ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

Автоматизированная система построения генеалогического дерева

Выпускная квалификационная работа
обучающегося по направлению подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем
очной формы обучения, группы 07001402
Рязанцева Егора Васильевича

Научный руководитель
к.т.н доцент Бурданова Е.В.

БЕЛГОРОД 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В наш стремительно меняющийся век необходимо не только познавать новое но и сохранять старое всегда которое всегда сможет пригодиться в нашей жизни. Одним из таких примеров может считаться генеалогия то есть наука о родословной человека , наука о его семье и дальних родственниках как то дедушках бабушках, прадедушках прабабушках и так далее. Это нужно человеку для осознания того какой путь он прошел, из чего он вышел и к чему пришел, его экзистенциальное место в этом мире.

Целью данной ВКР является создание автоматизированной системы построения генеалогического дерева. Генеалогическое дерево это записанная на бумаге и определенны способом обработанная информация о родословной человека. Оно представляет собой связанное линиями элементы . Элементы в разных версиях генеалогического дерева могут выглядеть по разному: это могут быть круги, прямоугольники. Каждый элемент может быть связан с другими элементами(людьми) тремя видами связей: с супругом/супругой ,детьми и родителями. Все кроме родителей может быть более одного. Родитель каждого пола есть у каждого человека только единственном экземпляре. Связи выглядят как линии соединяющие элементы дерева. Элементы дерева не должны располагаться на одном уровне, а должны располагаться по уровням, по поколениям то есть люди относительно составителя находящиеся с его родителями в отношениях брат, двоюродный брат и т.п. должны находиться на одном уровне.

Задачей ВКР является автоматизированная система генеалогического дерева. Основными задачами данной системы должны быть возможность добавления информации в генеалогическое дерево изменение этой информации, установление связей каждого отдельного элемента системы. Также должен быть реализован поиск по генеалогическому дереву. Хранение информации о пользователях должно быть реализовано на сервере с доступом посредством интернета. Доступ к информации должен быть реализован при

помощи авторизации и доступа каждого отдельного пользователя к информации который он сам и ввел

Данная ВКР состоит из трех частей: постановка задачи, проектирование и программная реализация информация системы , испытания.

Первая часть предназначена для того что бы понять в какой области будет выполняться ВКР, понять специфику данной области, выявить закономерности которые будут влиять на выполнение ВКР, определить направление работы во время выполнения, усвоить основные понятия и представить что необходимо реализовать в рамках работы а что не следует по причине нецелесообразности.

Вторая посвящена описанию инструментария при помощи которого производилась работа. Описывается основные особенности данных программных сред. Также описывается структура базы данных: поля, типы полей, их размер, связи с другими таблицами.

В третьей части производиться разрабатывается программа и методика испытаний, список действий необходимый для полного проведения испытаний а также проведены соответствующие ей испытания.

В заключении подводится итог проведенной работе.

В результате выполнения поставленной задачи будет разработано веб-приложение, способное выполнять заполнение, построение и работу с моделью генеалогического дерева.

Дипломная работа состоит из 51 страницу, 21 рисунков и приложения включающего страниц.

Глава 1. ПОСТАНОВКА ЗАДАЧИ

1.1 Анализ предметной области

Формально исследование происхождения человека или семьи от своих предков ныне известна как генеалогия. Построение генеалогического древа для своей семьи – кропотливая работа, которая требует тщательной проверки фактов, умения искать и анализировать обнаруженные источники. Генеалогическое дерево является схематическим отображением связей семьи или нескольких семей. В основании(в корне) находятся более старшие представители семьи, а в конце(ветвях) младшие. Помимо основной цели отображать связи человека или семьи. В центре располагается ветвь старшей в роду семьи. Следует отметить, что согласно традиции русской генеалогии прямым считается родство исключительно по мужской линии; эта норма закрепилась во времена дворянского сословия, которое по правилам не могло наследоваться по материнской линии, то есть предки и потомки по линии матери не пребывают в прямородственных отношениях (она является последним и единственным по своей линии прямым потомком), однако в эпоху «матриархата» потомки по материнской линии пребывали в прямом родстве. «Род пресёкся» - это означает что у рода отсутствует прямые потомки мужского пола.

Виды генеалогического древа

Есть несколько методов составления родословного древа:

1) Восходящее. Здесь цепочка строится по направлению от потомка к предкам. В качестве начального элемента выступает составитель схемы. Способ удобен тем, кто только начал изучение своей семьи. Составитель располагает информацией в основном о своих ближайших родственниках:

родителях, дедушках, бабушках и т. д. – и постепенно углубляется в прошлое.

2) Нисходящее. В этом случае цепочка имеет противоположное направление. В качестве начала выступает один предок (или супруги). Для такого построения нужно обладать достаточно обширными сведениями о своих родственниках.

Составляя родословное дерево нужно учитывать линии наследования. Они бывают двух видов:

1) Прямая ветвь. Цепочка включает, Вас, Ваших родителей, их родителей и т. д.
2) Боковая ветвь. Она учитывает Ваших братьев и племянников, братьев и сестер дедушек и бабушек, прадедушек и прабабушек и т. д.

Эти схемы – восходящую и нисходящую с прямыми и боковыми ветвями – можно составлять, как смешанную: одновременно для мужчин и женщин своего рода, – так и для отслеживания наследования только по роду отца или матери.

Генеалогическое дерево позволяет увидеть и изучить свою родословную а также визуально представить близкородственность как отдельных членов семьи так и семей находящихся с составителем в родственных отношениях.

При составлении генеалогического дерева есть у каждого элемента дерева(человека) присутствует две родственных связи с родителями и от одной до условной бесконечности супругов. На практике имеет место количество до десяти, учитывая нынешнего. И наконец у человека в генеалогическом дереве могут быть дети. Их также редко может быть больше десяти.

Таким образом в рамках генеалогического дерева есть широкий круг задач подлежащих автоматизации и сопровождении.

Исходя из вышеперечисленного можно выделить несколько задач для автоматизации:

1) Построение сайта для удобного пользования, ввода, изменения и отображения информации пользователя. Информация будет представляться в виде дерева, более старшие родственники которого будут располагаться вверху а более младшие внизу для удобства восприятия. В каждой ячейке которая будет обозначать человека будет расположено сведения о нем как то: имя, фамилия, отчество, пол, возраст. Каждый человек будет соединяться с двумя и более родственниками;
2) Возможность пользователем добавлять связи любого человека в генеалогическом дереве. У каждого человека есть три типа связей: родители - их может быть только два, мать и отец, супруг/ супруга – до десяти, и дети которых тоже может быть до десяти штук;
3) Возможность авторизации сохранения введенных значений. Довольно трудно каждый раз вводить данные дерева особенно если оно довольно большое поэтому необходима авторизация для сохранения конкретных данных которые ввел пользователь;
4) Возможность поиска по имени, фамилии. При больших разветвленных деревьев с большим количеством связей и людей в дереве трудно уследить за всеми людьми и запомнить всю информацию. Однако с помощью поиска по имени и/или фамилии можно будет быстро найти соответствующего родственника;
5) Возможность установления названия родственной связи между двумя людьми в генеалогическом дереве. То есть возможность в двух полях добавить сначала одного человека, затем другого и выяснить как называется родственная связь между ними, кем они друг другу приходятся. Например связь между людьми будет называться «племянник отца жены брата моей бабушки по отцовской линии» ;

1.2 Обзор и анализ существующих систем

Система 1. MyHeritage

В первую очередь рассмотрим систему сайта MyHeritage расположенного по интернет адресу: https://www.myheritage.com. На рисунке 1.1 главная страница сайта. Дизайн страницы крайне современен выглядит приятно.

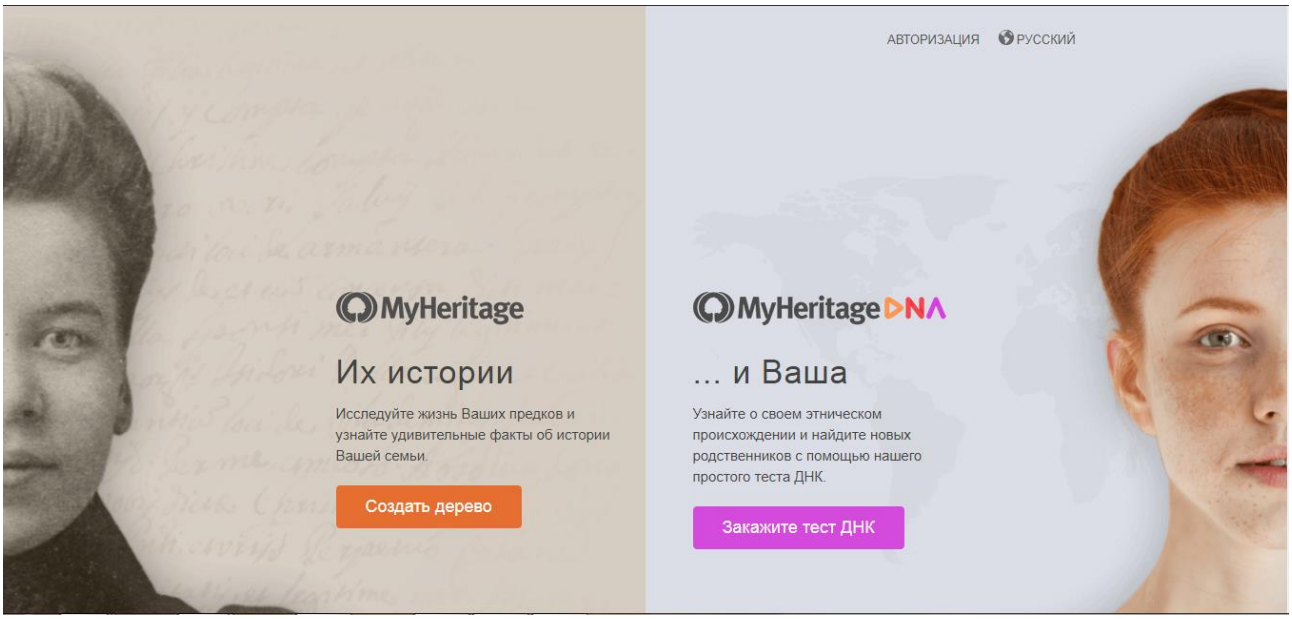

Рис. 1.1. Заглавная страница сайта

Посмотри на то, как сайт реализует приведенное выше и заявленное нами как требующее автоматизации:

1) Построение сайта для удобного пользования, ввода, изменения и отображения информации пользователя. На рисунке 1.2 изображено генеалогическое дерево сайта. На главной странице наравне с возможностью создать свое генеалогическое дерево нм предлагают узнать тест ДНК для того что бы определить карту ДНК и возможных родственников. То есть сайт является одновременно и сайтом для генеалогического дерева и сайтом для тестов ДНК что явно является

минусом поскольку наш сайт является исключительно специализированным на теме генеалогии. После того как пользователь введет самого себя и своих ближайших родственников таких как мать отец и родители родителей, дается возможность ввести остальных любых родственников. На отдельных «листьях» дерева кнопки для добавления связей и изменения информации человека слишком маленькие что бы по ним можно было бы попасть и ввести информацию;

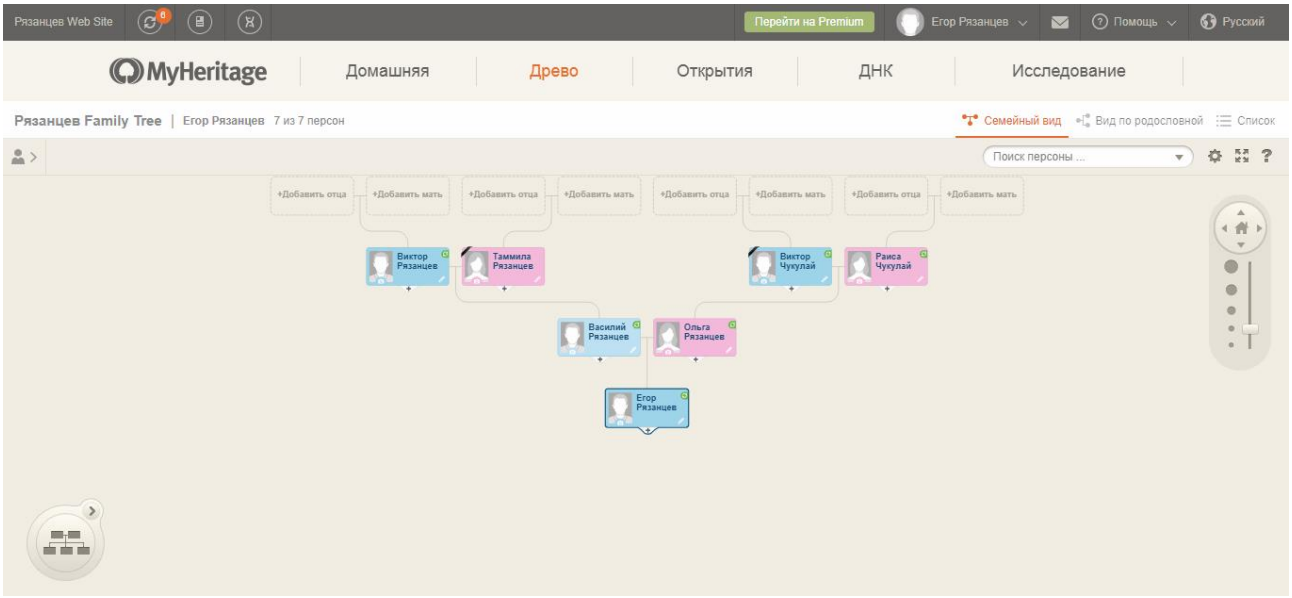

Рис. 1.2. Изображение генеалогического дерева

2) Возможность пользователем добавлять связи любого человека в генеалогическом дереве. У каждого человека есть три типа связей: родители - их может быть только два, мать и отец, супруг/ супруга – до десяти, и дети которых тоже может быть до десяти штук. На сайте MyHeritage отсутствует возможность добавления нескольких мужей или большого количества детей больше 5;
3) Возможность авторизации сохранения введенных значений. На рисунке 1.3 изображена страница авторизации и попутно создания первого элемента генеалогического дерева пользователя. Довольно трудно

