

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к автоматизированной системе

Вариант № 7 «Экологический фонд»

Составил:

студент гр. ИСТд-31

Желепов Алексей Сергеевич

«21» мая 2014 г.

Проверил:

доцент каф. ИВК, к.т.н., доцент

Родионов Виктор Викторович

« » 2014 г.

Ульяновск, 2014

Содержание

Список использованных сокращений и обозначений	4
Введение	5
1 Техническое задание.....	8
1.1 Общие сведения	8
2.1 Назначение и цели создания системы.....	8
2.2.1 Назначение системы	8
2.2.1 Цели создания системы	8
1.3 Характеристика объекта автоматизации	9
1.4 Требования к системе	9
1.4.1 Требования к системе в целом	9
1.4.2 Требования к функциям, выполняемым системой	10
1.4.3 Требования к видам обеспечения	11
1.5 Состав и содержание работ по созданию системы.....	12
1.6 Порядок контроля и приёмки системы.....	12
1.7 Требования к документированию	12
2 Информационное обеспечение системы.....	13
2.1.1 Диаграмма «сущность-связь»	13
2.1.2 Сущности и их атрибуты.....	13
2.1.3 Связи между сущностями	14
2.1.4 Инструментальное средство моделирования	17
2.2 Даталогическая модель предметной области.....	18
2.2.1 Анализ инфологической модели	18
2.2.1 База данных системы	19
3 Математическое обеспечение системы.....	32
3.1 Регистрация пользователя системы	32
3.2 Авторизация пользователя системы	33
3.3 Поиск информации об экологе	34
3.4 Фильтрация информационных данных об организациях/лицах-штрафниках	35
4 Прикладное программное обеспечение системы.....	37
4.1 Общая характеристика прикладного программного обеспечения.....	37
4.1.1. Оценка соответствия разработанного web-приложения общим требованиям	37
4.1.2. Оценка соответствия разработанного web-приложения дополнительным требованиям	38
4.2 Структура и состав прикладного программного обеспечения.....	39
4.2.1 Основные директории проекта приложения в VS 2012 Ultimate	39
4.2.2 Контроллеры web-приложения.....	41
4.2.3 Модели web-приложения	44
4.2.4 Контекст данных web-приложения	46
4.2.5 Представления web-приложения	46
4.3 Особенности реализации и сопровождения.....	53
5 Руководство пользователя.....	55
5.1 Общие сведения	55
5.2 Порядок и особенности работы	55
5.2.1 Описание основных частей интерфейса системы.....	55
5.2.2 Неавторизованный пользователь системы	57

5.2.3 Представитель организации-партнера	62
5.2.4 Администратор.....	63
5.2.5 Секретарь	70
5.2.6 Эколог-специалист.....	76
5.2.7 Авторизованный пользователь системы.....	79
5.3 Исключительные ситуации	80
Заключение	84
Список использованных источников	85
Приложение А. Исходные тексты программных модулей	86

Список использованных сокращений и обозначений

1НФ – первая нормальная форма

2НФ – вторая нормальная форма

3НФ – третья нормальная форма

НФБК – нормальная форма Бойса-Кодда

Введение

Экологические фонды – это внебюджетные организации, занимающиеся сбором и распределением средств на решения экологических проблем.

Источниками поступающих в фонд средств являются средства предприятий в виде платы за нормативные и сверхнормативные выбросы загрязняющих веществ в окружающую среду, также штрафы, которые могут взиматься с браконьеров и прочих лиц, подвергающих опасности дикую живую природу.

Размеры платы за вред, наносимый окружающей природной среде, установлены постановлением Совета Министров - Правительства Российской Федерации от 28.08.92 № 632 «Об утверждении Порядка определения платы и ее предельных размеров за загрязнение окружающей природной среды, размещение отходов, другие виды вредного воздействия», а также в соответствии с другими инструктивно-нормативными документами; сумм, полученных по искам о возмещении вреда, и штрафов за экологические правонарушения.

Штрафы налагаются в пределах компетенции специально уполномоченными государственными органами Российской Федерации, органами в области охраны природы, санитарно-эпидемиологического надзора и другими.

Конкретный размер штрафа определяется органом, налагающим штраф, в зависимости от характера и вида совершенного правонарушения, степени вины правонарушителя и причиненного вреда.

Другой группой средств, поступающих на деятельность фонда, являются пожертвования граждан и организаций-партнеров экологического фонда. В современных реалиях денежные переводы осуществляются при помощи информационных систем-сайтов, позволяющих пользователю переводить деньги со своего электронного кошелька.

Помимо сбора средств на решения экологических проблем, фонд обычно занимается многими другими видами деятельности, среди которых:

1. мониторинг окружающей среды на вид экологических нарушений;
2. сотрудничество с организациями-партнерами и другими фондами;

3. сохранение базы экологических нарушений и их решений;
4. проведение пропаганды среди молодежи о защите окружающей среды и важности ее сохранения;
5. проведение собраний специалистов-экологов для решения возникающих экологических проблем.

Мониторинг окружающей среды осуществляется экологами-специалистами, которые имеют договоренности о сотрудничестве с фондом или являются его работниками. Как правило, многие из них расквартированы в различных уголках страны. Это позволяет незамедлительно реагировать на поступающие жалобы об экологических нарушениях, проводить исследование «на месте» и поставлять точные данные организации-фонду.

Благодаря сотрудничеству с другими организациями, география действия фонда значительно расширяется, что делает работу экологической организации более эффективной.

На основе поступающей информации об экологических нарушениях формируются и назначаются советы экологов-специалистов, на которых выносятся важные решения по поводу хода решения экологических проблем, выставленных на обсуждение на текущем совещании.

Велика роль экологических фондов и в образовательной сфере. Часто специалисты организации назначаются на ведение экологических кружков в школах и ВУЗах по защите и сохранению окружающей среды.

Автором курсовой работы было рассмотрено множество информационных источников, которые помогли ему спроектировать и продумать механику работы системы. Например, на сайте бизнес-словаря «Ведомости»[1] очень подробно изложена информация об основных источниках поступления средств в экологический фонд. На сайте фонда имени В.И. Вернадского[2] приводится информация о том, что фонд – не только экономическая организация, но и социальная, которая активно взаимодействует с обществом, проводя множество экологических занятий и семинаров в различных государственных учреждениях (школах, колледжах, университетах).

Помимо источников, касающихся характеристик предметной области, автором курсовой работы было использовано много статей и книг (в том числе опубликованных на английском языке) о технологии программирования ASP.NET MVC 4 и технологии создания и развертывания баз данных EntityFramework 5. Наиболее полезным источником оказались книги [3], [4], [5], в которых описаны основные подходы разработки MVC-приложений. Также автор курсовой работы активно использовал сайт MSDN, где были найдены ответы на возникающие по ходу разработки вопросы.

1 Техническое задание

1.1 Общие сведения

Наименование разрабатываемой системы – Автоматизированная система управления экологическим фондом “Слон” (далее система).

2.1 Назначение и цели создания системы

2.2.1 Назначение системы

Разрабатываемая система предназначена для автоматизации процессов решения проблем экологии дикой природы. Система оказывает поддержку широкого круга пользователей, заинтересованных в поддержке автоматизированной системы, среди них: пользователи, люди, интересующиеся состоянием окружающей среды; экологи, пополняющие базу данных экологического фонда сведениями о нерешенных проблемах; секретарь, рассматривающий заявки организаций на сотрудничество, определяющий время проведения экологических советов по решению проблем и контролирующий выплаты штрафов со стороны нарушителей экологических прав; организации-партнеры, сообщаящие фонду об известных им экологических проблемах; администратор, ответственный за работу системы в целом.

2.2.1 Цели создания системы

В результате применения разрабатываемой системы управления будут достигнуты следующие цели:

1. Упрощение сбора информации о проблемах экологии окружающей среды при внедрении системы;
2. Проведение рекламы работы фонда, что должно увеличить количество организаций-партнеров и пользователей при размещении информации о достижениях экологического фонда;
3. Популяризация среди пользователей темы экологического воспитания и возможность электронной регистрации на экологические кружки, преподавателями которых будут специалисты-экологи.
4. Облегчение хранения протоколов и решений с экологических собраний.

