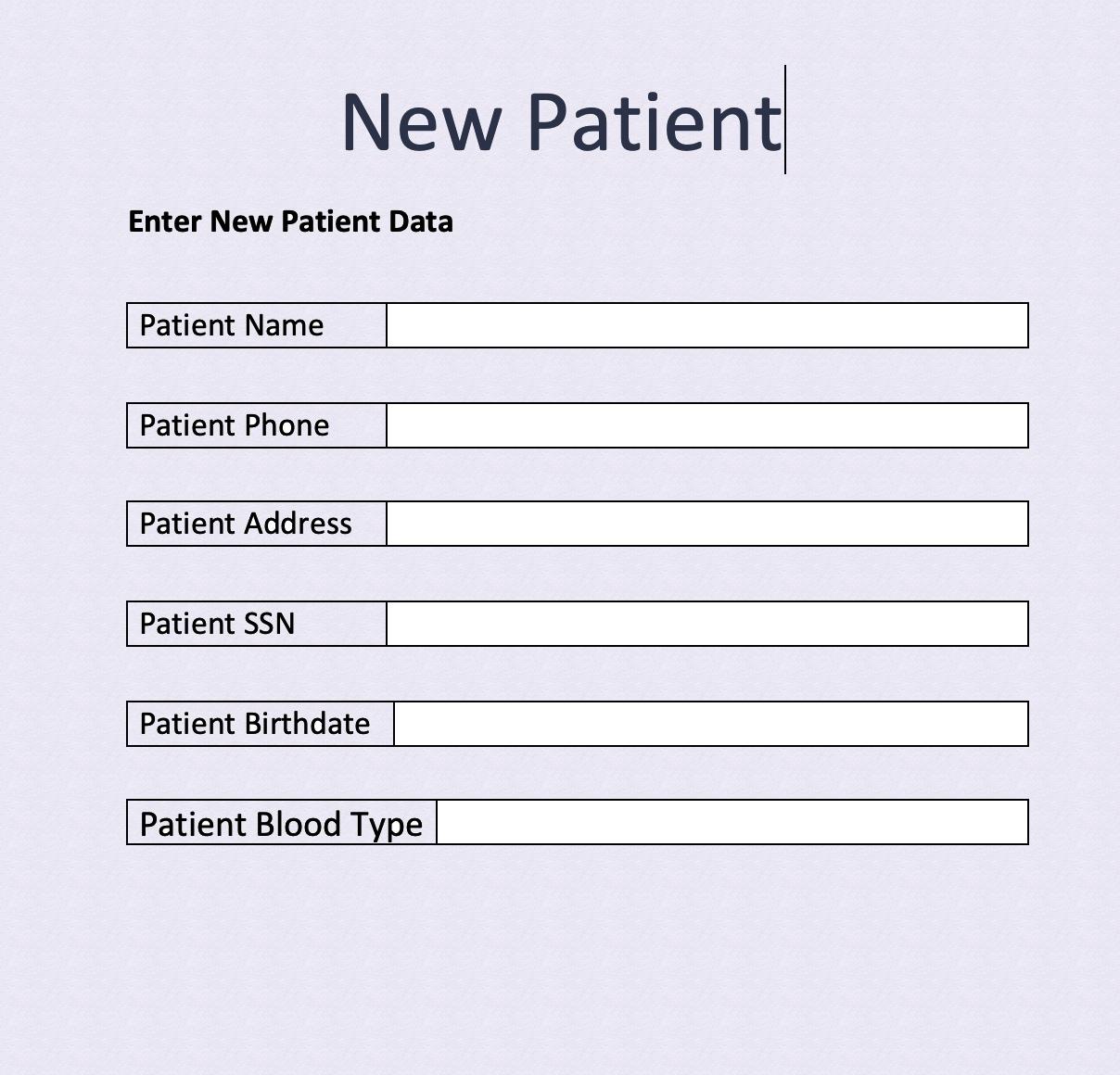
Front page asking for patient SSN, blood bank name, and address.



Donor page to give/show donor information.



Patient page for patient information, thank you option, and compatibility search.