каждый раз вводить данные дерева особенно если оно довольно большое поэтому необходима авторизация для сохранения конкретных данных которые ввел пользователь. На данном сайте авторизация довольно удобная однако отсутствует возможность просматривать текст пароля при введении, также есть возможность вводить любой пароль любо размера и сложности что не является правильным поскольку нерадивый пользователь может ввести слишком короткий и простой пароль или вообще ввести буквы подряд с клавиатуры чем и могут воспользоваться злоумышленники;

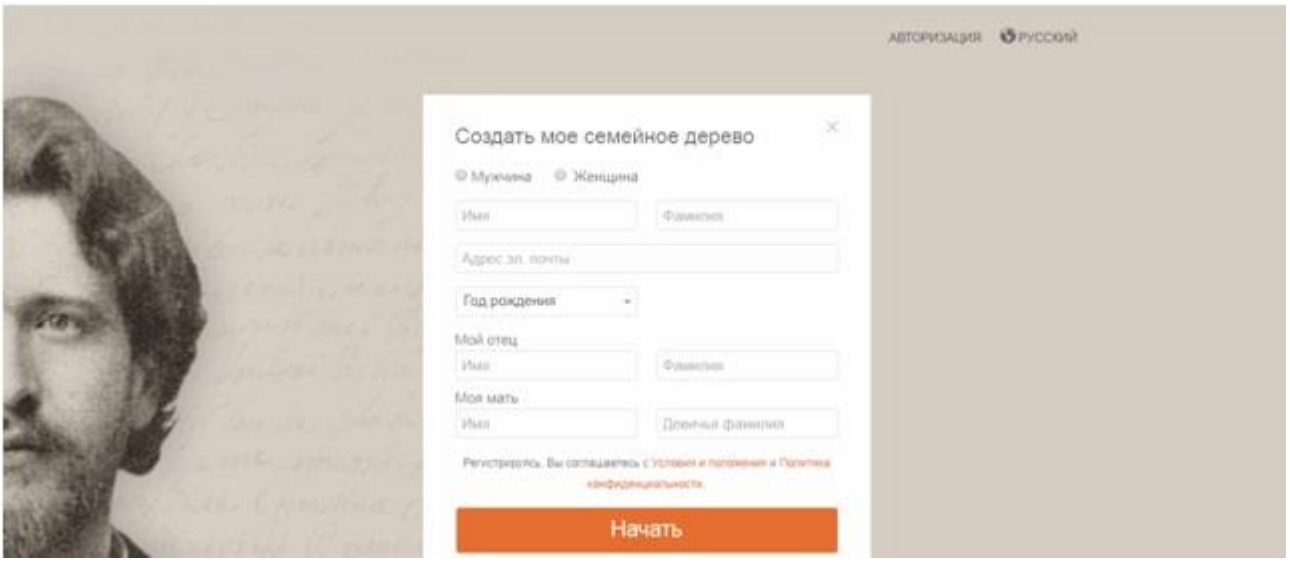

Рис. 1.3. Авторизация и создание первого элемента дерева

4) Возможность поиска по имени, фамилии. При больших разветвленных деревьев с большим количеством связей и людей в дереве трудно уследить за всеми людьми и запомнить всю информацию. Однако с помощью поиска по имени и/или фамилии можно будет быстро найти соответствующего родственника. На сайте MyHeritage присутствует поиск по имени и фамилии однако нет возможности использовать возможности расширенного поиска по различным параметрам как то

пол, примерный возраст, год рождения, отсутствие/наличие таких связей как супруг/супруга и тому подобное;

Рассмотрев данный сайт и оценив его возможности, можно подытожить, что данный сайт неплохи справляется со своими обязанностями но недостаточно для сайта главным предназначением которого является генеалогическое дерево.

Система 2**. family-tradition**

В первую очередь рассмотрим систему сайта family-tradition расположенного по интернет адресу: http://family-tradition.ru. На рисунке 1.4 главная страница сайта;

Рис. 1.4. Заглавная страница сайта

Посмотрим на то, как сайт реализует приведенное выше и заявленное нами как требующее автоматизации:

1) Построение сайта для удобного пользования, ввода, изменения и отображения информации пользователя. На рисунке 1.5 отображено генеалогическое древо полученное с помощью этого сайта. В первую очередь сайт сконструирован как место для заказа услуг по сбору и организации информации о генеалогии. Кроме того данные услуги в

отличие от нашего сайта являются платными и весьма дорогими рисунок 1.6;

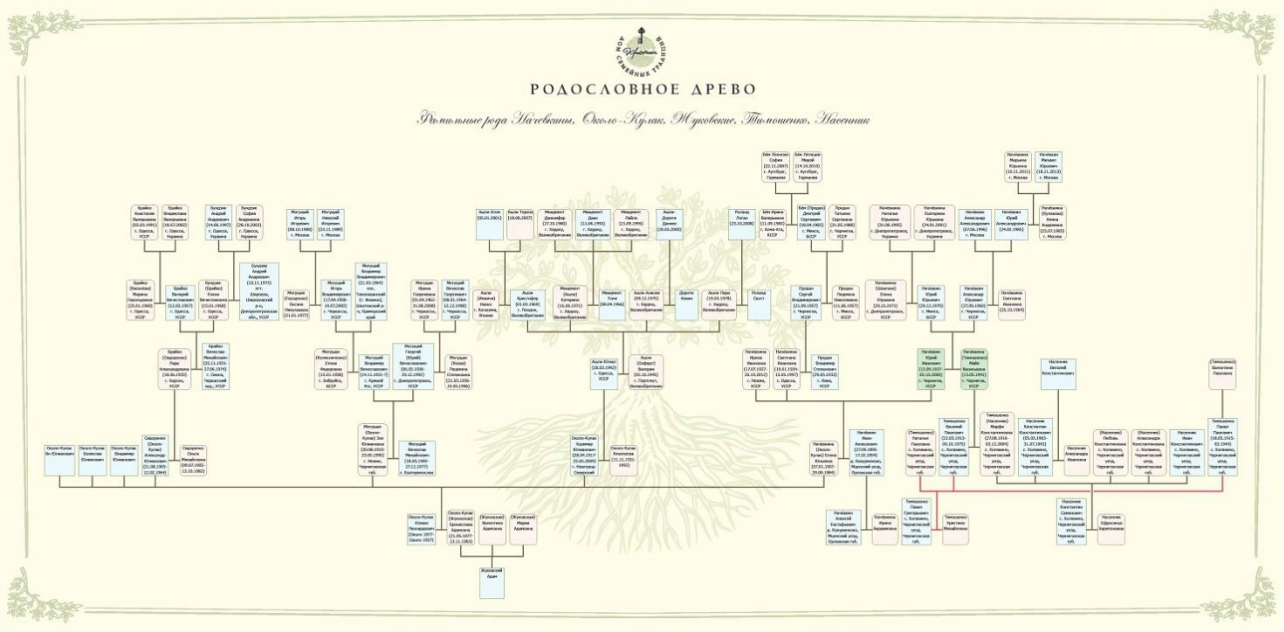

Рис. 1.5. Генеалогическое древо сайта

2) Возможность пользователем добавлять связи любого человека в генеалогическом дереве. У каждого человека есть три типа связей: родители - их может быть только два, мать и отец, супруг/ супруга – до десяти, и дети которых тоже может быть до десяти штук. На сайте family-tradition вся необходимая деятельность выполняется специальными людьми за деньги. Они же и проведут необходимую генеалогическую экспертизу на основе источников государственных документов и составят профессионально сформированное генеалогическое дерево. Однако в данном же пункте кроется большой минус поскольку данный процесс делается не вами большая вероятность что процесс затянется на неопределенный срок и будет довольно дорогим;
3) Возможность авторизации сохранения введенных значений. На рисунке 1.7 изображена страница авторизации и попутно создания первого элемента генеалогического дерева пользователя. Довольно трудно каждый раз вводить данные дерева особенно если оно довольно большое

поэтому необходима авторизация для сохранения конкретных данных которые ввел пользователь. На данном сайте авторизация довольно удобная однако она существует не для того что бы менять информацию введенную вами ранее а для оплаты по интернету заказанной вами работы и мониторинга выполнения работ. Поэтому данный момент является совсем не той же авторизацией на нашем сайте что является серьезным минусом;

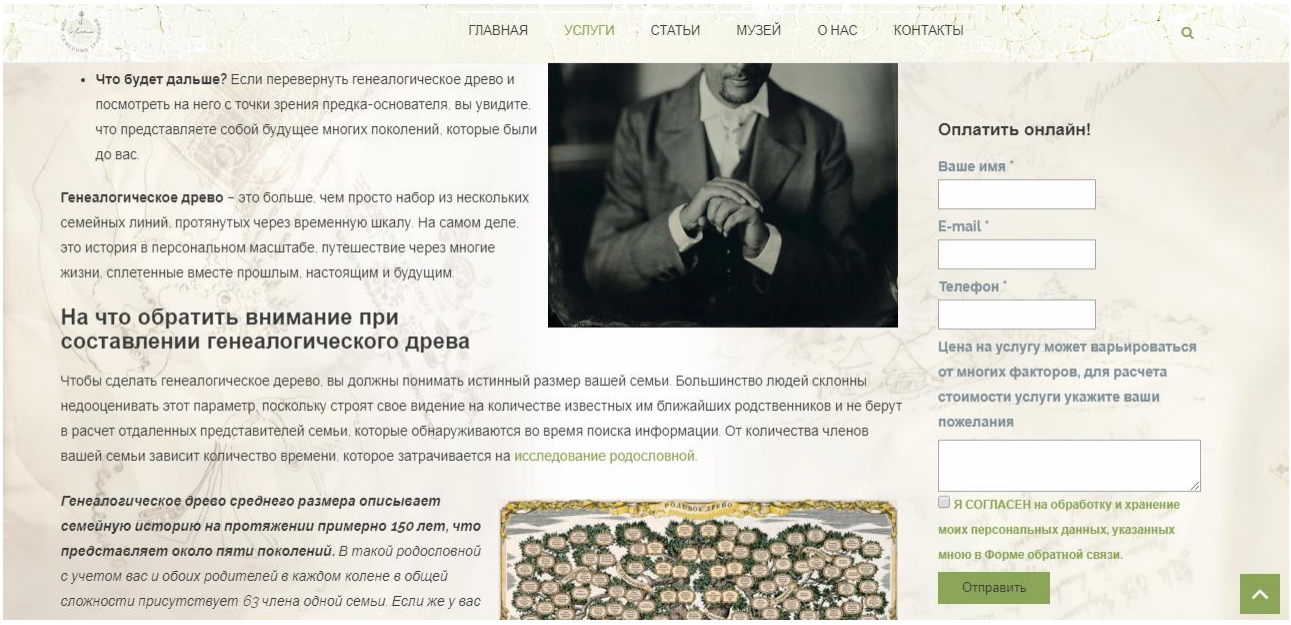

Рис. 1.7. Авторизация на сайте

4) Возможность поиска по имени, фамилии.. На сайте family-tradition присутствует поиск по имени и фамилии однако нет возможности использовать возможности расширенного поиска по различным параметрам как то пол, примерный возраст, год рождения, отсутствие/наличие таких связей как супруг/супруга и тому подобное. Возможность поиска отсутствует как факт потому что сама структура сайта не предполагает то что на сайте в доступности пользователя находиться информация введенная пользователем;

Рассмотрев данный сайт и оценив его возможности, можно подытожить, что данный сайт справляется со своими обязанностями но не в том значение в котором нужно что делает его абсолютно не приспособленным для нашего случая.

Система 3**. familyalbum**

В первую очередь рассмотрим систему сайта family-tradition расположенного по интернет адресу: https://familyalbum.me. На рисунке 1.8 главная страница сайта.