1.3 Характеристика объекта автоматизации

Экологический фонд является внебюджетным фондом, средства которого направляются на решение экологических проблем окружающей среды. Учредителями экологических фондов являются краевые, областные и республиканские комитеты по экологии и природоведению. Организации, как правило, являются самостоятельными юридическими лицами, имеют собственный баланс. Основными задачами работы экологических фондов являются проведение мероприятий и разработка программ по следующим направлениям:

1. создание информационной системы сбора, хранения, систематизации и обработки экологической информации;
2. проведение мероприятий с целью решения экологических проблем;
3. ведение базы сбора информации об экологических нарушениях. Нарушение может быть как со стороны организации (сброс химически опасных веществ заводом в реку), так и со стороны физического лица (браконьерство);
4. организация экологического образования и воспитания, пропаганда экологических знаний.

Экологический фонд является неотъемлемой частью механизма регулирования природопользования и образуется за счет поступлений средств от предприятий и физических лиц нарушителей. Таким образом, в основном денежный баланс фонда формируется из следующих платежей:

1. штрафы за загрязнение окружающей среды;
2. сверхнормативное использование природных ресурсов;
3. штрафы за нарушение природоохранного законодательства.

1.4 Требования к системе

1.4.1 Требования к системе в целом

1.4.1.1 Требования к структуре и функционированию системы

Определяется общей постановкой задачи задания на курсовую работу

1.4.1.2 Требования к защите информации от несанкционированного доступа

Защита информации от несанкционированного доступа является важной особенностью при разработке подобной системы, так как в этой системе имеются операции с банковскими счетами, переводами денежных средств и личными данными зарегистрированных пользователей. Утечка таких данных может повлечь за собой ухудшение взаимоотношений между фондом и его партнерами, а также тривиальную кражу денежных средств.

Основными мерами по сохранению данных от несанкционированного доступа к данным являются стандартные уровни защиты SQL Server, среди которых выделяются:

1. хранение строки подключения к базе данных в защищенном конфигурационном файле;
2. использование аутентификации Windows при работе с SQL-сервером базы данных.
3. использование Code First подхода технологии Entity Framework, позволяющего скрыть данные методом инкапсуляции объектно-ориентированного программирования.

1.4.2 Требования к функциям, выполняемым системой

Неавторизованный пользователь системы имеет следующие возможности:

1. просмотра общей информации об экологическом фонде, а также сведений об организациях-партнерах;
2. просмотра списка текущих экологических проблем, взятых в разработку фондом, и информация о них;
3. просмотра достижений экологического фонда;
4. просмотра информации об экологических кружках.

Представитель организации-партнера имеет следующую возможность:

1. добавления записи-жалобы на экологическое нарушение.

Рядовой пользователь системы имеет следующие возможности:

1. добавления записи-жалобы на экологическое нарушение;
2. электронной регистрации на посещение экологических кружков.

Эколог имеет следующие возможности при пользовании системой:

1. добавления записи о новых экологических проблемах на основе проводимого экологом мониторинга;
2. регистрации своего участия в организованных фондом экологических собраниях по отслеживанию хода решения определенной экологической проблемы;
3. рассмотрения жалоб пользователей системы на экологические проблемы и принятия решения по включению их в соответствующий список или их отклонение.

Секретарь имеет возможность доступа к следующим операциям системы:

1. рассмотрение заявок организаций на сотрудничество с экологическим фондом;
2. объявления информации по проведению экологических советов;
3. контроль выплаты штрафов со стороны организаций/лиц-штрафников.

Администратор системы имеет следующие возможности:

1. добавление/удаление учетных записей пользователей системы;
2. редактирование и добавление информации о достижениях фонда по решению экологических проблем;
3. создание записей о новых экологических кружках.

Следует отметить, что все авторизованные пользователи также могут просматривать информацию, как и неавторизованные.

1.4.3 Требования к видам обеспечения

1.4.3.1 Требования к техническому обеспечению

Материнская плата – Intel HM77 Express

Процессор – Intel Core i5-3120M, 2500 NHz

Оперативная память – DDR3-1333 Memory, 6 Гб

Жесткий диск – HDD Sata, 500 Гб.

Видеокарта – NVIDIA GeForce 710M, 2Гб

Звуковая карта – Intel High Definition Audio

Монитор – Acer Aspire V3-771G-53216G75Maii Display 17.3”.

Оптический привод – Acer Aspire V3-771G-53216G75Maii DVD-RW

1.4.3.2 Требования к программному обеспечению

При разработке системы будет использоваться операционная система семейства Windows версии 8.1.

При создании диаграммы “сущность-связь” будет использоваться программа Erwin Data Modeler.

1.5 Состав и содержание работ по созданию системы

Определяется этапами выполнения работы задания на курсовую работу.

1.6 Порядок контроля и приёмки системы

Определяется порядком защиты и критериями оценки работы задания на курсовую работу.

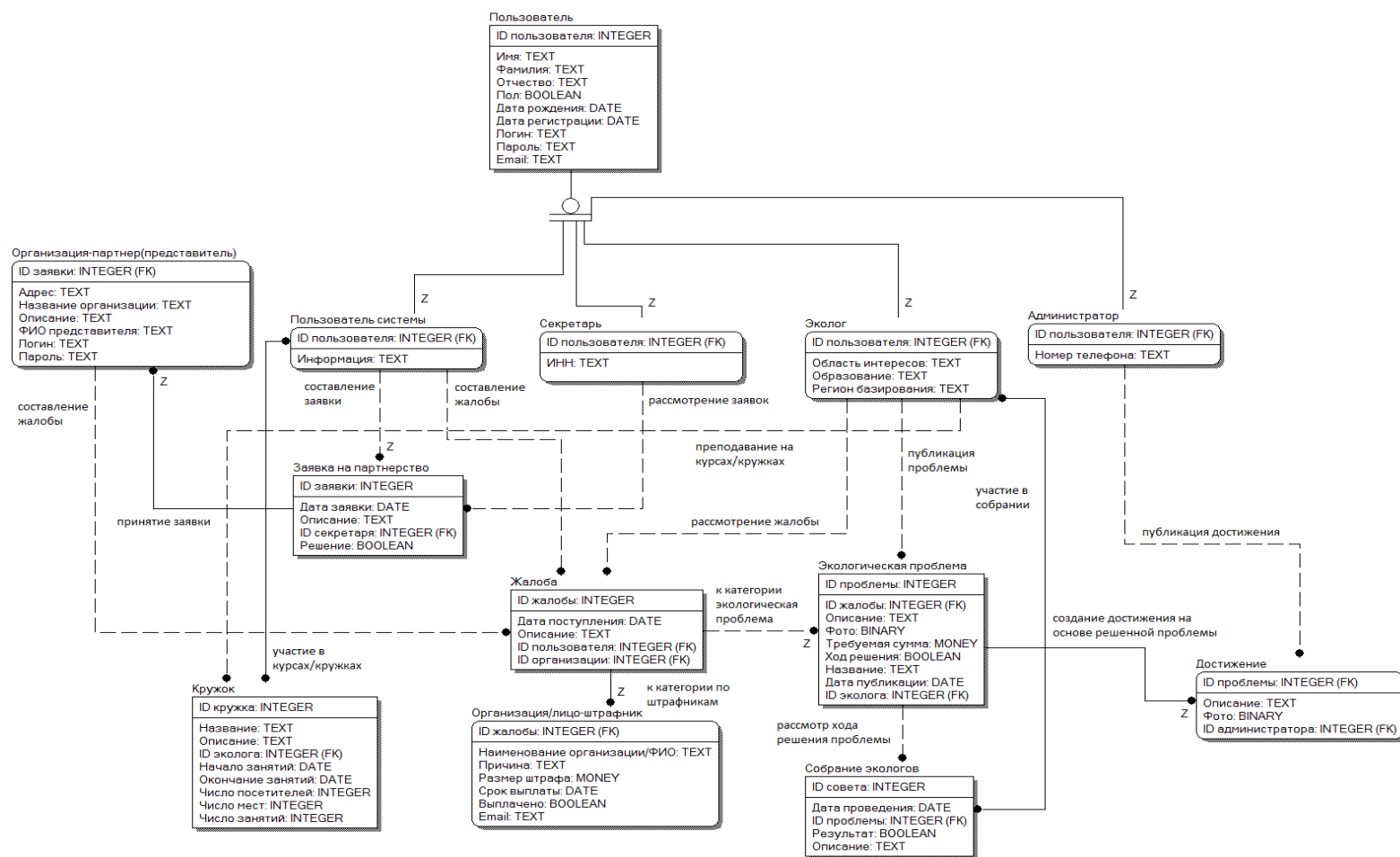
1.7 Требования к документированию

Ни один компонент из пояснительной записки удален не будет.