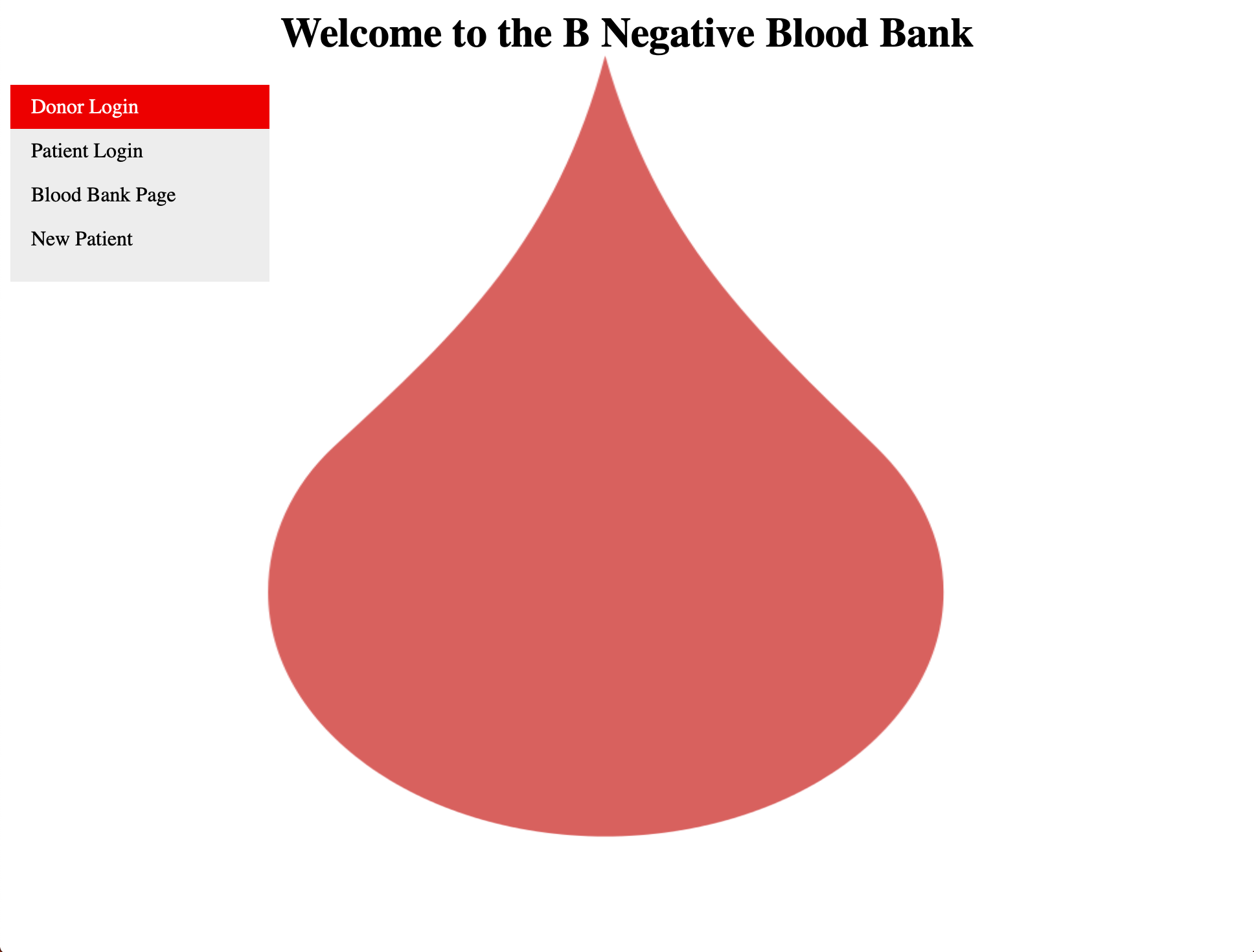


NEW patient page form for input of info : Name,Phone,address,ssn,birthdate,blood type.