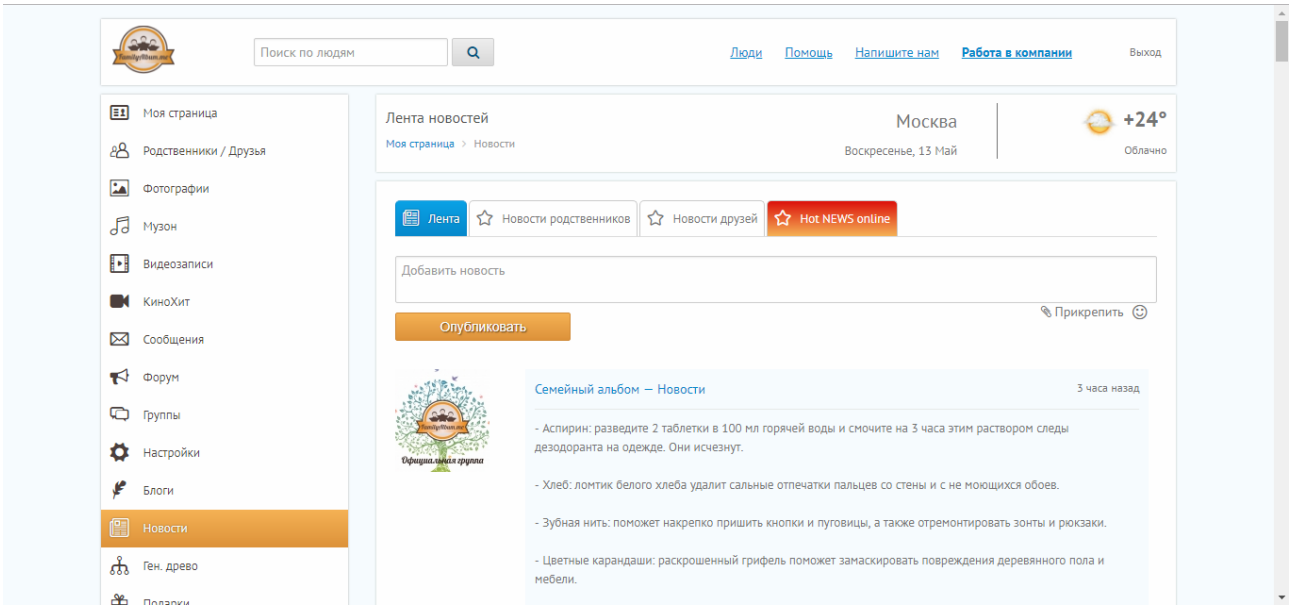

Рис. 1.8. Заглавная страница сайта

Посмотрим на то, как сайт реализует приведенное выше и заявленное нами как требующее автоматизации:

1) Построение сайта для удобного пользования, ввода, изменения и отображения информации пользователя. На рисунке 1.9 отображено генеалогическое древо полученное с помощью этого сайта. Добавление новых родственников на этом сайте происходит путем нажатия на кнопку «добавить родственника» после чего появляется форма добавления родственника одна для всех родственников. Различные варианты этой формы меняются путем переключения кнопок на форме как то супруг/супруга, ребенок, родитель. Рисунок 1.10 Данное решение

является не очень удобным и более удачным является решение вынесение подобного функционала в структуру генеалогического дерева;

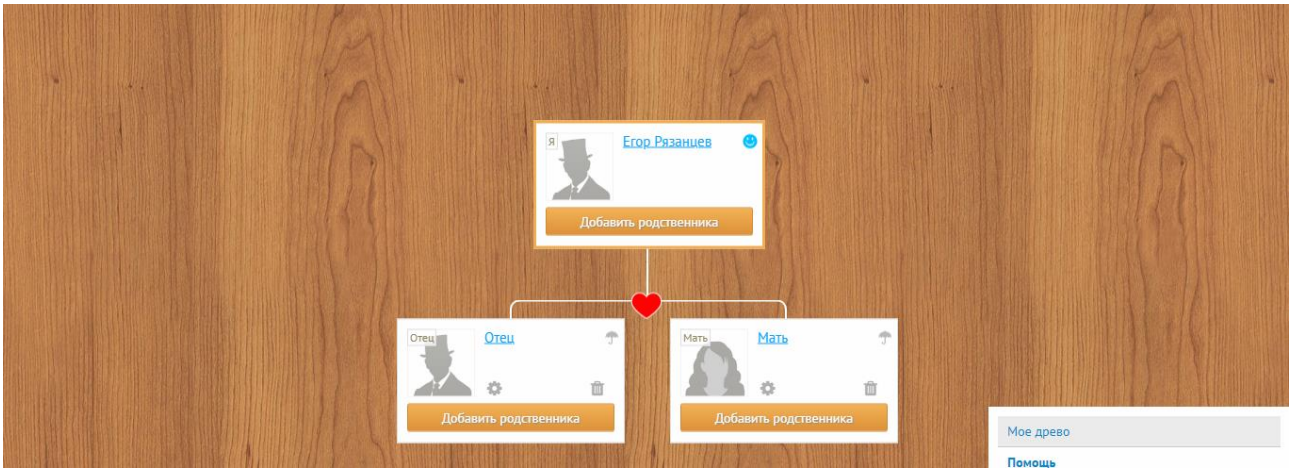

Рис. 1.9. Генеалогическое древо сайта

2) Возможность пользователем добавлять связи любого человека в генеалогическом дереве. У каждого человека есть три типа связей: родители - их может быть только два, мать и отец, супруг/ супруга – до десяти, и дети которых тоже может быть до десяти штук. На сайте familyalbum данный функционал присутствует но надо обратить внимание на то что данный аспект не предполагает добавление большого количества родственников к одному человеку что является серьезным минусом;

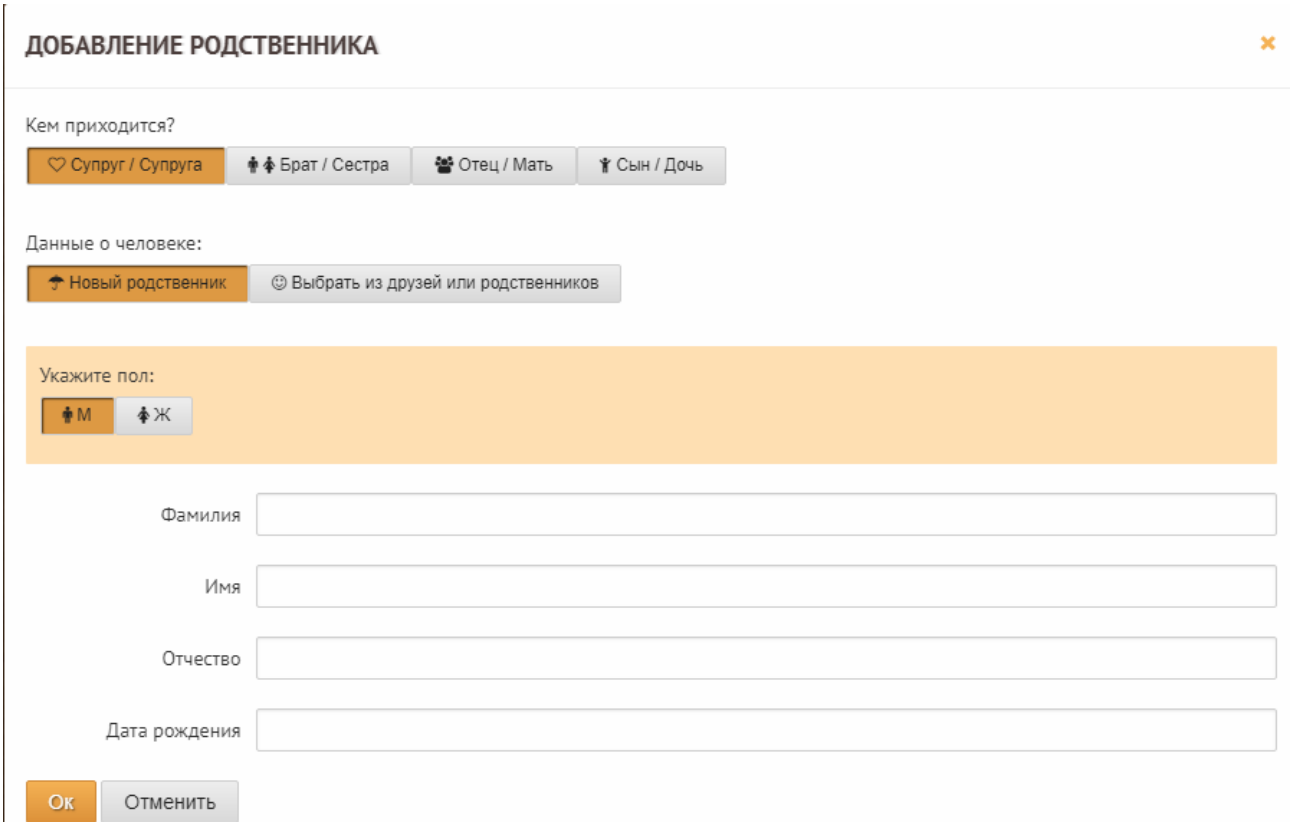

Рис. 1.10. Добавление родственника

3) Возможность авторизации сохранения введенных значений. На рисунке 1.11 изображена страница авторизации и попутно создания первого элемента генеалогического дерева пользователя.. На данном сайте авторизация довольно удобная однако необходимость введения большого количества личной информации в том числе города где живет пользователь и необходимость введения номера мобильного телефона сильно осложняет процесс;

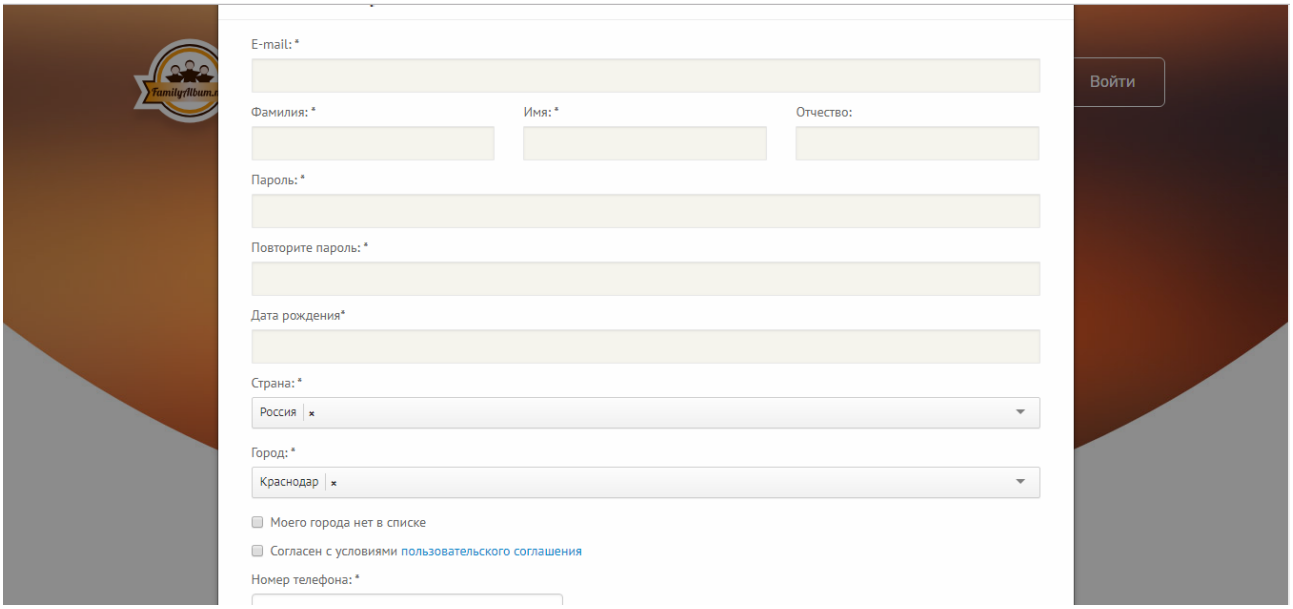

Рис. 1.11. Авторизация на сайте

4) Возможность поиска по имени, фамилии.. На сайте familyalbum Отсутствует возможность поиска родственника по имени и фамилии что являеться серьезным минусом. Вместо этого форме добавления родственника находиться возможность добавить родственника из тех что были уже добавлены пользователем. Но вряд ли это можно считать полноценным поиском;

Рассмотрев данный сайт и оценив его возможности, можно подытожить, что данный сайт не совсем справляется со своими обязанностями сайта посвященному генеалогическом дереву.

Система 4**. familyspace**

В первую очередь рассмотрим систему сайта familyspace расположенного по интернет адресу: https://www.familyspace.ru. На рисунке 1.12 главная страница сайта.