2 Информационное обеспечение системы

2.1 Инфологическая модель предметной области

2.1.1 Диаграмма «сущность-связь»



2.1.2 Сущности и их атрибуты

Сущность **«Пользователь»** – содержит в себе общую информацию обо всех пользователях системы.

Сущность **«Пользователь системы»** – описывает рядового пользователя системы.

Сущность **«Секретарь»** – описывает работника фонда, занимающегося рассмотрением заявок на сотрудничество и объявлением информации по проведению экологических советов.

Сущность **«Организация-партнер (представитель)»** – приводится описание организации, которая сотрудничает с экологическим фондом «Слон».

Сущность **«Эколог»** – описывает специалистов, занятых поиском и фиксированием в системе экологических проблем на территории.

Сущность **«Администратор»** – описывает сотрудников, занимающихся сопровождением и технической поддержкой системы.

Сущность **«Заявка на партнерство»** – описывает заявки, принятые от пользователей (представителей организаций), желающих заключить партнерские отношения с фондом.

Сущность **«Жалоба»** – описывает жалобы по нарушению экологических прав и законов, принятые от рядовых пользователей системы и представителей организаций-партнеров. Атрибут *ID пользователя* заполняется в случае указании причины пользователем системы, при этом атрибут *ID организации* не заполняется. Возможна и обратная ситуация.

Сущность **«Организация/лицо-штрафник»** – включает в себя записи о тех организациях и людях, которые нарушили экологические права и законы. Атрибут *Ход решения* показывает, была ли произведена выплата штрафа. Атрибут *Причина* содержит в себе пояснение, написанное экологом, о причинах введения штрафных санкций по отношению к организации/лицу-штрафнику.

Сущность **«Экологическая проблема»** – описывает экологическую проблему, обнаруженную экологом в результате мониторинга, или обозначенную

на основе жалобы, выраженной пользователем системы. Атрибут *Ход решения* показывает, была ли решена проблема. Если ответ на этот вопрос положительный – это сигнал администратору системы опубликовать отзыв о ней и добавить его в **Достижения** фонда.

Сущность **«Собрание экологов»** – описывает встречу экологов по решению определенной экологической проблемы. Атрибут *Результат* показывает, какое было принято решение по результатам собрания и были ли проведены соответствующие меры со стороны фонда по решению проблемы.

Сущность **«Достижения»** – включает в себя записи о тех экологических проблемах, которые фонду удалось решить.

Сущность **«Кружок»** – описывает всю необходимую информацию о кружках и секциях, посвященных экологическому воспитанию. Атрибут *ID эколога* определяет эколога-преподавателя, который будет вести занятия. Атрибут *Число мест* показывает максимальное количество людей, которое может быть включено в группу на данный курс.

2.1.3 Связи между сущностями

Сущности **«Пользователь»** и **«Пользователь системы»** соединены связью типа «Есть».

Сущности **«Пользователь»** и **«Секретарь»** соединены связью типа «Есть».

Сущности **«Пользователь»** и **«Эколог»** соединены связью типа «Есть».

Сущности **«Пользователь»** и **«Администратор»** соединены связью типа «Есть».

Сущности **«Заявка на партнерство»** и **«Организация-партнер (представитель)»** соединены связью, которая идентифицирует отношение, тип связи – «1,1:0,1». Каждая новая организация-партнер должна была оставить заявку на сотрудничество, каждой заявке соответствует одна организация-партнер или не соответствует в случае отклонения заявки секретарем.

Сущности **«Организация-партнер»** и **«Жалоба»** соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». У жалобы может быть только один инициатор (в данном случае представитель организации-партнера), в то время как представитель компании может давать неограниченное число жалоб.

Сущности **«Секретарь»** и **«Заявка»** соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». Конкретную заявку может рассматривать только один секретарь, в то время как секретарь может рассматривать множество заявок.

Сущности **«Пользователь системы»** и **«Заявка на партнерство»** соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,1». Заявка составляется одним пользователем системы, в то время как пользователь может составить либо одну заявку для секретаря, либо ни одной.

Сущности **«Пользователь системы»** и **«Кружок»** соединены связью, которая не идентифицирует отношение, тип связи – «0,N:0,M». Это означает, что пользователь может регистрироваться на множество кружков, а на кружок может быть зарегистрировано множество пользователей.

Сущности **«Пользователь системы»** и **«Жалоба»** соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0:N». Жалоба может быть написана одним пользователем, пользователь может писать множество жалоб.

Сущности **«Эколог»** и **«Жалоба»** соединены связью, не идентифицирующей отношение, тип связи – «1,1:0:N». Конкретная жалоба может рассматриваться только одним экологом, в то время как специалист может рассматривать множество жалоб.

Сущности **«Эколог»** и **«Экологическая проблема»** соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». Это означает, что проблема может быть опубликована одним экологом. В свою очередь, эко-

лог может записывать информацию о неограниченном количестве экологических проблем.

Сущности «**Эколог**» и «**Собрание экологов**» соединены связью, которая не идентифицирует отношение, тип связи – «0,N:0,M». Эколог может входить во множество рабочих групп по разрешению конкретных проблем, так и собрание экологов может состоять из нескольких специалистов.

Сущности «**Эколог**» и «**Кружок**» соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». Кружок ведется одним преподавателем, в то время как эколог может преподавать на нескольких кружках одновременно.

Сущности «**Жалоба**» и «**Организация/лицо-штрафник**» соединены связью, которая идентифицирует отношение, тип связи – «1,1:0:1». Организация/лицо появляются в списке только, если на нее/него была подана жалоба. При этом жалоба может быть написана на организацию/лицо или же такой жалобы может и не существовать.

Сущности «**Экологическая проблема**» и «**Собрание экологов**» соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». Группа экологов в течение заседания пытается разрешить взятую конкретную проблему. Проблема может рассматриваться на нескольких собраниях.

Сущности «**Экологическая проблема**» и «**Достижение**» соединены связью, которая идентифицирует отношение, тип связи – «1,1:0,1». Решенной проблеме соответствует либо одна публикация о достижении фонда или ни одной публикации. Одной публикации соответствует только одна решенная проблема.

Сущности «**Администратор**» и «**Достижение**» соединены связью, которая не идентифицирует отношение, тип связи – «1,1:0,N». Статья о решенной проблеме может быть опубликована только одним администратором, а каждый администратор может опубликовать множество статей.

2.1.4 Инструментальное средство моделирования

При разработке инфологической модели базы данных были рассмотрены следующие программные продукты: Erwin Data Modeler, ERConstructor и Microsoft Office Visio. На основании проведенного анализа (табл. 1) программного обеспечения была выбрана программа Erwin Data Modeler. Ключевыми критериями выбора программного пакета Erwin Data Modeler стали первый и четвертый критерии оценки.

Таблица 2.1.4.1.

«Сравнение программного обеспечения для построения инфологических моделей»

Критерий сравнения	ERConstructor	Erwin Data Modeler	Visio
Простота использования	Достаточно просто использовать	Достаточно просто использовать	Достаточно просто использовать
Быстрота создания модели	Быстро	Быстро	Быстро
Стоимость ПО	Бесплатное ПО	Платное ПО	Необходимо приобретать лицензию на использование
Опыт использования ПО на момент начала курсового проектирования	-	+	+

2.2 Даталогическая модель предметной области

2.2.1 Анализ инфологической модели

В представленной инфологической модели существуют следующие типы связи:

1. Связь типа «Есть». Например, между сущностями «Пользователь» и «Секретарь».
2. Связь типа 0,N:0,M (Многие-ко-многим). Например, данный тип используется между сущностями «Пользователь системы» и «Экологическая секция».
3. Связь типа 1,1:0,N. Например, сущности «Эколог» и «Экологическая секция» соединены таким типом связи.
4. Связь типа 1,1:0,1 (Один-к-одному). Например, тип используется между сущностями «Жалоба» и «Организация/лицо-штрафник».