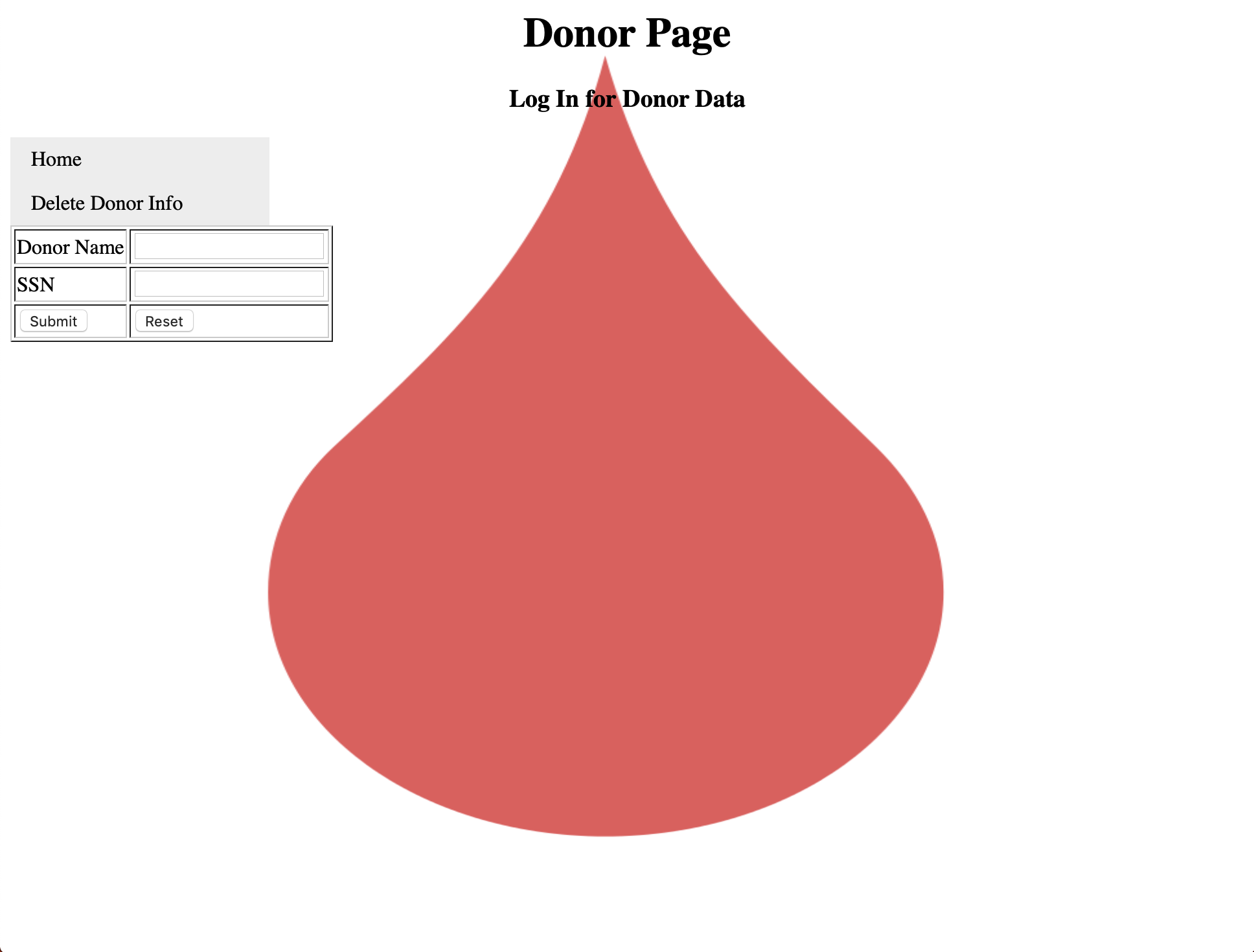
**Redesigned Application features:**

For the actual implementation of the website we used Xampp. We also changed a lot of features from the original mock-up to the final one. The initial mockup includes several ideas that we decided to get rid of for the final product including using the blood bank name, address to log in, thanking the donor on the patient page, and searching for compatibility with just a button. Blood banks In the initial mock up we included in the login for patients to also put in the blood banks name. We realized this is impossible since the donors are the only ones who donated blood so a regular patient would not know the name of the blood bank. Therefore we changed it in the actual website to have the patient put in their name as well as their SSN to log in. The same applies to donors. The original mock-up did not include a home page which we decided to add. The home page allows the user to get to patient login, donor login, blood bank page, and new patient pages. We also added that on every page on the website allows you to go back to the home page. We added submit and reset buttons to the login pages for donor and patient so they can submit or retype in their information if they mistyped it. Logging in will display the donor/ patient’s information which includes ID, name, phone number, address, ssn, birthdate and blood type. Donor and patient pages have the choice to delete their information from the database. It will automatically delete it once the delete information button is clicked after the donor/patient logins. The patient page has a choice to find a donor match. Clicking this will take you to a page that allows the user to put in their blood type. Once this is done it will query the database and display the name of a donor who does not have diseased blood and matches the blood type put in by the user. This page also has a submit and reset button like the donor/patient login. The Blood Bank Page displays information of the blood bank which includes the name of the blood bank, address, phone number and the donor’s names. This page has the option to update the blood bank’s phone number. By clicking this it will take you to a new page where you can type in the name of the blood bank and the new phone number and it will automatically update the database. This page also has a submit and reset button. The final thing you can access from the home page is New Patient. This is where a new patient can fill in their new information. Once it is filled in and the submit button is clicked it will automatically add the tuple to the *Patient\_Data* table