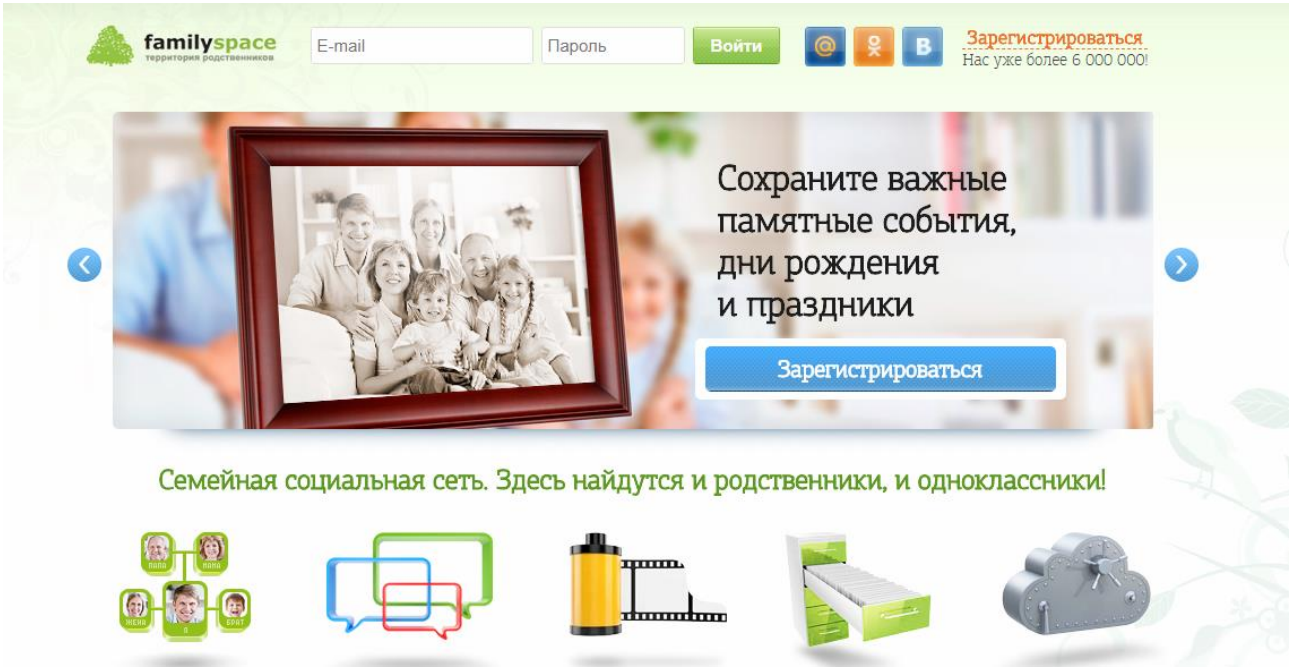

Рис. 1.12. Заглавная страница сайта

Посмотрим на то, как сайт реализует приведенное выше и заявленное нами как требующее автоматизации:

1) Построение сайта для удобного пользования, ввода, изменения и отображения информации пользователя. Добавление новых родственников на этом сайте происходит путем нажатия на кнопку «добавить родственника» после чего появляется форма добавления родственника одна для всех родственников. Различные варианты этой формы меняются путем переключения кнопок на форме как то супруг/супруга, ребенок, родитель. Рисунок 1.13 Данное решение является не очень удобным и более удачным является решение вынесение подобного функционала в структуру генеалогического дерева;

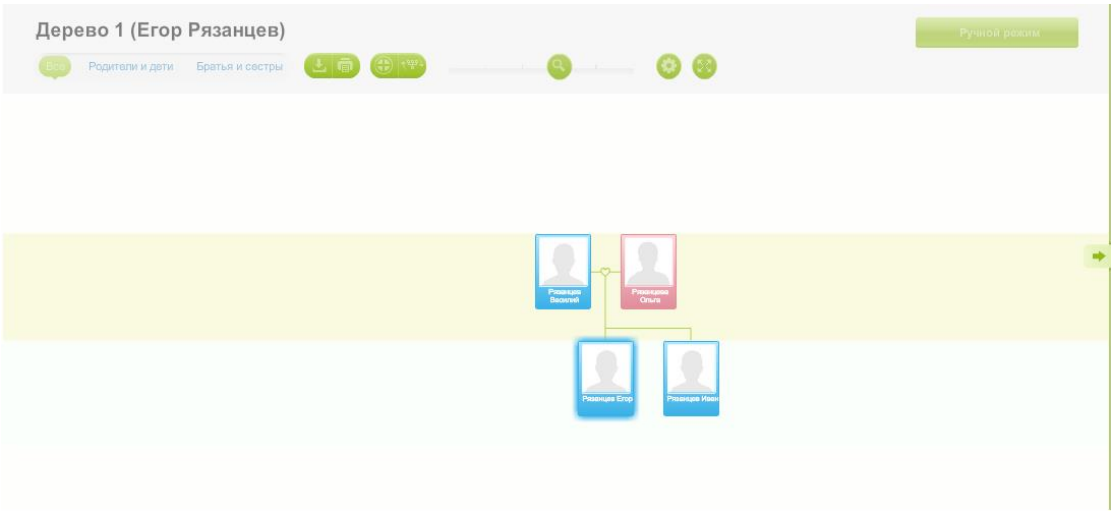

Рисунок 1.13. Генеалогическое древо сайта

2) Возможность пользователем добавлять связи любого человека в генеалогическом дереве. У каждого человека есть три типа связей: родители - их может быть только два, мать и отец, супруг/ супруга – до десяти, и дети которых тоже может быть до десяти штук. На сайте familyspace данный функционал присутствует но надо обратить внимание на то что данный аспект не предполагает добавление большого количества родственников к одному человеку что является серьезным минусом;

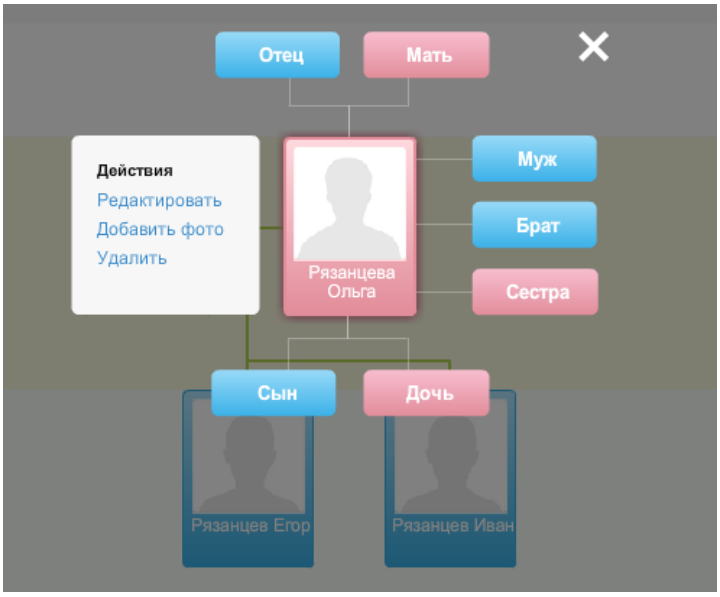

Рис. 1.14. Добавление родственника

3) Возможность авторизации сохранения введенных значений. Довольно трудно каждый раз вводить данные дерева особенно если оно довольно большое поэтому необходима авторизация для сохранения конкретных данных которые ввел пользователь. На рисунке 1.15 изображена страница авторизации и генеалогического дерева пользователя.. На данном сайте авторизация довольно удобная однако необходимость продолжительного времени ожидания авторизации сильно осложняет процесс;

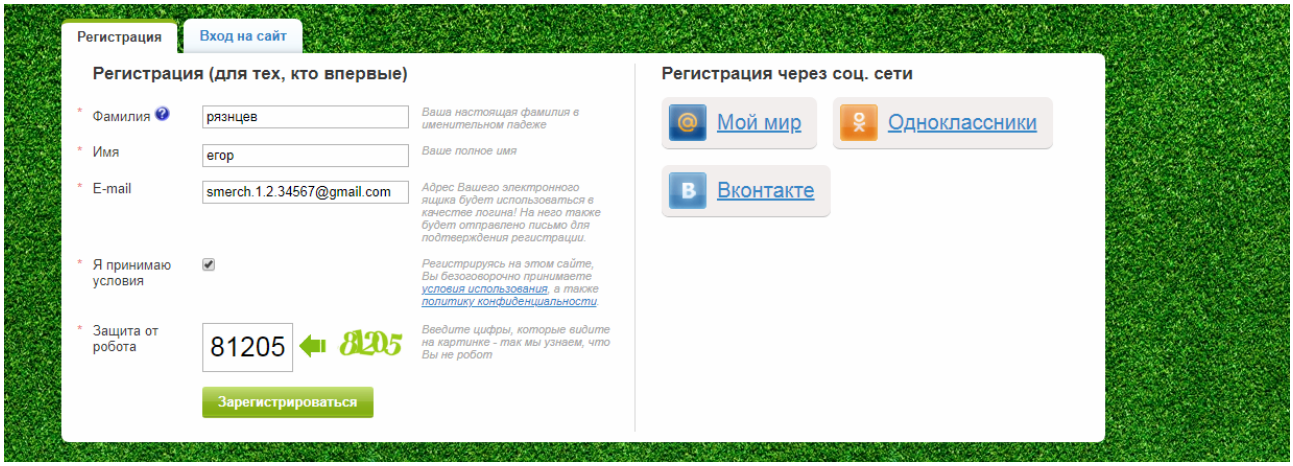

Рис. 1.3. Авторизация на сайте

4) Возможность поиска по имени, фамилии.. На сайте familyspace отсутствует возможность поиска родственника по имени и фамилии что является серьезным минусом;

Рассмотрев данный сайт и оценив его возможности, можно подытожить, что данный сайт справляется со своими обязанностями но не в том значение в котором нужно что делает его абсолютно не приспособленным для нашего случая.

1.3 Требования к автоматизированной системе

Требования к функциональным характеристикам

1) Автоматизированная система должна обеспечить выполнение следующих функций:

- Добавление родственников для каждого отдельно взятого элемента дерева;
- Возможность редактирования информации отдельно взятого человека
- Возможность поиска по родственников по фамилии и имени
- Возможность добавлении определения названия родственной вязи между двумя отдельными элементами
- Возможность авторизации и сохранения данных .

2) Требования к организации входных данных

Входными данными в программе являются необходимые данные, которые пользователи автоматизированной системы вводят в процессе авторизации – такие как:

- логин;
- пароль.

Те же входные данные которые соответствуют входным данным администратора должны вводиться соответствующим администратором на странице сайта

На странице администратор сайта обязан иметь все возможности для изменения следующих данных:

- изменение механизма редактирования;
- изменение механизма добавления;
- изменение механизма добавления связей;

В таблице людей администратор сайта должен иметь возможность ввести следующие данные:

- имя человека;

- фамилию;
- отчество;
- пол.
- возраст

В таблице возможных связей администратор сайта должен иметь возможность Изменять тип родственной связи

Общие требования к входным данным, вводимым пользователем автоматизированной системы: данные пользователя должны вводиться через специальные поля форм если это текстовые данные, либо если это файлы изображения то они перетягиваются мышью на форму. Те поля, в которые пользователь вводит информацию вручную, должны проверяться на наличие данных как на стороне клиента, так и после отправки на стороне сервера. Загруженные файлы изображений должны размещаться в специальной папке на сервере, текстовые данные должны быть отсортированы и храниться в базе данных.

3) Требования к организации выходных данных

Вывод данных программы обязан осуществляться путем вывода и преобразования информации. На странице информация должна выводиться в виде списка в следующих :

- Старшие представители семь должны идти выше чем младшие;
- Члены одного поколения должны находиться на одном уровне;
- Связи отдельных веток не должны пересекаться и «наезжать» друг на друга если только пересечения не требует схема пользователя.

4) Требования к временным характеристикам

Требования к временным характеристикам зависит от количества запросов и пропускной способности сервера. Информационная система должна выполнять работу, осуществляя как можно меньше запросов к серверу.

5) Требования к надежности

Надежная работа программы должна обеспечиваться выполнением оргтехнического комплекса мероприятий, приведенного ниже:

- организация бесперебойного и надежного питания серверного оборудования;

- исполнением всех необходимых рекомендаций Министерства труда и социального развития РФ, записанных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

- исполнением требований ГОСТ 51188-98. Защита информации;

- проверка аппаратной части на наличие вирусов

6) Требования к исходным кодам и языкам программирования

Языком программирования, реализующим исходные коды стороне сервера, должен быть язык программирования C#. Интегрированной средой разработки автоматизированной системы должна быть использована среда PHP. Для создания базы данных и работы с ней использована программа Mysql.

7) Требования к программным средствам, используемым программой

С клиентской стороны допускается наличие любой операционной системы и современного браузера

Глава 2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

2.1 Инфологическое проектирование

Концептуальное (инфологическое) проектирование — Цель инфологического этапа проектирования состоит в получении семантических (концептуальных) моделей, отражающих предметную область и информационные потребности пользователей. В качестве инструмента для построения семантических моделей данных на этапе инфологического проектирования является неформальная модель "Сущность-Связь" (ER-модель - Entity-Relationship). Моделирование предметной области базируется на использовании графических диаграмм, включающих небольшое число разнородных компонентов.

Основными понятиями ER-модели являются сущность, связь и атрибут.

Сущность (объект) - это реальный или представляемый объект предметной области, информация о котором должна сохраняться и быть доступна. Различают такие понятия, как тип сущности и экземпляр сущности. Понятие тип сущности относится к набору однородных предметов, событий, личностей, выступающих как единое целое. Экземпляр сущности относится к конкретной вещи в наборе. В диаграммах ER-модели сущность представляется в виде прямоугольника (в нотации Баркера), содержащего имя сущности.

Атрибут - поименованная характеристика сущности, определяющая его свойства и принимающая значения из некоторого множества значений. Каждый атрибут обеспечивается именем, уникальным в пределах сущности.

Атрибуты могут классифицироваться по принадлежности к одному из трех различных типов: описательные, указывающие, вспомогательные. Описательные атрибуты представляют факты, внутренне присущие каждому экземпляру сущности. Указывающие атрибуты используются для присвоения имени или обозначения экземплярам сущности. Вспомогательные атрибуты

используются для связи экземпляра одной сущности с экземпляром другого. Атрибуты подчиняются строго определенным правилам.

Множество из одного или нескольких атрибутов, значения которых однозначно определяют каждый экземпляр сущности, называются идентификатором. Каждый экземпляр сущности должен иметь хотя бы один идентификатор. Если идентификаторов несколько, один из них выбирается как привилегированный.