Наиболее сложной для реализации оказалась связь типа Многие-ко-многим. Для ее объявления необходимо создание третьей таблицы, содержащей пары значений соответствующих идентификаторов. На примере связи между сущностями «Пользователь системы» и «Экологическая секция» это будет выглядеть следующим образом: «Идентификатор пользователя, Идентификатор секции».

Еще одна сложность создания связи Многие-ко-многим состояла в том, что при удалении одной из записей, необходимо было определять: применять каскадное удаление записей или разрушать только связь с удалением одной из записей. При реализации был выбран второй вариант. Этот выбор объясняется требованиями к системе.

Связи типа 1,1:0,N или 1,1:0,1 между сущностями не вызвали затруднений, так как создаются две отдельные таблицы, связанные между собой по идентифицирующему столбцу.

Связь типа «Есть» была реализована при помощи связи типа 1,1:0,1, что является логичным, так как запись из первой таблицы соответствует только одной записи из второй.

2.2.1 База данных системы

2.2.1.1 Таблица [dbo].[Users]

```
CREATE TABLE [dbo].[Users](
    [ID] [int] IDENTITY(1,1) NOT NULL,
    [Name] [nvarchar](max) NOT NULL,
    [Surname] [nvarchar](max) NOT NULL,
    [FatherName] [nvarchar](max) NOT NULL,
    [Sex] [bit] NOT NULL,
    [BirthDate] [datetime] NOT NULL,
    [Login] [nvarchar](max) NOT NULL,
    [Password] [nvarchar](max) NOT NULL,
    [Email] [nvarchar](max) NOT NULL,
    [RegistrationDate] [datetime] NOT NULL,
    [Address] [nvarchar](max) NULL,
    [CompanyName] [nvarchar](max) NULL,
    [Description] [nvarchar](max) NULL,
    [Reason] [nvarchar](max) NULL,
    [IsSolved] [bit] NULL,
    [Discriminator] [nvarchar](128) NULL,
    [Secretary_ID] [int] NULL,
    CONSTRAINT [PK_dbo.Users] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.2 Таблица [dbo].[Secretaries]

```
CREATE TABLE [dbo].[Secretaries](
    [ID] [int] NOT NULL,
    [IndividualTaxNumber] [nvarchar](max) NULL,
    CONSTRAINT [PK_dbo.Secretaries] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.3 Таблица [dbo].[RankUsers]

```
CREATE TABLE [dbo].[RankUsers](
    [ID] [int] NOT NULL,
    [Information] [nvarchar](max) NULL,
    CONSTRAINT [PK_dbo.RankUsers] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.4 Таблица [dbo].[Administrators]

```
CREATE TABLE [dbo].[Administrators](
    [ID] [int] NOT NULL,
    [PhoneNumber] [nvarchar](max) NULL,
    CONSTRAINT [PK_dbo.Administrators] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.5 Таблица [dbo].[Ecologists]

```
CREATE TABLE [dbo].[Ecologists](
    [ID] [int] NOT NULL,
    [Education] [nvarchar](max) NOT NULL,
    [InterestsSphere] [nvarchar](max) NULL,
    [DistrictLocation] [nvarchar](max) NULL,
    CONSTRAINT [PK_dbo.Ecologists] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.6 Таблица [dbo].[Partners]

```
CREATE TABLE [dbo].[Partners](
    [ID] [int] NOT NULL,
    [Secretary_ID] [int] NOT NULL,
    [Address] [nvarchar](max) NOT NULL,
    [CompanyName] [nvarchar](max) NOT NULL,
    [Description] [nvarchar](max) NOT NULL,
    [Reason] [nvarchar](max) NOT NULL,
    [IsSolved] [bit] NOT NULL,
    CONSTRAINT [PK_dbo.Partners] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет первичный ключ ID, который является единственным детерминантом.

2.2.1.7 Таблица [dbo].[EcologicalProblems]

```
CREATE TABLE [dbo].[EcologicalProblems](
    [ProblemID] [int] NOT NULL,
    [Description] [nvarchar](max) NOT NULL,
    [PhotoType] [nvarchar](max) NULL,
    [PhotoFile] [varbinary](max) NULL,
    [RequiredSum] [decimal](18, 2) NOT NULL,
    [IsSolved] [bit] NOT NULL,
    [Title] [nvarchar](max) NOT NULL,
    [PublicationDate] [datetime] NOT NULL,
    [Creator_ID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.EcologicalProblems] PRIMARY KEY CLUSTERED
(
    [ProblemID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ProblemID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Title, Creator_ID) являющийся ключом-кандидатом.

2.2.1.8 Таблица [dbo].[Councils]

```
CREATE TABLE [dbo].[Councils](
    [CouncilID] [int] IDENTITY(1,1) NOT NULL,
    [Title] [nvarchar](max) NOT NULL,
    [AssignmentDate] [datetime] NOT NULL,
    [CouncilResult] [bit] NOT NULL,
    [Description] [nvarchar](max) NOT NULL,
    [Problem_ProblemID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.Councils] PRIMARY KEY CLUSTERED
(
    [CouncilID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```


Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ детерминант (Title, Problem_ProblemID) является ключом-кандидатом.

2.2.1.9 Таблица [dbo].[Complaints]

```
CREATE TABLE [dbo].[Complaints](
    [ComplaintID] [int] IDENTITY(1,1) NOT NULL,
    [AppearingDate] [datetime] NOT NULL,
    [Description] [nvarchar](max) NOT NULL,
    [Title] [nvarchar](max) NOT NULL,
    [IsHidden] [bit] NOT NULL,
    [Creator_ID] [int] NOT NULL,
    [Partner_ID] [int] NULL,
    [Ecologist_ID] [int] NULL,
    CONSTRAINT [PK_dbo.Complaints] PRIMARY KEY CLUSTERED
(
    [ComplaintID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута ComplaintID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Title, Creator_ID), который является ключом-кандидатом.

2.2.1.10 Таблица [dbo].[Achievements]

```
CREATE TABLE [dbo].[Achievements](
    [AchievementID] [int] NOT NULL,
    [Description] [nvarchar](max) NULL,
    [PhotoType] [nvarchar](max) NULL,
    [PhotoFile] [varbinary](max) NULL,
    [Title] [nvarchar](max) NOT NULL,
    [Administrator_ID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.Achievements] PRIMARY KEY CLUSTERED
(
    [AchievementID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута AchievementID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Title, Administrator_ID), который является ключом-кандидатом.

2.2.1.11 Таблица [dbo].[OrganisationDebtors]

```
CREATE TABLE [dbo].[OrganisationDebtors](
    [OrganisationDebtorID] [int] NOT NULL,
    [Name] [nvarchar](max) NOT NULL,
    [Reason] [nvarchar](max) NOT NULL,
    [FineAmount] [decimal](18, 2) NOT NULL,
    [PayTime] [datetime] NOT NULL,
    [IsPayed] [bit] NOT NULL,
    [Email] [nvarchar](max) NULL,
    [ResponsiblePerson_ID] [int] NULL,
    CONSTRAINT [PK_dbo.OrganisationDebtors] PRIMARY KEY CLUSTERED
(
    [OrganisationDebtorID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута OrganisationDebtorID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант OrganisationDebtorID, который является первичным ключом таблицы.

2.2.1.12 Таблица [dbo].[Sections]

```
CREATE TABLE [dbo].[Sections](
    [SectionID] [int] IDENTITY(1,1) NOT NULL,
    [Title] [nvarchar](max) NOT NULL,
    [Description] [nvarchar](max) NOT NULL,
    [StartLessonsTime] [datetime] NOT NULL,
    [ParticipantsCount] [int] NOT NULL,
    [SpotsCount] [int] NOT NULL,
    [FreeSpotsCount] [int] NOT NULL,
    [LessonsCount] [int] NOT NULL,
    [Ecologist_ID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.Sections] PRIMARY KEY CLUSTERED
(
    [SectionID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута Section_ID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа Section_ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Title, Ecologist_ID), который является ключом-кандидатом.