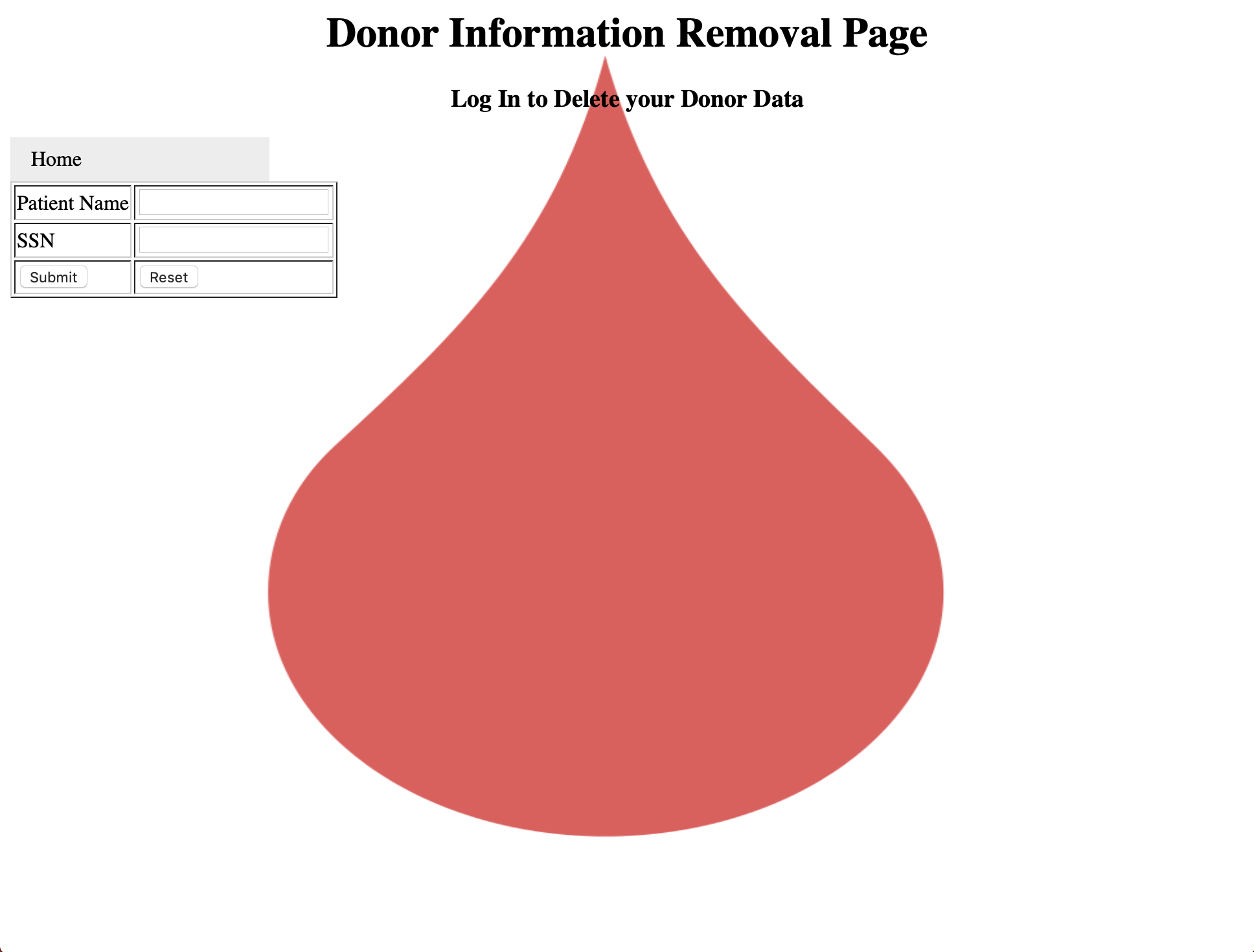
**Redesigned GUI:**

****

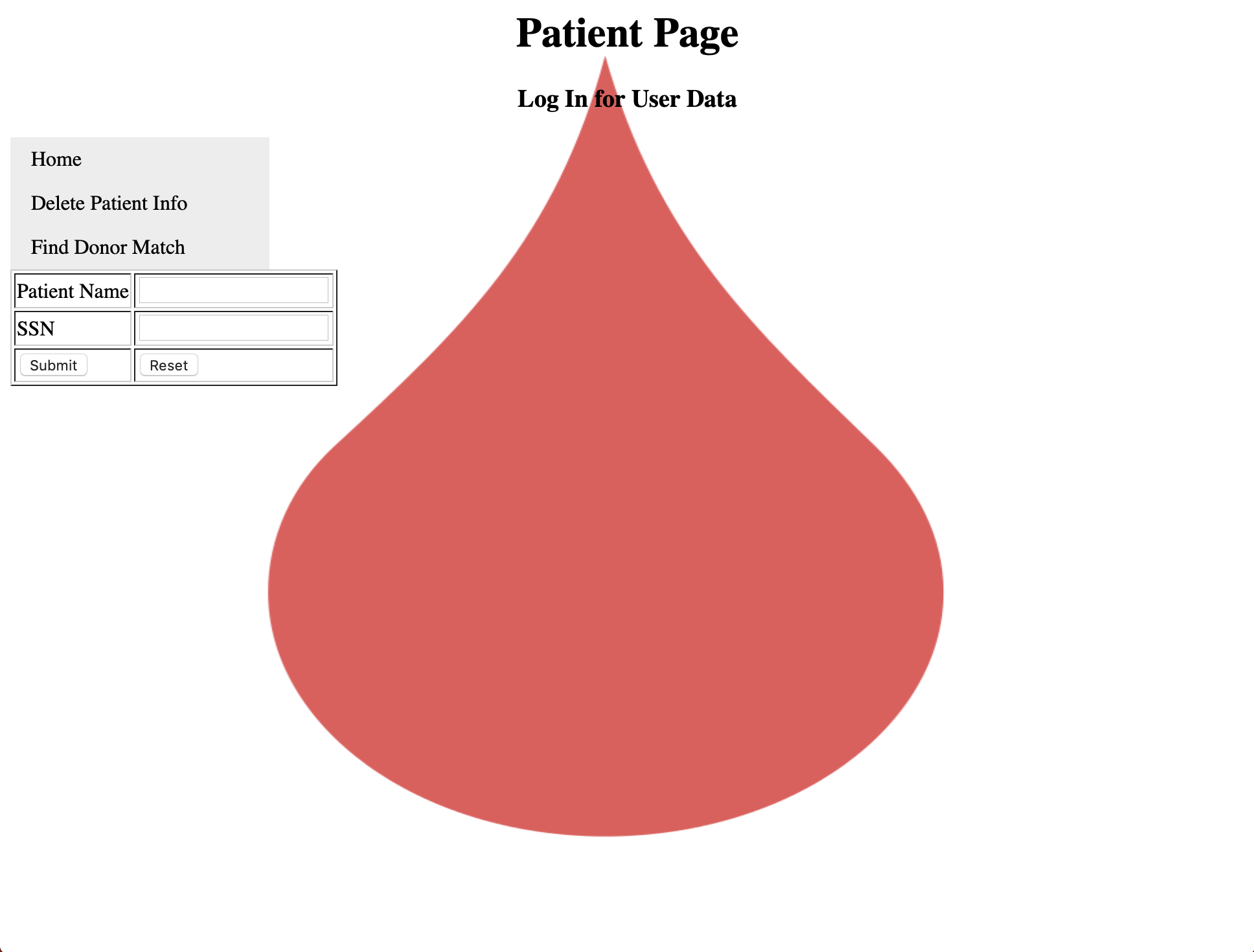
This is the first page the user sees, otherwise known as the homepage. The user has the choice to click donor login, patient login, blood bank page and new patient.