Связь (Relationship) - это поименованная графически изображаемая ассоциация, устанавливаемая между сущностями и представляющая собой абстракцию набора отношений, которые систематически возникают между различными видами предметов в реальном мире. Большинство связей относятся к категории бинарных и имеют место между двумя сущностями.

Среди бинарных связей существуют три фундаментальных вида связи: один-к-одному (1:1), один-ко-многим (1:M), многие-ко-многим (M:M). Связь один-к-одному (1:1) существует, когда один экземпляр одной сущности связан с единственным экземпляром другой сущности. Связь один-ко-многим (1:M) имеет место, когда один экземпляр одной сущности связан с одним или более экземпляром другой сущности и каждый экземпляр второй сущности связан только с одним экземпляром первой сущности. Связь многие-ко-многим (M:N) существует, когда один экземпляр одной сущности связан с одним или более экземпляром другой сущности и каждый экземпляр второй сущности связан с одним или более экземпляром первой сущности.

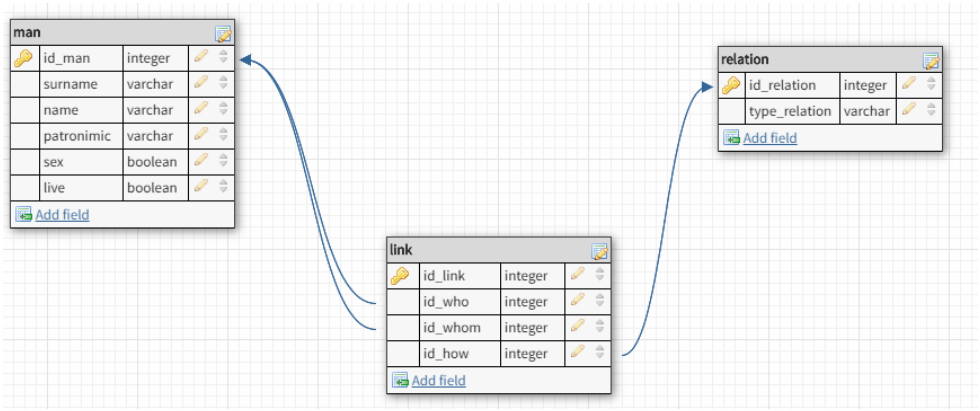

Рис. 2.1. Инфологическая модель

2.2 Даталогическое проектирование

Логическое (даталогическое) проектирование — В реляционных БД даталогическое или логическое проектирование приводит к разработке схемы БД, то есть совокупности схем отношений, которые адекватно моделируют абстрактные объекты предметной области и семантические связи между этими объектами. Основой анализа корректности схемы являются так называемые функциональные зависимости между атрибутами БД. Некоторые зависимости между атрибутами отношений являются нежелательными из-за побочных эффектов и аномалий, которые они вызывают при модификации БД. При этом под процессом модификации БД мы понимаем внесение новых данных в БД или удаление некоторых данных из БД, а также обновление значений некоторых атрибутов.

Однако этап логического или даталогического проектирования не заканчивается проектированием схемы отношений. В общем случае в результате выполнения этого этапа должны быть получены следующие результирующие документы:

Описание концептуальной схемы БД в терминах выбранной СУБД.

Описание внешних моделей в терминах выбранной СУБД.

Описание декларативных правил поддержки целостности базы данных.

Разработка процедур поддержки семантической целостности базы данных.

Однако перед тем как описывать построенную схему в терминах выбранной СУБД, нам надо выстроить эту схему. Именно этому процессу и посвящен данный раздел.

Мы должны построить корректную схему БД, ориентируясь на реляционную модель данных.

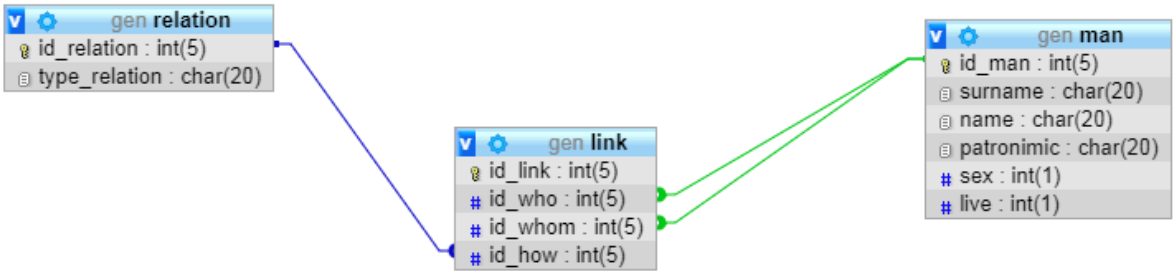

Рис. 2.2 Даталогическая модель

Глава 3. РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ

3.1 Выбор средств разработки

На данный момент существует большое количество средств, с помощью которых возможно создать сайт, который смог бы автоматизировать генеалогическое дерево. Однако для реализации данной системы со стороны сервера был выбран Mysql и Apache. Mysql выбран за высокий уровень безопасности. Mysql не требовательный к вычислительным мощностям. Это делает Mysql невероятно гибким и надежным. Среди прочего, он умеет создавать, хранить и извлекать сложные структуры данных. Еще одним аргументом к использованию данного продукта является то, что весь исходный код данного инструмента находится в открытом доступе, а компоненты из которых она состоит связаны между собой с помощью интерфейсов, что позволяет заменить любой компонент собственной реализацией в случае если его функционал не отвечает требованиям поставленной задачи. Еще одним достоинством является то что Mysql одна из самых быстрых СУБД существующих в настоящий момент. Поскольку возможна придется иметь дело с большим количеством данных в базе данных то этот момент являеться одним из весьма серьезных аргументов в пользу выбора данной СУБД. В отличии от postgres Mysql не требует лишних усилий при установке и администрировании поскольку не требуется подключать ничего что уже не будет установлено в процессе самой установки. Также Mysql не требовательный к большим объемам данных поскольку может обрабатывать большие количества записей одной базе, что также весьма важно. Так например в одной таблице, по уверению компании разработчика, может помещаться до 50 миллионов строк. может помещаться до 50 миллионов строк. Mysql совместим со всеми операционными системами, и не требовательный к аппаратной части следствии чего он сможет эффективно работать даже на устаревших ПК что весьма удовлетворительно.

Малый размер Mysql в сравненни с postgres выдвигает Mysql на первые места поскольку у конкурентов требуется большое дисковое пространство для той же работы.

Apache HTTP web – сервер выбран за его возможность работать на любой операционной системе. Так же Apache выбирают за его гибкость, мощность и широкую распостранненость. Он может быть дополнен большим количеством дополняемых модулей и выполнять программы на большом количестве языков программирования. Apache позволяет конфигурировать на уровне директорий.

Так как конфигурационные файлы находятся в директориях с контентом, Apache вынужден при обработке каждого запроса проверять не содержит ли каждый компонент запрашиваемого пути файл .htaccess и исполнять директивы в найденных файлах. Это позволяет децентрализовать конфигурирование веб-сервера, что позволяет реализовать модификацию URL'ов (URL rewrite), ограничения доступа, авторизацию и аутентификацию и даже политики кеширования.

PHP(Hypertext processor) используется за пять его основных характеристик:

- Традиционность;
- Простоту;
- Эффективность;
- Безопасность;
- Гибкость.

Код PHP будет понятен большинству программистов поскольку он основан на таких языках как Си и Pascal. Что снижает усилия для первоначального обучения в структуру языка. PHP – язык с в общем-то универсальным и понятным синтаксисом. Именно поэтому PHP является наиболее распостраненным среди программистов программирующих веб-приложения.

PHP не требует подключения никаких библиотек или указывать специальные параметры компиляции которые потребовались бы в других

языках. Механизм выполнения кода на языке PHP очень простой. В PHP выполняется все что находится между двумя экранирующими последовательностями (<? ?>). При условии правильного написания кода он исполниться в точности как он был написан программистом. PHP может быть встроен непосредственно в код html страницы, и код все равно будет корректно обрабатываться PHP интерпретатором. Большое разнообразие функций PHP избавят вас от написания многострочных пользовательских функций

Эффективность является исключительно важным фактором при программировании для многопользовательских сред, к числу которых относится и web .

Важное преимущество PHP заключается в его «движке». Не являясь ни компилятором ни интерпретатором «движок» PHP является транслирующим интерпретатором. Такое устройство «движка» PHP позволяет обрабатывать сценарии с достаточно высокой скоростью. Что крайне полезно при написании программы использующей базу данных с большим количеством данных.

Большинство PHP-сценариев обрабатываются быстрее аналогичных им программ, написанных на аналогичных программных продуктах. Что делает PHP весьма полезным для и более выгодным для написания веб приложений по сравнению с остальными программными продуктами.

PHP в контексте средств безопасности делиться на средства безопасности системного уровня и средства безопасности которые работают на уровне приложения. предоставляет в распоряжение разработчиков и администраторов гибкие и эффективные средства безопасности, которые условно делятся на две категории: средства системного уровня и средства уровня приложения.

В PHP реализованы механизмы безопасности, находящиеся под управлением администраторов; при правильной настройке PHP это обеспечивает максимальную свободу действий и безопасность. PHP может работать в так называемом безопасном режиме (safe mode), который ограничивает возможности применения PHP пользователями по ряду важных

показателей. Например, можно ограничить максимальное время выполнения и использование памяти (неконтролируемый расход памяти отрицательно влияет на быстродействие сервера). По аналогии с cgi-bin администратор также может устанавливать ограничения на каталоги, в которых пользователь может просматривать и исполнять сценарии PHP, а также использовать сценарии PHP для просмотра конфиденциальной информации на сервере (например, файла passwd).

В стандартный набор функций PHP входит ряд надежных механизмов шифрования. PHP также совместим с многими приложениями независимых фирм, что позволяет легко интегрировать его с защищенными технологиями электронной коммерции (e-commerce). Другое преимущество заключается в том, что исходный текст сценариев PHP нельзя просмотреть в браузере, поскольку сценарий компилируется до его отправки по запросу пользователя. Реализация PHP на стороне сервера предотвращает похищение нетривиальных сценариев пользователями, знаний которых хватает хотя бы для выполнения команды View Source.

Поскольку PHP является встраиваемым (embedded) языком, он отличается исключительной гибкостью по отношению к потребностям разработчика. Хотя PHP обычно рекомендуется использовать в сочетании с HTML, он с таким же успехом интегрируется и в JavaScript, WML, XML и другие языки. Кроме того, хорошо структурированные приложения PHP легко расширяются по мере необходимости (впрочем, это относится ко всем основным языкам программирования).

Нет проблем и с зависимостью от браузеров, поскольку перед отправкой клиенту сценарии PHP полностью компилируются на стороне сервера. В сущности, сценарии PHP могут передаваться любым устройствам с браузерами, включая сотовые телефоны, электронные записные книжки, пейджеры и портативные компьютеры, не говоря уже о традиционных ПК. Программисты, занимающиеся вспомогательными утилитами, могут запускать PHP в режиме командной строки.

Поскольку PHP не содержит кода, ориентированного на конкретный web-сервер, пользователи не ограничиваются определенными серверами (возможно, незнакомыми для них). Apache, Microsoft IIS, Netscape Enterprise Server, Stronghold и Zeus — PHP работает на всех перечисленных серверах. Поскольку эти серверы работают на разных платформах, PHP в целом является платформенно-независимым языком и существует на таких платформах, как UNIX, Solaris, FreeBSD и Windows 95/98/NT/2000/XP/2003.

Наконец, средства PHP позволяют программисту работать с внешними компонентами, такими как Enterprise Java Beans или COM-объекты Win32. Благодаря этим новым возможностям PHP занимает достойное место среди современных технологий и обеспечивает масштабирование проектов до необходимых пределов.

Стратегия Open Source, и распространение исходных текстов программ в массах, оказало несомненно благотворное влияние на многие проекты, в первую очередь — Linux, хотя и успех проекта Apache сильно подкрепил позиции сторонников Open Source. Сказанное относится и к истории создания PHP, поскольку поддержка пользователей со всего мира оказалась очень важным фактором в развитии проекта PHP.

Принятие стратегии Open Source и бесплатное распространение исходных текстов PHP оказало неоценимую услугу пользователям. Вдобавок, отзывчивое сообщество пользователей PHP является своего рода «коллективной службой поддержки», и в популярных электронных конференциях можно найти ответы даже на самые сложные вопросы.

3.2 Технология построения веб сайта

Проектирование базы данных.

Для работы программы необходима программа для построения базы данных и программа для построения веб-сайта.

Автоматизированная система должна обеспечить выполнение следующих функций:

- Добавление родственников для каждого отдельно взятого элемента дерева;
- Возможность редактирования информации отдельно взятого человека
- Возможность поиска по родственников по фамилии и имени
- Возможность добавлении определения названия родственной вязи между двумя отдельными элементами
- Возможность авторизации и сохранения данных .

Чтобы реализовать базу данных была выбрана программа Mysql. Таким образом, выбрав Mysql можно проектировать базу данных и манипулировать этими данными. При разработке было принято решение о том что необходимыми таблицами для реализации автоматизированной системы. Ниже следует перечень таблиц на основе которых реализуется база данных.