2.2.1.13 Таблица [dbo].[PartnershipRequests]

```
CREATE TABLE [dbo].[PartnershipRequests](
    [RequestID] [int] IDENTITY(1,1) NOT NULL,
    [Reason] [nvarchar](max) NULL,
    [IsAccepted] [bit] NOT NULL,
    CONSTRAINT [PK_dbo.PartnershipRequests] PRIMARY KEY CLUSTERED
(
    [RequestID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута RequestID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей, так как все атрибуты полностью и функционально зависят от ключа Request_ID.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант RequestID, который является первичным ключом таблицы.

2.2.1.14 Таблица [dbo].[SectionRankUsers]

```
CREATE TABLE [dbo].[SectionRankUsers](
    [Section_SectionID] [int] NOT NULL,
    [RankUser_ID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.SectionRankUsers] PRIMARY KEY CLUSTERED
(
    [Section_SectionID] ASC,
    [RankUser_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута Section_SectionID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Section_SectionID, RankUser_ID), который является первичным составным ключом таблицы.

2.2.1.15 Таблица [dbo].[CouncilEcologists]

```
CREATE TABLE [dbo].[CouncilEcologists](
    [Council_CouncilID] [int] NOT NULL,
    [Ecologist_ID] [int] NOT NULL,
    CONSTRAINT [PK_dbo.CouncilEcologists] PRIMARY KEY CLUSTERED
(
    [Council_CouncilID] ASC,
    [Ecologist_ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

Таблица находится в 1НФ, так как содержит только простые и однозначные атрибуты.

Таблица находится во 2НФ, так как она находится в 1НФ и каждый неключевой атрибут функционально зависит только от атрибута Council_CouncilID, который является ключом таблицы.

Таблица находится в 3НФ, так как находится в 2НФ и не содержит транзитивных зависимостей.

Таблица находится в НФБК, так как находится в 3НФ и имеет детерминант (Council_CouncilID, Ecologist_ID), который является составным первичным ключом таблицы.

2.2.1.16 Хранимая процедура [dbo].[GetCrucialDebtor] с использованием транзакции

```
CREATE PROCEDURE [dbo].[GetCrucialDebtor]
```

```

        @day_count int
    AS
    BEGIN
        BEGIN TRANSACTION;
        SET NOCOUNT ON;
        SELECT * FROM [dbo].[OrganisationDebtors] WHERE (PayTime
<= DATEADD(DAY, @day_count, GETDATE())) AND IsPaid = 0)
        COMMIT TRANSACTION;
    END

```

Примечание: Данная процедура используется секретарем системы для получения списка организаций/лиц-должников, срок выплаты штрафных санкций которых, наступает в течение @day_count дней.

2.2.1.17 Хранимая процедура [dbo].[GetEcologicalProblems] с использованием транзакции

```

CREATE PROCEDURE [dbo].[GetEcologicalProblems]
    @days_range int
AS
BEGIN
    BEGIN TRANSACTION;
    SET NOCOUNT ON;
    SELECT * FROM [dbo].[EcologicalProblems] WHERE
(PublicationDate >= DATEADD(DAY, -@days_range, GETDATE()))
    COMMIT TRANSACTION;
END

```

Примечание: Данная процедура используется секретарем системы для получения списка экологических проблем, которые были опубликованы в интервале @days_range числа дней.

2.2.1.18 Функция [dbo].[GetAdultsQuantity]

```

CREATE FUNCTION [dbo].[GetAdultsQuantity]
(
)
RETURNS int
AS
BEGIN
    DECLARE @adults int;
    SELECT @adults = COUNT(*)
    FROM [dbo].[Users] WHERE BirthDate <= DATEADD(YEAR, -
18, GETDATE())
    RETURN @adults
END

```

Примечание: Данная функция возвращает скалярное значение числа совершеннолетних пользователей.

2.2.1.19 Функция [dbo].[GetChildrenQuantity]

```
CREATE FUNCTION [dbo].[GetChildrenQuantity]
(
)
RETURNS int
AS
BEGIN
    DECLARE @children int;
    SELECT @children = COUNT(*)
    FROM [dbo].[Users] WHERE BirthDate >= DATEADD(YEAR, -
18, GETDATE())
    RETURN @children
END
```

Примечание: Данная функция возвращает скалярное значение числа несовершеннолетних пользователей.

2.2.1.20 Триггер [dbo].[CalculateSpotsForInsert]

```
CREATE TRIGGER [dbo].[CalculateSpotsForInsert]
ON [dbo].[SectionRankUsers]
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE dbo.Sections SET ParticipantsCount = (SELECT
COUNT(*) FROM [dbo].[SectionRankUsers] WHERE [dbo].[SectionRankUsers].Section_SectionID =
SectionId) WHERE SectionId IN (SELECT i.Section_SectionId FROM inserted AS i)
    UPDATE dbo.Sections SET FreeSpotsCount = SpotsCount -
(SELECT COUNT(*) FROM [dbo].[SectionRankUsers] WHERE
[dbo].[SectionRankUsers].Section_SectionID = SectionId) WHERE SectionId IN (SELECT
i.Section_SectionId FROM inserted AS i)
END
```

Примечание: Данный триггер предназначен для пересчета числа свободных мест и числа посетителей экологической секции после регистрации нового участника.

2.2.1.21 Триггер [dbo].[CalculateSpotsForDelete]

```
CREATE TRIGGER [dbo].[CalculateSpotsForDelete]
ON [dbo].[SectionRankUsers]
AFTER DELETE
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE dbo.Sections SET ParticipantsCount = (SELECT
COUNT(*) FROM [dbo].[SectionRankUsers] WHERE [dbo].[SectionRankUsers].Section_SectionID =
SectionId) WHERE SectionId IN (SELECT i.Section_SectionId FROM deleted AS i)
```