****

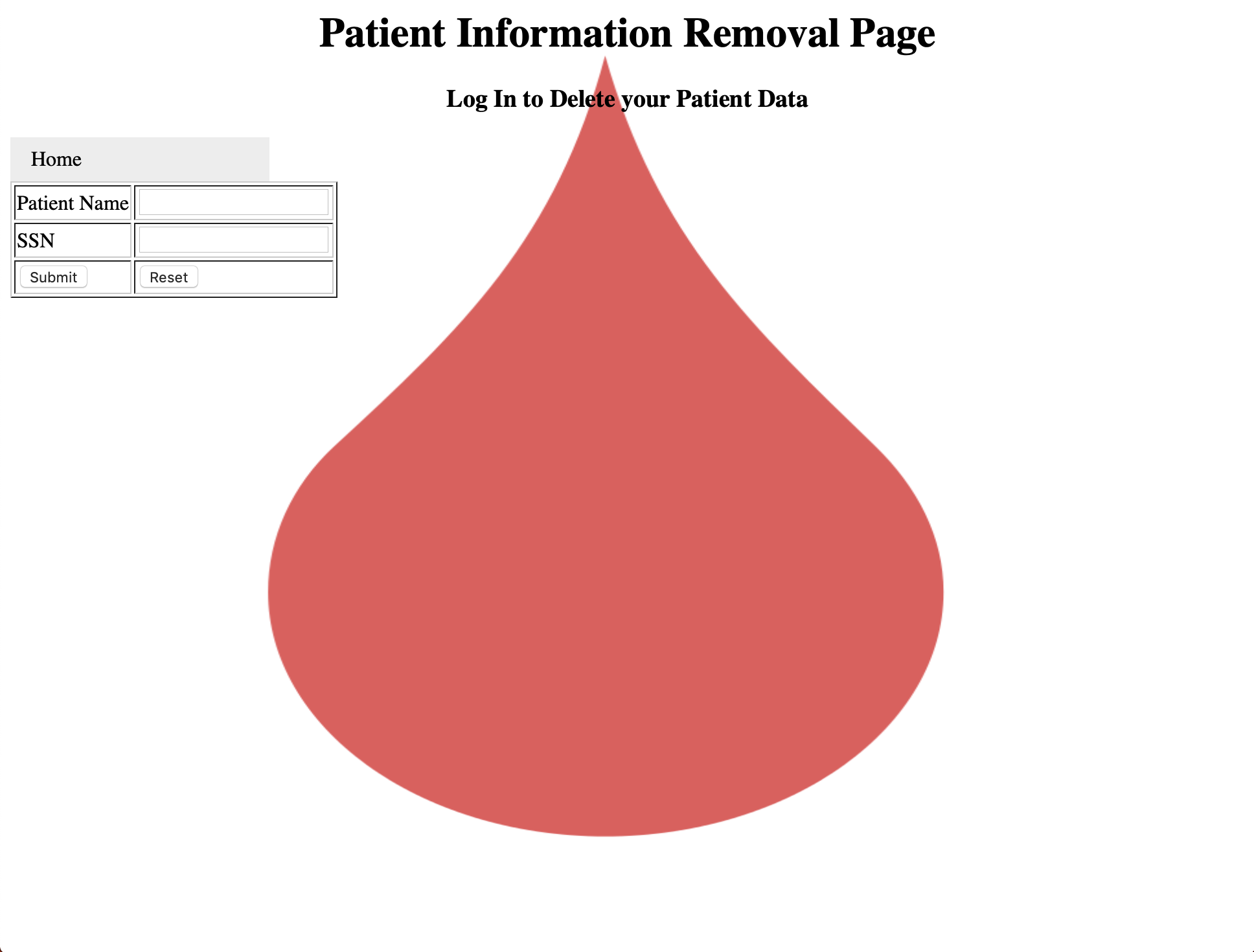
When the user clicks donor login they see this page where they must fill in their name and SSN then click submit to log in or they may hit reset if they type in their information wrong. They also have the choice to go back to the home page or delete donor info page.

****

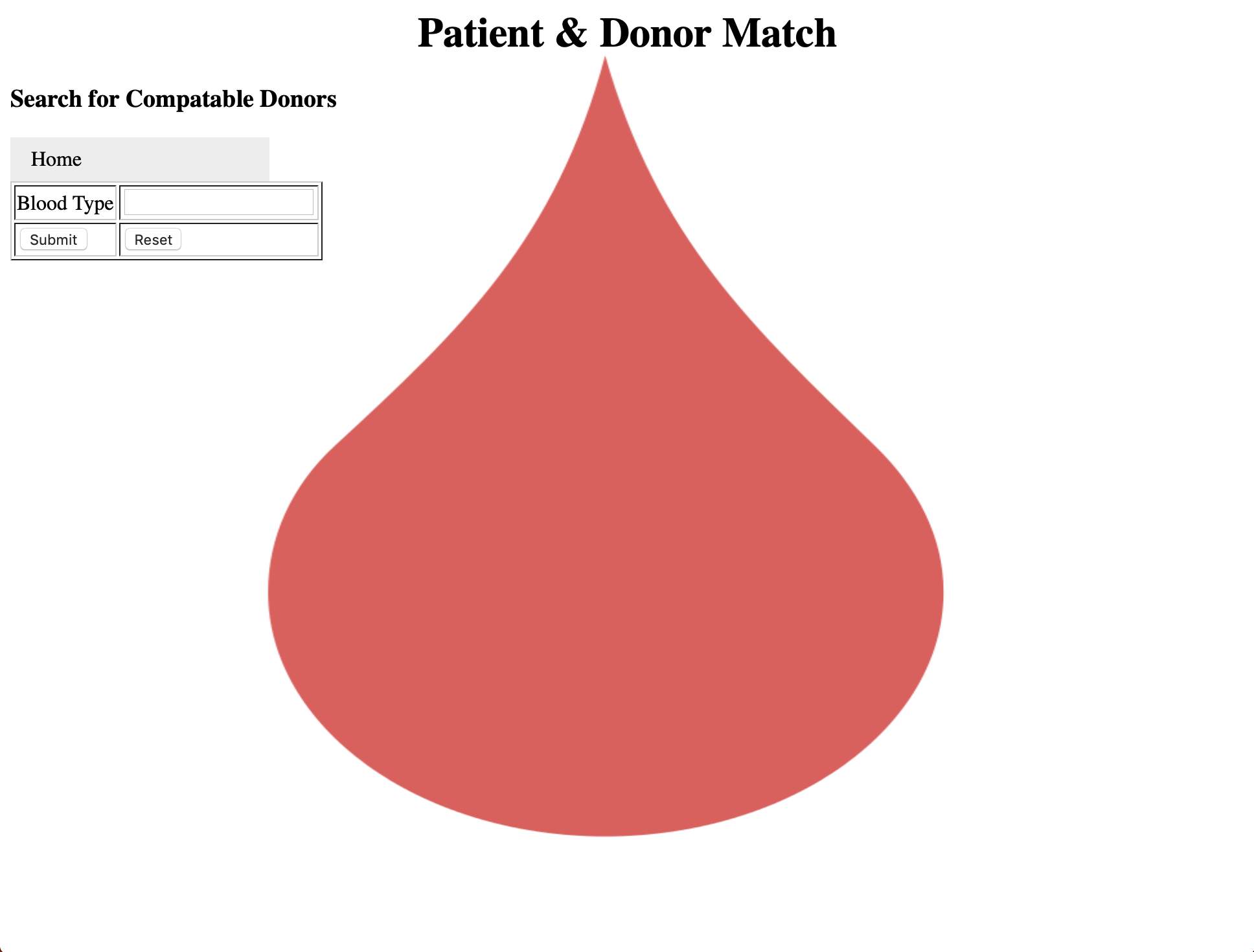
When the user clicks delete donor info this is what appears. They must log in if they want to delete their information. They can also go back to the home page on this page.