1) Таблица man создана для заполнения этой таблицы людьми содержит следующие поля
   - «id-man» – это первичный ключ, который базируется на идентификаторе;
   - «surname» типа «varchar» предназначено фамилию;
   - строковое свойство «name» хранит имя;
   - "patronimic" типа varchar предназначенная для хранения отчества;
   - "sex" типа BOOLEAN предназначенного для хранения пола;
   - "live" типа BOOLEAN для хранения информации о смерти человека;

2) Таблица «Link» предназначена для хранения родственной связи кто кому и кем приходиться для этого в таблице используются внешние ключи:

- "id_link" типа int первичный ключ;
- "id_who" типа integer предназначен для связывания с таблицей man для того что бы указать первоначального участника связи «кто приходиться» ;
- "id_whom" integer предназначен для связывания с таблицей man для того что бы указать второстепенного участника связи «кому приходиться» ;
- "id_how" integer предназначен для связывания с таблицей relation для того что бы указать тип связи;

3) Таблица «Relations» - таблица с перечнем типов связей между людьми:

- "id_relation" типа int - первичный ключ;
- "type_relation" типа varchar(10) хранит тип родственной связи между людьми;

4) Класс «users» - таблица с хранением авторизированных пользователей:

- «id-user» первичный ключ;
- username, типа string для хранения логина пользователя;
- «password» типа string для хранения пароля пользователя;

Поскольку именно сайт является основным способом взаимодействия с пользователем то необходимо рассмотреть структуру сайта и вкратце объяснить его работу.

Первой странице которой пользуется кто-нибудь кто зашел на сайт – это главная страница. Дизайн странице представлен навигационным меню в верхней части страницы. Листинг меню приведен в листинге 3.1

Листинг 3.1. Начало листинга

```
<div class="navbar fixed-top flex-md-nowrap">
```

```
<div class="navbar-inner ">
<div class="container">
<a class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
 <span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</a>
<a class="brand" href="#">Генеалогическое дерево</a>
<div class="nav-collapse">
<ul class="nav">
<li class="active"><a href="index.php">Главная</a></li>
<li><a href="search.php">Поиск</a></li>
</ul>
</div>
</div>
</div>
</div>
```

Листинг 3.1. Конец листинга

Для того что бы увидеть людей и связи между ними необходимо выводить эти данные в виде генеалогического дерева. Данные выводятся из базы данных где эта информация храниться. Подключение к базе данных можно увидеть в листинге 3.2.

Листинг 3.2. Начало листинга

```
$db_host = 'localhost';
$db_name = 'gen';
$db_username = 'mysql';
$db_password = 'mysql';
$connect_to_db = mysql_connect($db_host, $db_username, $db_password)
```

```
or die("Could not connect: " . mysql_error());
mysql_select_db($db_name, $connect_to_db)
or die("Could not select DB: " . mysql_error());
```

Листинг 3.2. Конец листинга

За основу берется вывод при помощи многоуровневого списка. Представление этого списка изменено на странице при помощи стилей CSS. Вывод осуществляется при помощью рекурсии и запроса к базе данных. Это можно увидеть в листинге 3.3.

Листинг 3.3. Начало листинга

```
function view_cat($mas,$par) {
if($mas<$par) {
return;
}
$par1=$par*2;
$par2=$par*2;
$par2=$par2+1;
   $qr_result = mysql_query("select surname,name from man where
id_man='".$par."'")
   or die(mysql_error());
  while($data = mysql_fetch_array($qr_result)){
$row=$data;
  }
echo '<ul>';
echo '          <li>';
echo '                  <a href="update.php">'.$row[0].' '.$row[1].'</a>';
echo '                  <ul>';
echo '                          <li>';
if($par<$mas)
        {       view_cat($mas,$par1);}
```

```
echo '                          </li>';
echo '                          <li>';
if($par<$mas)
{       view_cat($mas,$par2);}
echo '                          </li>';
echo '                          </li>';
echo '                  </ul>';
echo '</ul>';Листинг 3.3. Конец листинга
```

Однако для того что бы вывести дерево необходимо сначала авторизороваться. Регистрация происходит при помощи сессий на PHP. Что можно увидеть на листинге 3.4.

Листинг 3.4. Начало листинга

```
if(!defined("IN_ADMIN")) die;

session_start();
$access = array();
$access = file("access.php");
$login = trim($access[1]);
$passw = trim($access[2]);
if(!empty($_POST['enter']))
{
        $_SESSION['login'] = $_POST['login'];
        $_SESSION['passw'] = $_POST['passw'];
}

if(empty($_SESSION['login']) or
   $login != $_SESSION['login'] or
   $passw != $_SESSION['passw']     )  echo '</ul>';
```

Листинг 3.4. Конец листинга

Для добавления человека существует форма в которой пользователь вводит данные человека. А именно имя, фамилию и отчество. И выбрать человека из выпадающего списка родственника которого вы введете. Также присутствует выпадающий список из возможных родственных отношений. Это изображено на листинге 3.5.

Листинг 3.5. Начало листинга

```
    echo '<form id="centerLayer" action = "upd.php" method = "POST">';
    echo '<p><label for="ln">Фамилия:</label> <input type="text" name="new_surname" /></p>';
    echo '<p><label for="ln">Имя:</label> <input type="text" name="new_name" /></p>';
    echo '<p><label for="ln">Отчество:</label> <input type="text" name="new_patronimic" /></p>';
    echo '<label class="checkbox">';
    echo '<input type="checkbox"  class="layer2" value="sex"> Пол ';
    echo '</label>';
    echo '<label class="checkbox">';
    echo '<input type="checkbox" class="layer2" value="live"> Жив ';
    echo '</label>';
    echo '<p><select size="3" multiple name="rel[]">';
    echo '    <option disabled>степень родства</option>';
    echo '   </select></p>';
    echo '<p><input type="submit" name = "add" value = "Изменить"/></p>';
```

Листинг 3.5. Конец листинга

Для изменения данных в каком либо объектов дерева необходимо нажать на интересующий объект дерева нажать на него и пользователя перенесет на страницу редактирования дерева где пользователь сможет ввести обновленные данные взамен устаревших. Это можно увидеть на листинге 3.6.

Листинг 3.6. Начало листинга

```
echo '<form action = "ins.php" method = "POST">';
echo '<p><select size="3" multiple name="hero[]">';
echo '    <option disabled>Выберите родственника</option>';
echo '   </select></p>';
echo '<p><label for="ln">Фамилия:</label> <input type="text" name="new_surname" /></p>';
echo '<p><label for="ln">Имя:</label> <input type="text" name="new_name" /></p>';
echo '<p><label for="ln">Отчество:</label> <input type="text" name="new_patronimic" /></p>';
echo '<label class="checkbox">';
echo '<input type="checkbox"  class="layer2" value="sex"> Пол ';
echo '</label>';
echo '<label class="checkbox">';
echo '<input type="checkbox" class="layer2" value="live"> Жив ';
echo '<p><select size="3" multiple name="hero[]">';
echo '   </select></p>';
echo '</label>';
echo '<p><input type="submit" name = "add" value = "Добавить"/></p>';
echo '</form>';
```

Листинг 3.6. Конец листинга

Для поиска в генеалогическом дереве необходимо нажать на кнопку поиск в навигационном меню верху страницы. После чего откроется страница с результатами запроса пользователя. С кодом можно ознакомится в листинге 3.7.

Листинг 3.7. Начало листинга

```
$search=$_POST['search_class'];
$sql= "SELECT * FROM man where name LIKE '%".$search."%'";
$res = $db->query($sql);
echo '<table border=1>';
echo '<tr><th>Фамилия</th>' .
'<th>Имя</th>' .
'<th>Отчество</th>'
 '</tr>';
$result = $res->FetchAll(PDO::FETCH_NUM);
foreach($result as $row) {
echo "<tr>";
echo "<td>$row[1]</td>";
echo "<td>$row[2]</td>";
echo "<td>$row[3]</td>";
echo "</tr>";
}
```

Листинг 3.7. Конец листинга

Одной из важнейших частей проектирования базы данных является нормализация базы данных то есть разделение групп элементов, уменьшение избыточности. При использовании не нормализированных то есть не приведенных к третьей нормальной форме может привести к нарушению базы данных.

Нормализация – процесс организации баз данных согласно правилам организации баз данных позволяющие создавать более гибкие, менее избыточные базы данных без несогласованностей между таблицами.

Избыточность данных это непродуктивное использование базы данных. Если данные которые требуется изменить находятся не в одной в одной таблице а в нескольких это приведет к тому что эти изменение необходимо производить и в других таблицах в которых находятся значения которые требуется изменить.

Несогласованные зависимости в рамках базы данных это записи расположенные не в тех таблицах и не в такой последовательности в какой это требует логика составления базы данных. Несогласованные зависимости могут затруднять доступ к данным, так как путь к данным при этом может отсутствовать или быть неправильным.

Существует шесть правил нормализации базы данных. Традиционно сложилось правило называть их формами. При создании базы данных соответствующей первому правилу из шести говориться что эта база данных приведена к первой нормальной форме. При выполнении всех трех правил говориться что база приведена к третьей нормальной форме. Конечно же есть и другие правила, то есть формы нормализации но обычно используют только 3 первые нормальные формы.

Как обычно бывает невозможно привести базу данных к нормализации то есть полностью избежать избыточности и несогласованности данных невозможно. Как правило, для выполнения нормализации приходится создавать дополнительные таблицы, и некоторые клиенты считают это нежелательным. Собираясь нарушить одно из первых трех правил нормализации, убедитесь в том, что в приложении учтены все связанные с этим проблемы, такие как избыточность данных и несогласованные зависимости.

Первая нормальная форма

Не должно быть повторяющихся групп в каждой отдельно взятой таблице

Для каждого набора связанных данных необходимо создать свою таблицу логически объединяющая все эти поля.

Каждая таблица должна иметь поле которое называется первичным ключом для идентификации таблицы как уникальную единицу базы данных. Первичный ключ для каждого поля должен быть уникальным.

Нельзя использовать больше одного поля для хранения данных которые имеют схожую или вовсе идентичный характер данных. Стоит либо разнести эти поля в разные таблицы либо заносить все данные в оно поле одной таблицы.

Что произойдет при добавлении третьего поставщика? Добавление третьего поля нежелательно, так как для этого нужно изменять программу и таблицу, поэтому данный способ плохо адаптируется к динамическому изменению числа поставщиков. Вместо этого можно поместить все сведения о поставщиках в отдельную таблицу и связать товары с поставщиками с помощью кодов товаров или поставщиков с товарами с помощью кодов поставщиков.

Вторая нормальная форма

Создайте отдельные таблицы для наборов значений, относящихся к нескольким записям.

Свяжите эти таблицы с помощью внешнего ключа.

Записи могут зависеть только от первичного ключа таблицы (составного ключа, если необходимо). То есть некя таблиц может быть связана посредством своего внешнего ключа только с первичным ключом той таблицы на которую он ссылается.

Третья нормальная форма

Устраните поля, не зависящие от ключа.

Значения, входящие в запись и не являющиеся частью ключа этой записи, не принадлежат таблице. Если содержимое группы полей может относиться более чем к одной записи в таблице, подумайте о том, не поместить ли эти поля в отдельную таблицу.

Исключение. Выполнять нормализацию баз данных до третьей нормальной формы теоретически желательно, но не всегда практично.. С теоретической точки зрения нормализация желательна. Однако значительное увеличение числа маленьких таблиц может привести к снижению производительности СУБД или исчерпанию памяти и числа дескрипторов открытых файлов.

Выполнять нормализацию до третьей нормальной формы может быть целесообразно только для часто изменяемых данных. Если при этом сохранятся зависимые поля, спроектируйте приложение так, чтобы при изменении одного из этих полей пользователь должен был проверить все связанные поля.

Другие формы нормализации

Кроме описанных нормальных форм, существует четвертая нормальная форма, которую также называют нормальной формой Бойса-Кодда (BCNF), и пятая нормальная форма, но на практике они применяются редко. Несоблюдение этих правил может привести к ухудшению архитектуры базы данных, но на функциональности это сказаться не должно.

ГЛАВА 4. Тестирование

4.1 Программа и методика тестирования

Объектом испытаний, рассматриваемым в данной главе, является ранее спроектированная и разработанная «Автоматизированная система генеалогическое дерево».

Целью испытаний системы является проверка ее работоспособности.

Полный перечень требований к данной системе перечислен в третьем разделе первой главы, однако пунктами, требующими указания в данном разделе, являются следующие:

- возможность предоставить посетителям уже введенной информации;
- возможность посетителей изменять уже введенную информацию;
- возможность пользователем добавлять связи любого человека в генеалогическом дереве.;
- возможность авторизации сохранения введенных значений.;
- возможность поиска по имени, фамилии;
- возможность регистрации и авторизации в системе;
- разграничение установления названия родственной связи между двумя людьми в генеалогическом дереве;

Исходя из приведенных выше требований - был разработан следующий алгоритм для проведения испытаний автоматизированной системы.

1. Тестирование возможностей системы для авторизованного пользователя.

1.1. Запустить информационную систему.

1.2. Нажать на ссылку «вход в систему».

1.3. Ввести email.

1.4. Введите пароль.

1.5. После появления на главной странице сайта введите фамилию первого объекта дерева.

1.6. Введите имя.

1.7. Введите отчество.

1.8. Укажите пол: галочка если мужчина, нет если женщина.