```
UPDATE dbo.Sections SET FreeSpotsCount = SpotsCount -  
(SELECT COUNT(*) FROM [dbo].[SectionRankUsers] WHERE  
[dbo].[SectionRankUsers].Section_SectionID = SectionId) WHERE SectionId IN (SELECT  
i.Section_SectionId FROM deleted AS i)  
END
```

Примечание: Данный триггер предназначен для пересчета числа свободных мест и числа посетителей экологической секции после deregистрации одного из участников.

3 Математическое обеспечение системы

В этой части пояснительной записки приводится описание основных алгоритмов.

3.1 Регистрация пользователя системы

1. Общая характеристика: Алгоритм регистрации и добавления сведений о новом зарегистрированном пользователе в базу данных.

2. Используемые данные: Таблицы [dbo].[RankUsers], [dbo].[Partners] и [dbo].[Ecologists].

3. Результаты выполнения: Добавление записи о новом пользователе в одну из таблиц (в зависимости, от того какой тип регистрации выбрал пользователь): [dbo].[RankUsers], [dbo].[Partners] или [dbo].[Ecologists]. В случае успешного прохождения регистрации осуществляется переход на страницу с соответствующим сообщением, иначе - указание пользователю всех совершенных им ошибок в ходе регистрации.

4. Математическое описание: -.

5. Логическое описание: Блок-схема алгоритма регистрации пользователя приведена на рисунке 3.1.1.

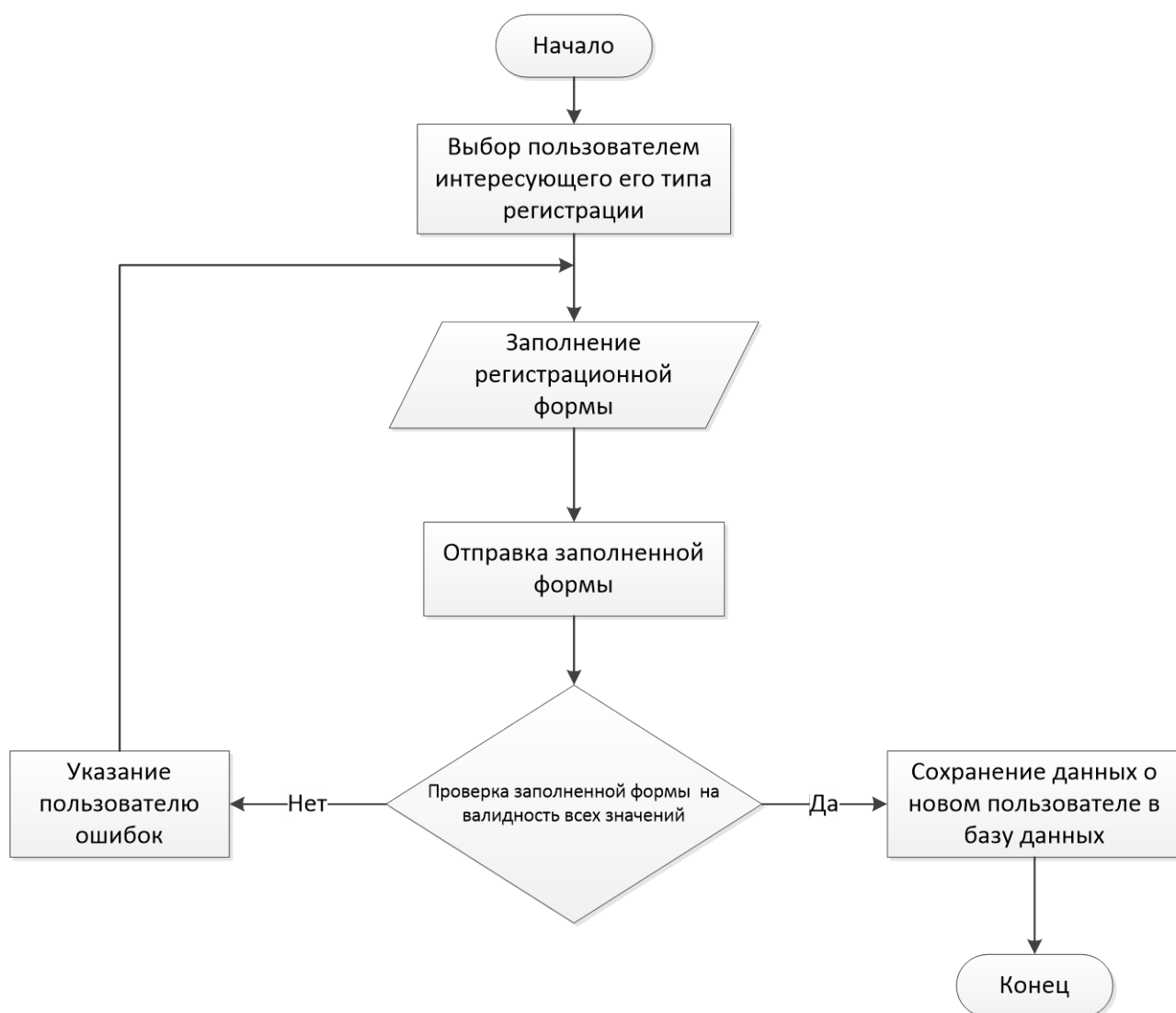


Рис. 3.1.1. Блок-схема алгоритма регистрации пользователя

3.2 Авторизация пользователя системы

1. Общая характеристика: Алгоритм авторизации пользователя системы предназначен для определения его роли в системе с целью открытия определенного дополнительного функционала.

2. Используемые данные: Таблицы [dbo].[RankUsers], [dbo].[Partners] и [dbo].[Ecologists], [dbo].[Administrators], [dbo].[Secretaries].

3. Результаты выполнения: Открытие дополнительных возможностей при пользовании системой.

4. Математическое описание: -.

5. Логическое описание: Блок-схема алгоритма авторизации пользователя приведена на рисунке 3.2.1.

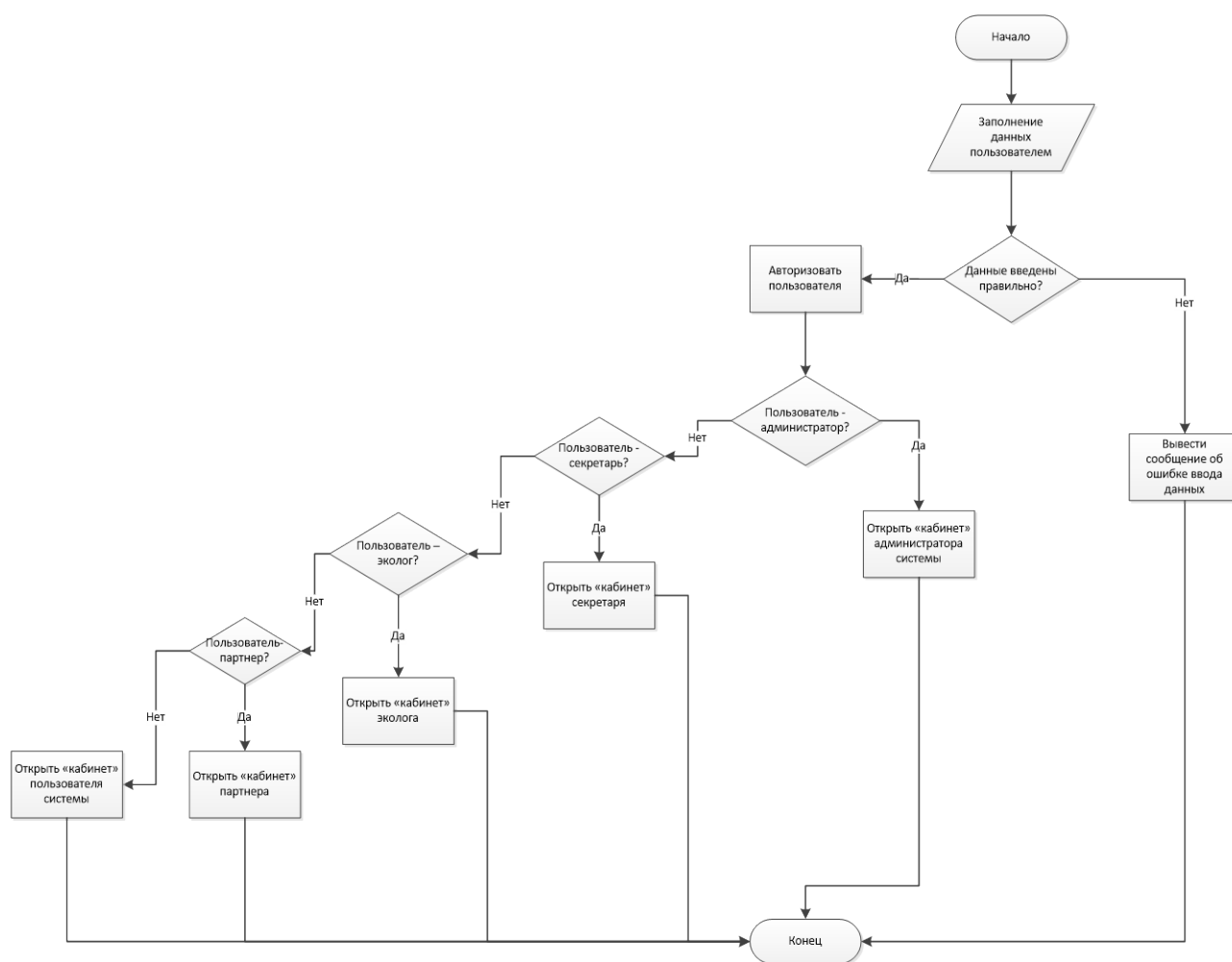


Рис. 3.2.1. Блок-схема алгоритма авторизации пользователя

3.3 Поиск информации об экологе

1. Общая характеристика: Алгоритм поиска осуществляет поиск информации о конкретном экологе или группе специалистов.

2. Используемые данные: Таблицы [dbo].[Ecologists].

3. Результаты выполнения: Отображение информации о конкретном экологе или группе специалистов в виде таблицы.

4. Математическое описание: -.

5. Логическое описание: Блок-схема алгоритма поиска информации об экологах приведена на рисунке 3.3.1.

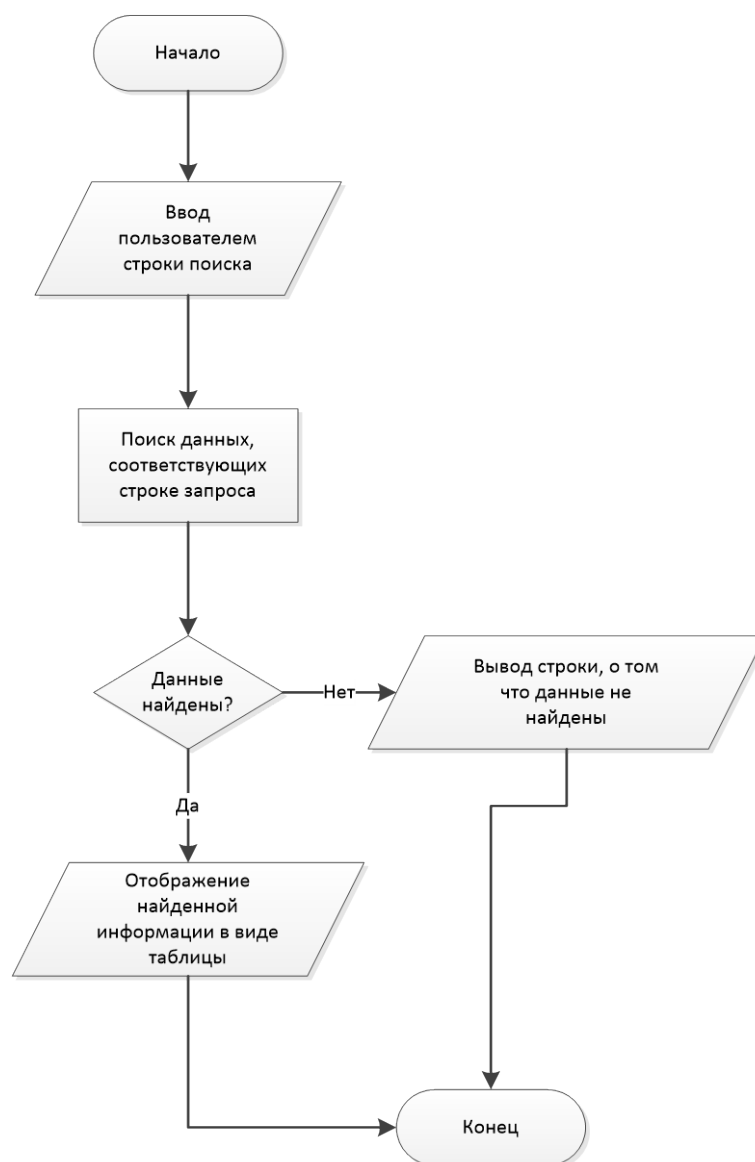


Рис. 3.3.1. Блок-схема алгоритма поиска информации об экологах

3.4 Фильтрация информационных данных об организациях/лицах-штрафниках

1. **Общая характеристика:** Алгоритм фильтрации данных по атрибуту «IsPayed».
2. **Используемые данные:** таблицы [dbo].[OrganisationDebtors].
3. **Результаты выполнения:** Отображение информации о выплативших/не выплативших штрафы организаций.
4. **Математическое описание:** -.

5. **Логическое описание:** Блок-схема алгоритма фильтрации организаций/лиц-штрафников приведена на рисунке 3.4.1

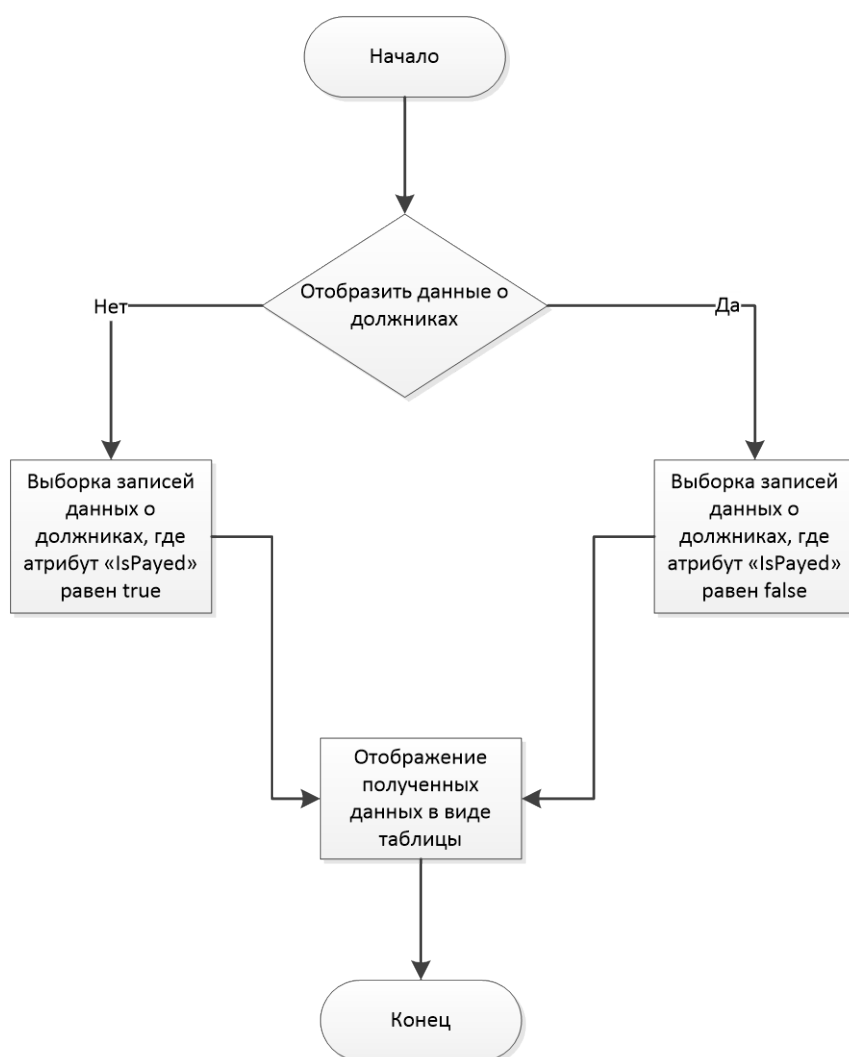


Рис. 3.4.1. Блок-схема алгоритма фильтрации организаций/лиц-штрафников

4 Прикладное программное обеспечение системы

4.1 Общая характеристика прикладного программного обеспечения

4.1.1. Оценка соответствия разработанного web-приложения общим требованиям

1. Инструментальная среда разработки – Visual Studio 2012 Ultimate.
2. СУБД – SQL Server 2012.
3. Просмотр всех данных в табличном виде – на страницах web-приложения все данные представлены в табличном виде. Например, отображаемые данные о достижениях экологического фонда заключают в себе заголовок, описание и файл-фотографию.
4. Ввод новых записей/редактирование уже существующих записей - осуществляется при помощи специальных форм-бланков, отображаемых отдельными страницами в приложении.
5. Удаление записей – все таблицы поддерживают интерфейс удаления записей. В таблице [dbo].[Complaints] удаления не происходит, а вместо этого происходит сокрытие записей (реализовано в учебных целях).
6. Все операции реализуются корректно с сохранением логической и ссылочной целостности.
7. Фильтрация и поиск – реализованы без применения языка SQL, но при помощи механизма «точечной нотации» языка программирования высокого уровня C#. При этом при фильтрации пользователей системы имеется возможность установки нескольких критериев фильтрации.
8. Моделирование данных выполнено при помощи программного пакета Erwin Data Modeler. Основания, на которых производился выбор, подробно представлены в пункте 2.1.4 пояснительной записки.