****

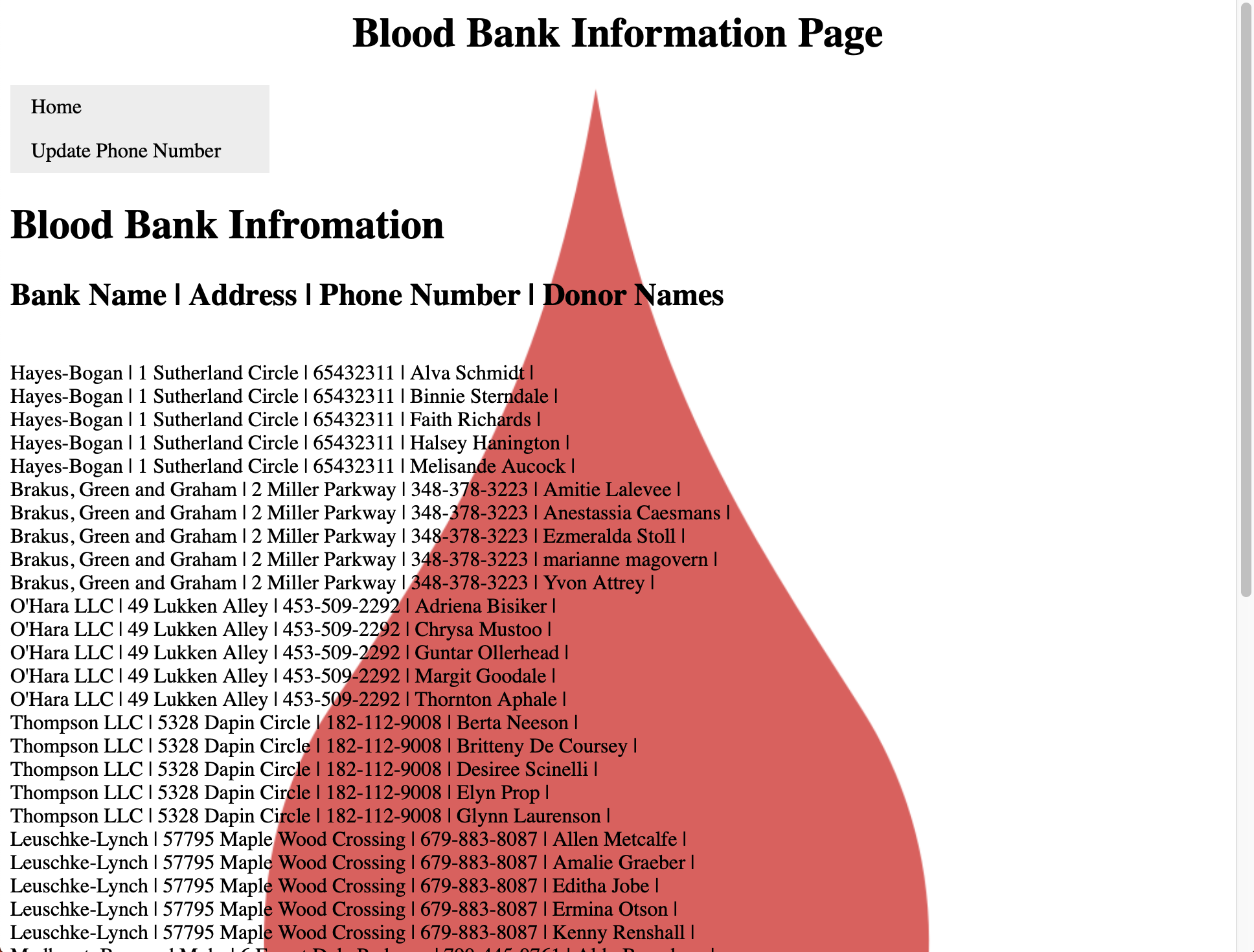
From the home page if the user clicks patient login it takes them to this page. They must fill in their name and SSN then hit submit to login. If the user types something in wrong they may click reset to try again. From this page the user can also go back to the home page, click delete patient info or find a donor match. They can also go back to the home page on this page.