1.9. Поставте галочку если человек жив.

1.10. Указать степень родства.

1.11. В случае необходимости добавления объекта в дерево который не является первым выберите родственника из выпадающего списка.

1.12. Повторите пункты с 1.6 по 1.10.

1.13. Для поиска родственника нажмите поиск в навигационном меню.

1.14. После перехода на страницу поиска введите условия поиска.

1.15. После обновления страницы появятся результаты запроса в виде таблицы.

1.16. Для изменения нажать на соответствующий объект в дереве .

1.17. В появившимся окне изменить старые значения введя новые.

1.18. Для удаления необходимо нажать на кнопку креста в левом верхнем углу нужного квадрата.

4.2 Тестирование информационной системы

Начать проведение испытаний следует с тестирования возможностей системы для авторизованного пользователя.

Первым делом запускаем информационную системы, для этого в адресной строке веб браузера вводим адрес страницы. После того как главная страница сайта загрузилась, мы видим главную страницу с навигационным меню с названием возможность регистрации и авторизации. Работа главной страницы изображена на рисунке 4.1.

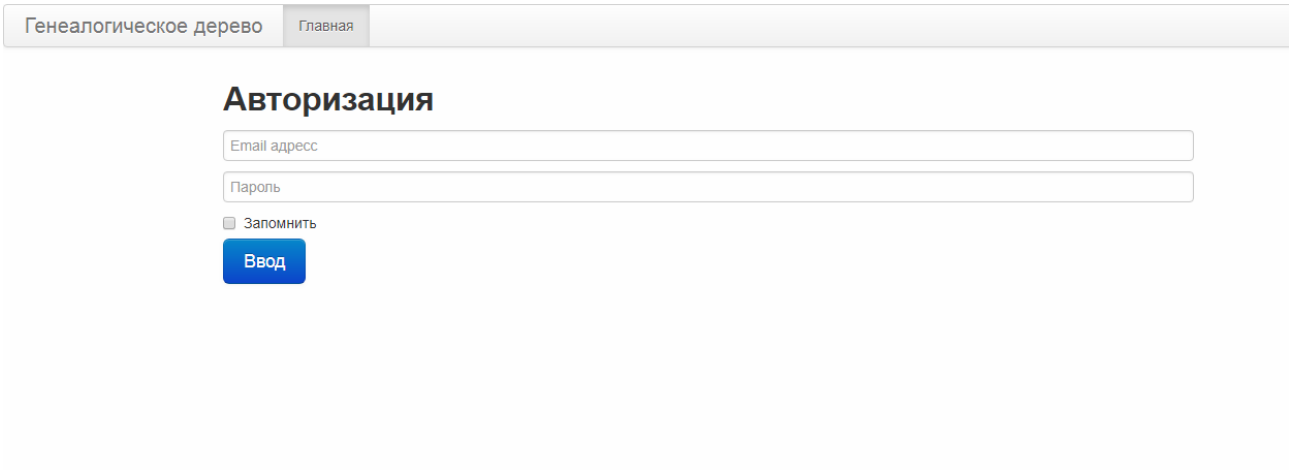

Рис. 4.1. Авторизация

Выполняя следующий шаг и зарегистрировавшись мы попадаем на страницу на которой можно добавить родственников. Результат выполнения этого действия проиллюстрирован на рисунке 4.3

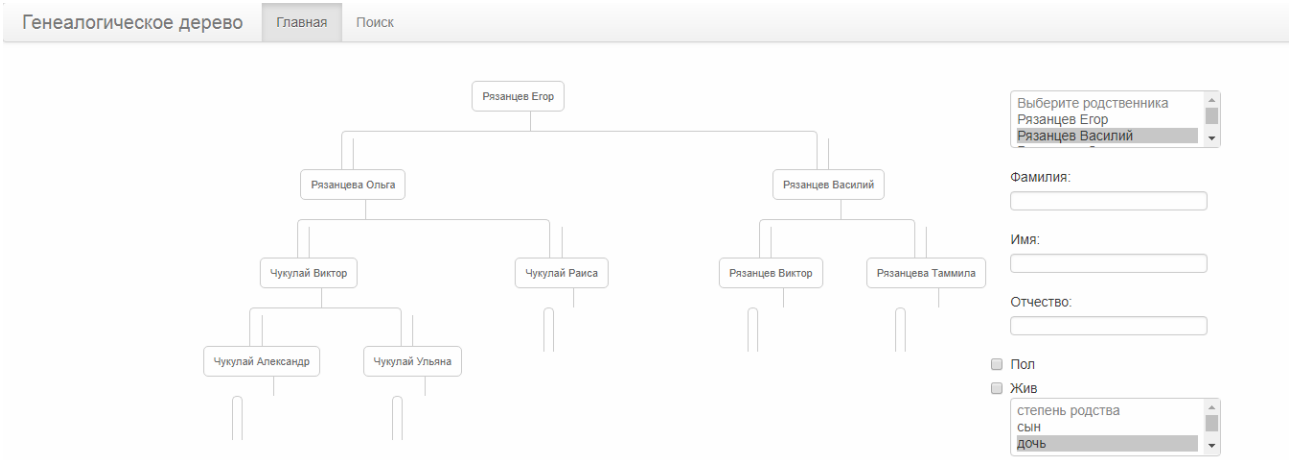

Рис. 4.3. Страница с генеалогическим деревом

Далее убедимся в том, что система предоставляет пользователям возможность изменить информацию объекта. Для этого нажимаем на кнопку стилизированную под карандаш. С результатами этого действия можно ознакомиться на рисунке 4.4

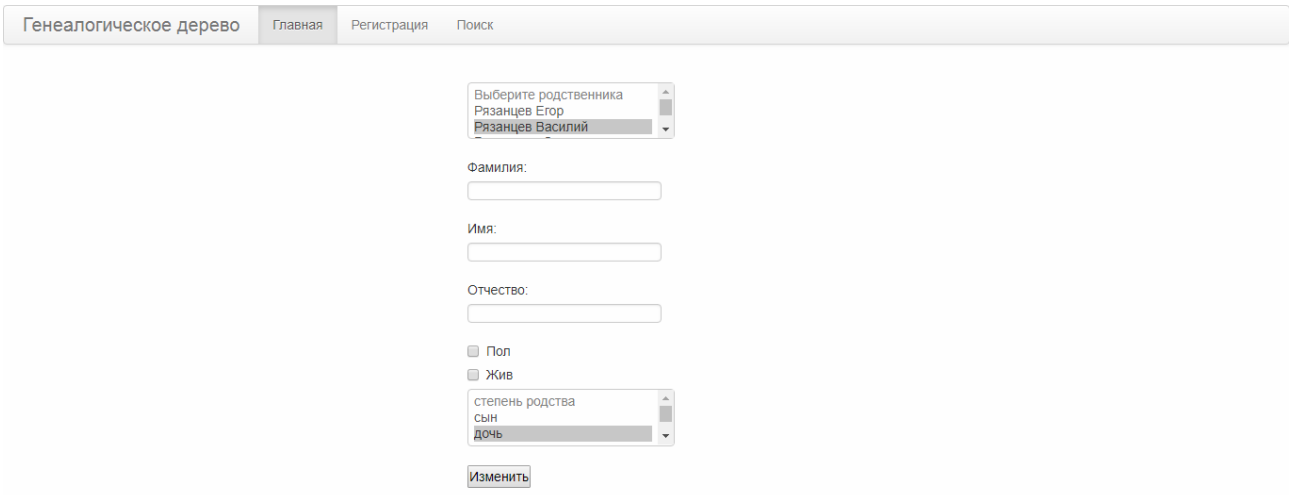

Рис. 4.4. Страница редактирования объекта

Следующим шаг - проверка поиска. Для того что бы проверить необходимо ввести в строку поиска интересующую информацию. Рисунок 3.5

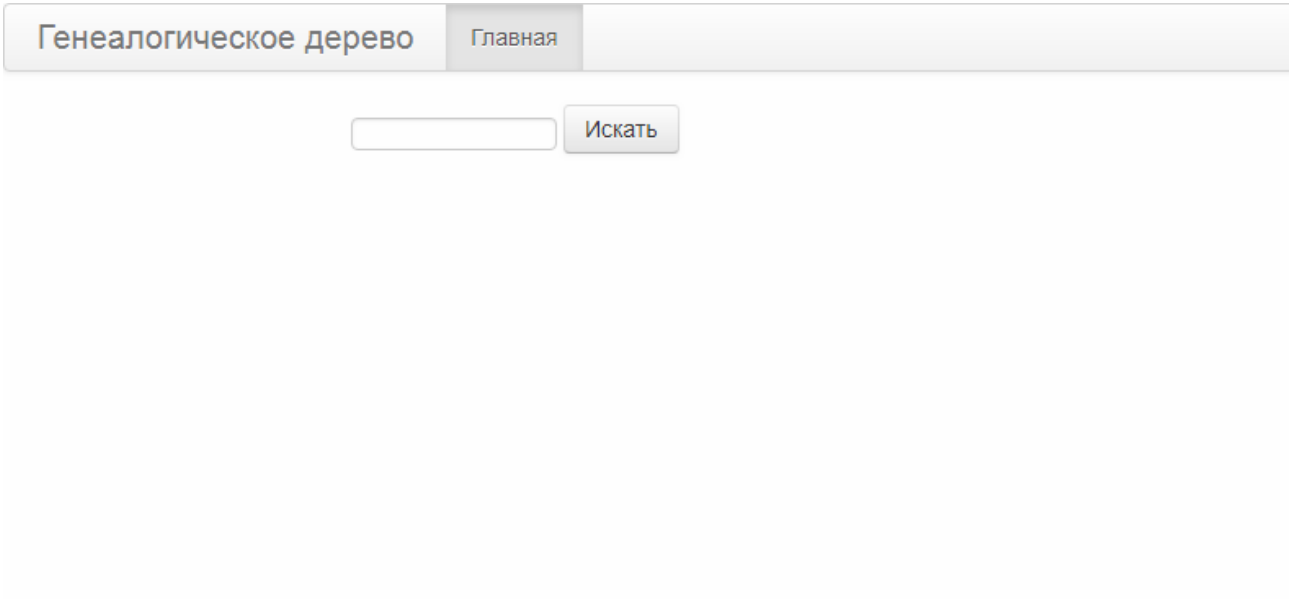

Рис. 4.5. Поиск по дереву

Для проверки поиска введем значение «Василий» результаты поиска будут видны на рисунке 4.6

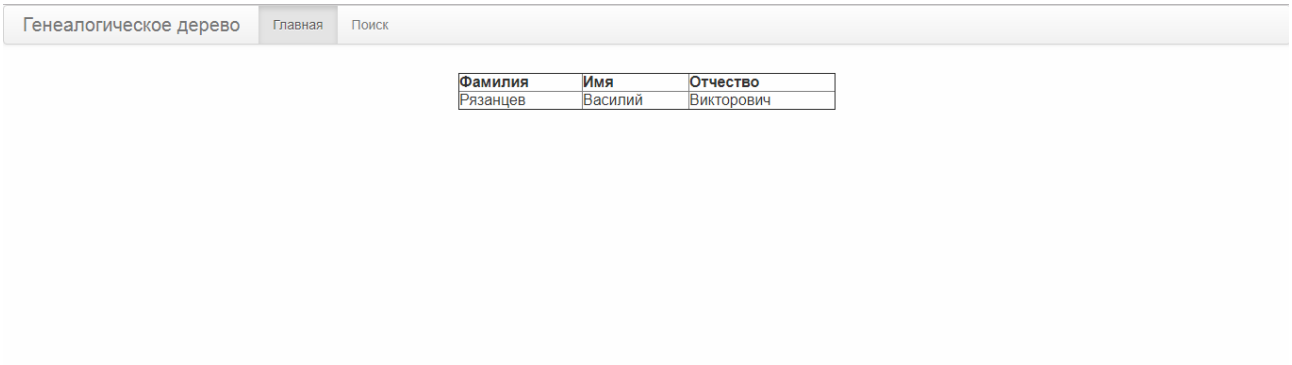

Рис. 4.6. Результаты поиска

По результатам проведения испытаний разработанной информационной системы было установлено, что система полностью работоспособна и готова к введению в эксплуатацию. Функциональные возможности разработанной системы полностью удовлетворяют по всем пунктам требований к системе, поставленным в начале ее проектирования

ЗАКЛЮЧЕНИЕ

В результате выполнения дипломной работы была спроектирована и разработана автоматизированная генеалогического дерева.

Данная информационно-справочная система реализована в форме веб-приложения, которое выполняет информационное сопровождение деятельности генеалогического дерева и позволяет любому пользователю сети интернет ввести изменить и посмотреть на отображение информации. Помимо этого система позволяет проводить поиск родственников и производить выяснение название родственных связей .

После анализа предметной области, изучения уже имеющихся систем решающих схожие задачи, были сформулированы требования, на основе которых спроектирована и разработана автоматизированная система. По результатам тестирования системы, следует сделать вывод – разработанное веб приложение полностью удовлетворяет всем требования, предъявленным к системе на этапе постановки задачи.