4.1.2. Оценка соответствия разработанного web-приложения дополнительным требованиям

В данном подразделе проводится оценка соответствия разработанного web-приложения уровню «5» по следующим критериям:

1. технология разработки web-системы – ASP .NET MVC;
2. количество таблиц в базе данных 15 (10) с суммарным числом атрибутов 76 (50);
3. используемая технология доступа к данным – ADO.NET Entity Framework 5;
4. система безопасности – с учетом того, что в Entity Framework 5 при подходе Code First невозможно создать пользователей базы данных, было принято решение создать этих пользователей внутри web-системы, при этом каждая вид пользователя обладает своими возможностями доступа к данным;
5. технология ADO.NET Entity Framework поддерживает возможность создания индексов и значений по умолчанию;
6. в ходе разработки web-приложения для доступа к данным применялись такие возможности БД, как триггеры, транзакции, хранимые процедуры и функции;
7. имена файлов, классов, объектов являются значимыми и позволяют легко ориентироваться в структуре проекта и коде приложения;
8. пользовательский интерфейс программы разрабатывался средствами HTML 5, CSS 3, jQuery и библиотеки стилей Bootstrap. Наличие такого числа технологий говорит о его достаточной сложности. Интерфейс выполняет и главную функцию системы – минимизация усилий пользователя в процессе его работы с системой;
9. web-приложение поддерживает расширенную обработку исключительных ситуаций. Например, при переходе пользователя на несуществующую страницу отобразится страница-ошибка с кодом 404;

10. в разработанном приложении используется шаблон главной страницы.
Это означает, что при переходе между страницами приложения основная часть (меню, форма авторизации, галерея, футер) не перерисовывается, а изменяется только информационная часть;
11. в ходе проектирования системы использовались многие элементы управления формы, применялись оригинальные решения при верстке макета сайта, вместо стандартных стилей элементов управления применялись стили из библиотеки Bootstrap;
12. выполненная работа является полностью оригинальной и влияние работ других студентов отсутствует.

4.2 Структура и состав прикладного программного обеспечения

4.2.1 Основные директории проекта приложения в VS 2012 Ultimate

В таблице 4.2.1.1 представлены основные директории структуры проекта с необходимыми описаниями.

Таблица 4.2.1.1

Основные директории проекта web-приложения

Название директории проекта	Описание
Properties	Директория, содержащая файл расширения .cs, в котором задаются основные параметры сборки приложения (copyright, trademark, culture и др.)
References	Директория содержит ссылки на сборки и библиотеки, включенные в проект.
App_Data	Директория, предназначенная для хранения данных приложения (в нашем случае подключенной базы данных).
App_Start	Директория, в которой хранятся фай-

	<p>лы .cs, задающие параметры подключения js-скриптов, выбора главной страницы приложения и т.д.</p>
Content	<p>Директория, в которой хранятся стили, отвечающие за внешний вид приложения. Папка не претерпевала никаких изменений в ходе разработки проекта, так как автор создал свои директории, где хранил изображения, стили и js-скрипты.</p>
Controllers	<p>Директория, содержащая MVC контроллеры.</p>
Images	<p>Директория, в которой хранятся изображения и картинки, которые были использованы при разработке сайта.</p>
Migrations	<p>Директория, содержащая файлы-миграции, которые автор автоматически генерировал в консоли Package Manager Console для создания базы данных. Такой подход к созданию БД объясняется тем, что была выбрана методика Code First при работе с Entity Framework 5. Файл configuration.cs содержит переопределенный метод void Seed(FundContext), который каждый раз выполняется при обновлении БД. В этом методе можно определить значения по умолчанию для БД, а также триггеры, функции и хранимые проце-</p>

	дуры.
Models	Директория, содержащая классы, на основе которых происходило построение базы данных.
Scripts	Директория, содержащая js-скрипты, которые были использованы при разработке web-приложения. Например, скрипт галереи, скрипты библиотеки стилей Bootstrap и jQuery.
Styles	Директория, в которой находятся стили, определяющие внешний вид пользовательского интерфейса.
Views	Директория, в которой хранятся представления страниц приложения, в том числе и шаблон мастер-страницы.

С точки зрения технологии паттерна MVC (Model-View-Controller) наибольший интерес представляет содержимое директорий Controllers, Views, Models, речь о которых пойдет в следующих трех частях пояснительной записки.