****

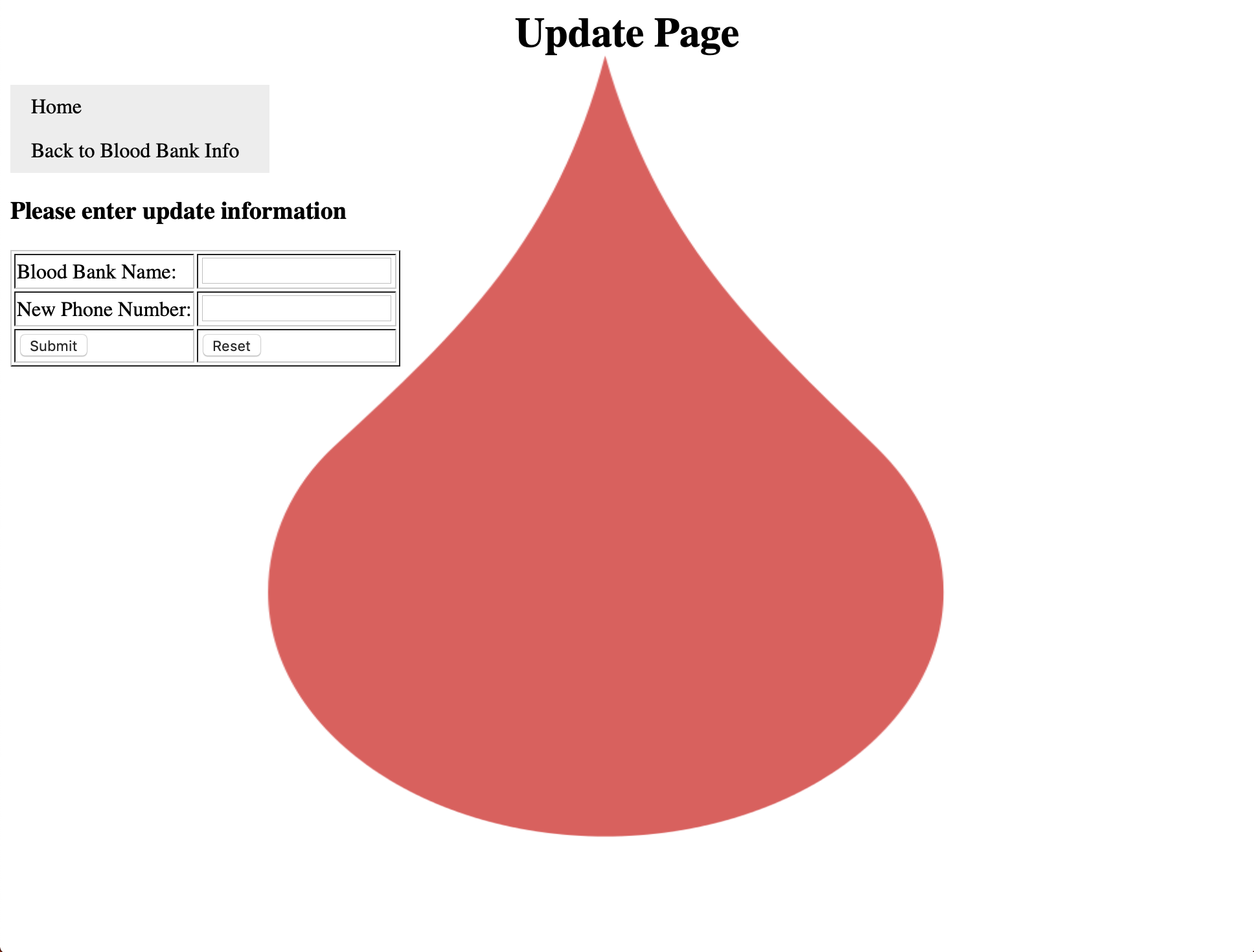
From the patient login page if the user clicks delete patient info they are taken to this page. If a user would like to delete their information they must log in first. They can also go back to the home page on this page.