СПИСОК ЛИТЕРАТУРЫ

1. Чем MySql лучше других SQL баз данных с открытым исходным кодом.- 2012 [Электронный ресурс]. – URL: https://habr.com/post/282764/ (дата обращения: 07.10.2017).
2. PHP.SU - Преимущества PHP | Основы PHP - 2011 [Электронный ресурс]. – URL: http://www.php.su/php/?opport (дата обращения: 25.03.2018).
3. Web-сервер - Apache // Около программирования - тьюториалы и статьи по веб-программированию. - 2011 [Электронный ресурс]. – URL: ipmnet.ru/~sadilina/Fedora/50.html (дата обращения: 21.11.2017).
4. MySQL: особенности и сферы применения - BYTE/Россия // Типичный программист – 2015 [Электронный ресурс]. – URL: https://www.bytemag.ru/articles/detail.php?ID=6547 (дата обращения: 03.11.2017).
5. Примеры SQL запросов к базе данных MySQL - URL: http://tradebenefit.ru/primery-mysql-zaprosov (дата обращения: 20.12.2017).
6. Лекция 5: Диаграмма активностей: крупным планом // Национальный Открытый Университет «ИНТУИТ» - 2003 [Электронный ресурс]. – URL: http://www.intuit.ru/studies/courses/1007/229/lecture/5958 (дата обращения: 20.12.2016).
7. Кузнецов Максим, Симдянов Игорь. MySQL на примерах. — Спб.: «БХВ-Петербург», 2008. — С. 952.
8. Кузнецов Максим, Симдянов Игорь. Самоучитель MySQL 5. — Спб.: «БХВ-Петербург», 2006. — С. 560.
9. Поль Дюбуа. MySQL, 3-е издание = MySQL, 3ed. — М.: «Вильямс», 2006. — 1168 с.
10. В. Васвани. MySQL: использование и администрирование = MySQL Database Usage & Administration. — М.: «Питер», 2011. — 368 с.
11. Чем MySql лучше других SQL баз данных с открытым исходным кодом.- 2012 [Электронный ресурс]. – URL: https://habr.com/post/282764/ (дата обращения: 07.10.2017).

12. Стив Суэринг, Тим Конверс, Джойс Парк. PHP и MySQL. Библия программиста, 2-е издание = PHP 6 and MySQL 6 Bible. — М.: «Диалектика», 2010. — 912 с.

13. Кузнецов Максим, Симдянов Игорь. PHP на примерах. — 2-е изд. перераб. и доп. — СПб.: «БХВ-Петербург», 2011. — С. 400.

14. Квентин Зервас. Web 2.0: создание приложений на PHP = Practical Web 2.0 Applications with PHP. — М.: «Вильямс», 2009. — С. 544.

ПРИЛОЖЕНИЕ 1

Tree.css

```
* {margin: 0; padding: 0;}
                .tree ul {
                        padding-top: 20px; position: relative;


                        -webkit-transition: all 0.5s;
                        -moz-transition: all 0.5s;
                        transition: all 0.5s;
                }


  .centerLayer {
    margin-left: 30%;
    margin-right: 30%;
    background: #fc0;
    padding: 10px;
   }


                .tree li {
                        float: left; text-align: center;
                        list-style-type: none;
                        position: relative;
```

```
        padding: 20px 5px 0 5px;


        -webkit-transition: all 0.5s;
        -moz-transition: all 0.5s;
        transition: all 0.5s;
}


.tree li::before, .tree li::after{
        content: '';
        position: absolute; top: 0; right: 50%;
        border-top: 1px solid #ccc;
        width: 50%; height: 45px;
        z-index: -1;
}
.tree li::after{
        right: auto; left: 50%;
        border-left: 1px solid #ccc;
}


.tree li:only-child::after, .tree li:only-child::before {
        display: none;
}
```

```
.tree li:only-child{ padding-top: 0;}
.tree li:first-child::before, .tree li:last-child::after{
	border: 0 none;
}
.tree li:last-child::before{
	border-right: 1px solid #ccc;
	border-radius: 0 5px 0 0;

	-webkit-transform: translateX(1px);
	-moz-transform: translateX(1px);
	transform: translateX(1px);

	-webkit-border-radius: 0 5px 0 0;
	-moz-border-radius: 0 5px 0 0;
	border-radius: 0 5px 0 0;
}
.tree li:first-child::after{
	border-radius: 5px 0 0 0;
	-webkit-border-radius: 5px 0 0 0;
	-moz-border-radius: 5px 0 0 0;
}
.tree ul ul::before{
```

```
        content: '';
        position: absolute; top: -12px; left: 50%;
        border-left: 1px solid #ccc;
        width: 0; height: 32px;
        z-index: -1;
}

.tree li a{
        border: 1px solid #ccc;
        padding: 5px 10px;
        text-decoration: none;
        color: #666;
        font-family: arial, verdana, tahoma;
        font-size: 11px;
        display: inline-block;
        background: white;

        -webkit-border-radius: 5px;
        -moz-border-radius: 5px;
        border-radius: 5px;

        -webkit-transition: all 0.5s;
```

```
        -moz-transition: all 0.5s;
        transition: all 0.5s;
}
.tree li a+a {
        margin-left: 20px;
        position: relative;
}
.tree li a+a::before {
        content: '';
        position: absolute;
        border-top: 1px solid #ccc;
        top: 50%; left: -21px;
        width: 20px;
}
.tree li a:hover, .tree li a:hover~ul li a {
        background: #c8e4f8; color: #000; border: 1px solid #94a0b4;
}
.tree li a:hover~ul li::after,
.tree li a:hover~ul li::before,
.tree li a:hover~ul::before,
.tree li a:hover~ul ul::before
{
```

```
                    border-color: #94a0b4;
                }
```

Index.php

```
<html lang="ru">
  <body>

  <link rel="stylesheet" href="tree.css" />
  <link rel="stylesheet" href="bootstrap.css" />
<div class="navbar fixed-top flex-md-nowrap">
<div class="navbar-inner ">
<div class="container">
<a class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
 <span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</a>
<a class="brand" href="#">Генеалогическое дерево</a>
<div class="nav-collapse">
<ul class="nav">
<li class="active"><a href="index.php">Главная</a></li>

<li><a href="search.php">Поиск</a></li>
```

```
</ul>
</div>
</div>
</div>
</div>
</div>
<?php
include 'functions.php';
    $db_host = 'localhost';
    $db_name = 'gen';
    $db_username = 'mysql';
    $db_password = 'mysql';
    //$db_table_to_show = 'student';
    $connect_to_db = mysql_connect($db_host, $db_username, $db_password)
    or die("Could not connect: " . mysql_error());
    mysql_select_db($db_name, $connect_to_db)
    or die("Could not select DB: " . mysql_error());
$par=1;
        $sql="select surname,name from man where id_man='".$par."'";
    $qr_result = mysql_query($sql)
    or die(mysql_error());
  while($data = mysql_fetch_array($qr_result)){
```

```
$row=$data;
  }
$mas=get_cat();
$par=1;
echo ' <div class="tree">';
view_cat($mas,$par);
echo ' </div>';
echo '<form action = "ins.php" method = "POST">';
echo '<p><select size="3" multiple name="hero[]">';
echo '    <option disabled>Выберите родственника</option>';
echo '    <option value="">Рязанцев Егор</option>';
echo '    <option selected value=" ">Рязанцев Василий</option>';
echo '    <option value="">Рязанцева Ольга</option>';
echo '    <option value=" ">Рязанцев Виктор</option>';
echo '    <option value=" ">Рязанцева Таммила</option>';
echo '    <option value=" ">Чукулай Раиса</option>';
echo '    <option value=" ">Чукулай Александр</option>';
echo '    <option value=" ">Чукулай Ульяна</option>';
echo '   </select></p>';
echo '<p><label for="ln">Фамилия:</label> <input type="text" name="new_surname" /></p>';
```

```
echo '<p><label for="ln">Имя:</label> <input type="text" name="new_name" /></p>';

echo '<p><label for="ln">Отчество:</label> <input type="text" name="new_patronimic" /></p>';

echo '<label class="checkbox">';

echo '<input type="checkbox"  class="layer2" value="sex"> Пол ';

echo '</label>';

echo '<label class="checkbox">';

echo '<input type="checkbox" class="layer2" value="live"> Жив ';

echo '<p><select size="3" multiple name="hero[]">';

echo '    <option disabled>степень родства</option>';

echo '    <option value="">сын</option>';

echo '    <option selected value=" ">дочь</option>';

echo '    <option value="">мать</option>';

echo '    <option value=" ">отец</option>';

echo '    <option value=" ">супруг</option>';

echo '    <option value=" ">супруга</option>';

echo '   </select></p>';

echo '</label>';



echo '<p><input type="submit" name = "add" value = "Добавить"/></p>';
```

```
echo '</form>';


        ?>
          <script src="jquery.js"></script>
          <script src="bootstrap.js"></script>
</body>
</html>
```

Functions.php

```
<html lang="ru">
  <body>
  <link rel="stylesheet" href="tree.css" />
<?php
function get_cat() {
$sql = "SELECT * FROM man";
$result = mysql_query($sql);
$mas=mysql_num_rows($result);
if(!$result) {
return NULL;
}
return $mas;
```

```
}
function view_cat($mas,$par) {
if($mas<$par) {
return;
}
$par1=$par*2;
$par2=$par*2;
$par2=$par2+1;
   $qr_result = mysql_query("select surname,name from man where id_man='".$par."'")
    or die(mysql_error());
  while($data = mysql_fetch_array($qr_result)){
$row=$data;
  }

echo '<ul>';
echo '          <li>';
echo '                  <a href="update.php">'.$row[0].' '.$row[1].'</a>';
echo '                  <ul>';
echo '                          <li>';
if($par<$mas)
        {       view_cat($mas,$par1);}
```

```
echo '                    </li>';
echo '                    <li>';
if($par<$mas)
{       view_cat($mas,$par2);}
echo '                    </li>';
echo '                    </li>';
echo '                </ul>';
echo '</ul>';
}
?>
</body>
</html>
```

Reg.php

```
<html lang="ru">
  <body>
  <link rel="stylesheet" href="tree.css" />
  <link rel="stylesheet" href="bootstrap.css" />
<div class="navbar fixed-top flex-md-nowrap">
<div class="navbar-inner ">
<div class="container">
<a class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
 <span class="icon-bar"></span>
```

```
<span class="icon-bar"></span>

<span class="icon-bar"></span>

</a>

<a class="brand" href="#">Генеалогическое дерево</a>

<div class="nav-collapse">

<ul class="nav">

<li class="active"><a href="index.php">Главная</a></li>

</ul>

</div>

</div>

</div>

</div>

</div>

<?php

if(!defined("IN_ADMIN")) die;


session_start();

$access = array();

$access = file("access.php");

$login = trim($access[1]);

$passw = trim($access[2]);

if(!empty($_POST['enter']))
```

```
{
        $_SESSION['login'] = $_POST['login'];
        $_SESSION['passw'] = $_POST['passw'];
}

if(empty($_SESSION['login']) or
   $login != $_SESSION['login'] or
   $passw != $_SESSION['passw']    )

{
?>
<div class="container">
<form class="form-signin">
<h2 class="form-signin-heading">Авторизация</h2>
<input type="text" class="input-block-level" name="login" placeholder="Email адресс">
<input type="password" class="input-block-level" name="passw" placeholder="Пароль">
<label class="checkbox">
<input type="checkbox" value="remember-me"> Запомнить
</label>
<button class="btn btn-large btn-primary" type="submit">Ввод</button>
```

```
</form>
</div>
  <?php
   die;
}
?>
</body>
</html>
```

Search.php

```
<html lang="ru">
  <body>

  <link rel="stylesheet" href="tree.css" />
  <link rel="stylesheet" href="bootstrap.css" />
<div class="navbar fixed-top flex-md-nowrap">
<div class="navbar-inner ">
<div class="container">
<a class="btn btn-navbar" data-toggle="collapse" data-target=".nav-collapse">
 <span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
```

```
</a>
<a class="brand" href="#">Генеалогическое дерево</a>
<div class="nav-collapse">
<ul class="nav">
<li class="active"><a href="index.php">Главная</a></li>
</ul>
</div>
</div>
</div>
</div>
</div>
<?php
include 'functions.php';
    $db_host = 'localhost';
    $db_name = 'gen';
    $db_username = 'mysql';
    $db_password = 'mysql';
    //$db_table_to_show = 'student';
    $connect_to_db = mysql_connect($db_host, $db_username, $db_password)
    or die("Could not connect: " . mysql_error());

    mysql_select_db($db_name, $connect_to_db)
```

```
    or die("Could not select DB: " . mysql_error());


echo '  <div class="container">';
echo '<form class="navbar-form action = "search1.php" pull-left">';
echo '  <input type="text" class="span2">';
echo '  <button type="submit" class="btn">Искать</button>';
echo '</form>';
$search=$_POST['search_class'];
$sql= "SELECT * FROM man where name LIKE '%".$search."%'";
$res = $db->query($sql);
echo '<table border=1>';
echo '<tr><th>Фамилия</th>' .
'<th>Имя</th>' .
'<th>Отчество</th>'
 '</tr>';
$result = $res->FetchAll(PDO::FETCH_NUM);
foreach($result as $row) {
echo "<tr>";
echo "<td>$row[1]</td>";
echo "<td>$row[2]</td>";
echo "<td>$row[3]</td>";
echo "</tr>";
```

```
}

?>

</body>

</html>
```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

«___» _____ _____ г.

Рязанцев Е В

(подпись) *(Ф.И.О.)*