****

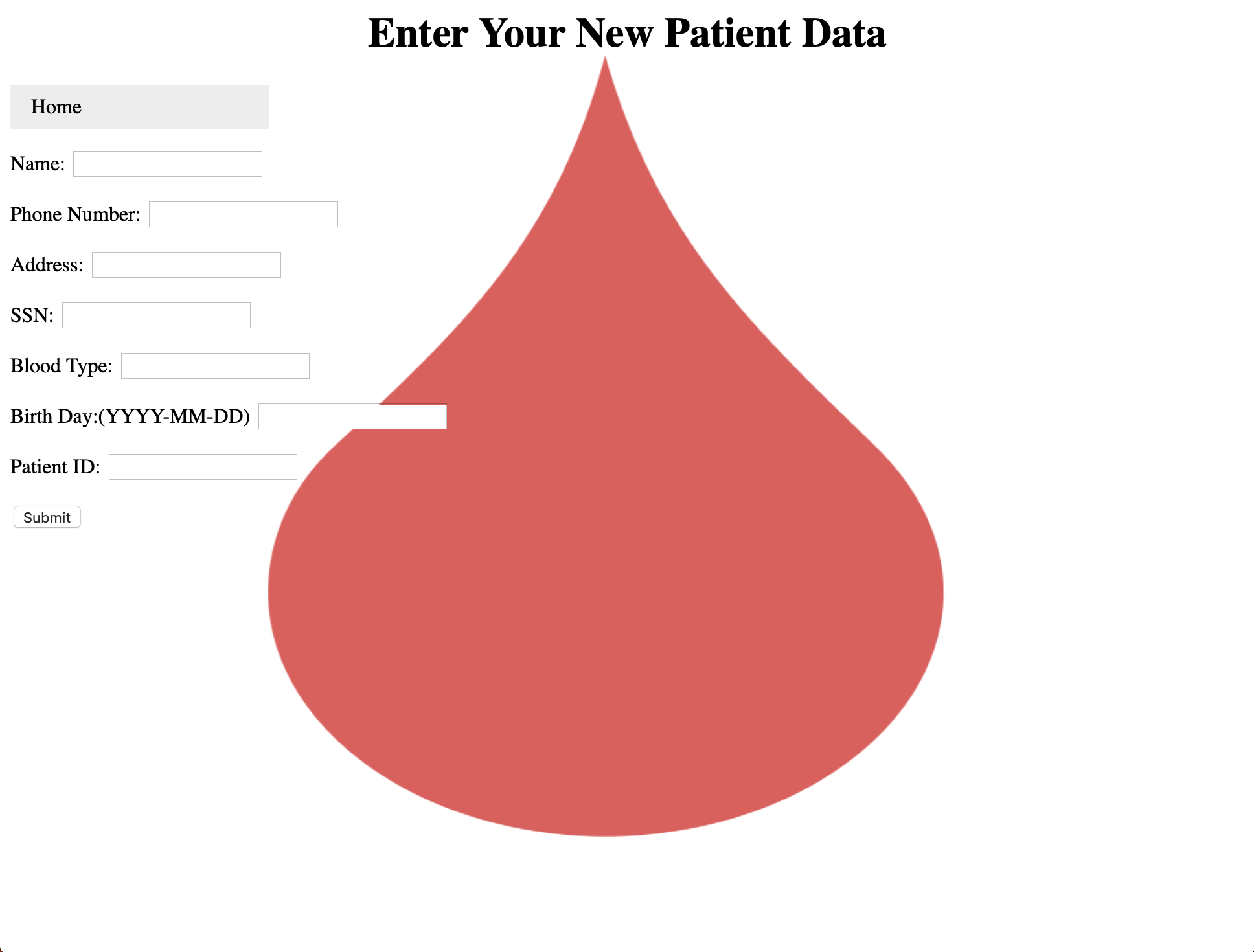
From the patient login page if the user clicks find donor match it takes them to this page. Where they can type in their blood type then the website will fetch the user a match for a blood transfusion. They can also go back to the home page on this page.

****

From the homepage if the user selects blood bank information it will display the blood bank’s name, address, phone number and the names of the donors who have donated blood at this particular bank. The user can go to the update phone number page. They can also go back to the home page on this page.

****

If the user clicks update phone number from the blood bank information page it will take the user to this page. The user can update the blood bank's phone number by typing in the blood bank’s name and the new phone number. They can also go back to the home page on this page.



If the user clicks a new patient from the homepage this page is displayed. Here the user may type in their information and it will be added to the database.They can also go back to the home page on this page.

**Executive Summary:**The purpose of this project was to create a database that was useful for patients, hospitals, donors and blood banks alike.The web-application features many uses such as: displaying donor, patient, and blood bank information as well as searching for matches for a blood transfusion via queries. The application also boasts the ability for blood banks to update their information, patients to add themselves, and allow for patients/donors to delete their own information. Finally, this web application also encourages and makes it easy for direct communication between blood drive companies, hospitals, donors and patients. This project uses MySQL Workbench for the database, as well as PHP and HTML for the web application. It is performed locally but has the potential to use a massive server where many users can use it simultaneously. It is not the most secure system but it does have a password like system using SSN. It is fairly reliable when used properly and once again has the potential to be checked on and updated daily. Overall this project has fantastic potential to be used by mainstream companies like the red cross and hospitals all over the nation.

Cody Hutton:

I, Cody Hutton, learned quite a lot through this project. Although the entire course was over database systems and queries, this project definitely played a major role. Through creating a whole database and the web application I was able to better understand the important role of understanding the inner workings of the system. It’s important to not only know the coding aspect of sql, but also the fundamental and inner logical connections of the tables,tuples, and entities.

Isabelle Harbuck

I, Isabelle Harbuck, got great experience through this project. Since moving back home I have not had the easiest access to a reliable computer so I focused more on writing up the report as well as the presentation. It helped to learn how to be able to write up a similar report when I finally get to work in the field. I learned how to write detailed descriptions of relationships, constraints, data tables, ER diagrams, functional reliability, etc. it is one thing to know how to create a database but it is a whole other when you are able to understand the subject so well you can write it out and explain it to someone else. It was a great experience and I learned a lot.

Whitson Sutherland

I, Whitson Sutherland, got a lot of the project. I created the web applet and its subsequent connection to the database through php. Having no previous experience with html or php, I learned quite a lot along the way when completing the code for this project. I learned how to use XAMPP to connect to php admin, and then how to use that connection in an html website application. This was both interesting and important to me as a programmer as it allowed me to learn and apply sql to a real world application. I also enjoyed learning how to create a website in general, as I enjoyed the benefit of being able to instantly see the results when changing html and php code.