4.2.2 Контроллеры web-приложения

Контроллер (Controller) с точки зрения паттерна программирования MVC – это набор функций, которые выполняются по запросу со стороны пользователя. В таблице 4.2.1.2 представлены контроллеры web-приложения и соответствующие описания к ним.

Таблица 4.2.1.2

Контроллеры разработанного web-приложения

Контроллер	Назначение
------------	------------

AccountController	В контроллере реализован функционал авторизации пользователя системы. Количество значимых строк кода: 61.
AchievementsController	Отображение и поиск данных страницы достижений экологического фонда. Количество значимых строк кода: 70.
EcologistsController	Отображение и поиск данных на странице экологов приложения. Количество значимых строк кода: 55.
ErrorController	Отображение страниц-ошибок в случае возникновения исключительной ситуации. Количество значимых строк кода: 9.
HomeController	Отображение главной страницы сайта экологического фонда. Количество значимых строк кода: 3.
PartnersController	Отображение и поиск данных на странице организаций-партнеров. Количество значимых строк кода: 44.
ProblemsController	Отображение и поиск данных на странице экологических проблем web-приложения. Количество значимых строк кода: 23.
RegistrationController	В контроллере реализован функционал регистрации нового пользователя системы. Количество значимых строк кода: 54.
RoomAdministratorController	В контроллере реализован функционал администраторской секции приложе-

	<p>ния. Наиболее интересная часть - отображение пользователей системы по ролям и их фильтрация. Количество значимых строк кода: 163.</p>
RoomEcologistController	<p>В контроллере реализован функционал секции пользователя-эколога. Наиболее интересная часть – реализация механизма создания экологической проблемы. Количество значимых строк кода: 61.</p>
RoomPartnerController	<p>В контроллере реализован функционал секции представителя организации-партнера. Количество значимых строк кода: 11.</p>
RoomSecretaryController	<p>В контроллере реализован функционал секции секретаря системы, включая фильтрацию организаций/должников два из трех запросов системы. Количество значимых строк кода: 154.</p>
RoomUserController	<p>В контроллере реализован функционал секции авторизованного пользователя системы. Количество значимых строк кода: 32.</p>
SectionsController	<p>Отображение страницы экологических секций, поиск данных, а также запрос по экологическим секциям. Количество значимых строк кода: 70.</p>

4.2.3 Модели web-приложения

Модель (Model) – это класс, заключающий в себя поля-свойства и отношения, которые используются для генерации базы данных.

В проекте присутствует 13 моделей, которые подробно рассмотрены в таблице 4.2.3.1.

Таблица 4.2.3.1

Модели web-приложения

Модель	Назначение
Achievement	Описание сущности «Достижение» базы данных. В классе модели определены связи между данной сущностью и сущностями «Администратор» и «Экологическая проблема».
Complaint	Описание сущности «Жалоба» базы данных. В классе определены связи между данной сущностью и сущностями «Организация/Лицо-штрафник», «Пользователь» и «Экологическая проблема»
Council	Описание сущности «Экологический совет» базы данных. Определены связи между данной сущностью и сущностями «Экологическая проблема» и «Эколог».
EcologicalProblem	Описание сущности «Экологическая проблема» базы данных. Определены связи между данной сущностью и сущностями «Жалоба», «Эколог» и «Достижение».

OrganisationDebtor	Описание сущности «Организация/Лицо-штрафник». Определены связи между данной сущностью и сущностями «Жалоба» и «Секретарь»
PartnerShipRequest ¹	Описание сущности «Запрос на сотрудничество».
Section	Описание сущности «Экологическая секция». Определены связи между данной сущностью и сущностями «Эколог» и «Пользователь системы».
User	Описание сущности «Пользователь». Следует заметить, что от класса User.cs наследуются остальные типы пользователей системы, тем самым образуя связь типа «Есть».
RankUser	Описание сущности «Пользователь системы». Связь типа «Есть» с сущностью «Пользователь»
Secretary	Описание сущности «Секретарь». Связь типа «Есть» с сущностью «Пользователь».
Ecologist	Описание сущности «Эколог». Связь типа «Есть» с сущностью «Пользователь».
Administrator	Описание сущности «Администратор». Связь типа «Есть» с сущностью «Пользователь».
Partner	Описание сущности «Организация-

¹ Об использовании этой сущности более подробно рассказывается в пункте 4.3

	партнер (представитель)». Связь типа «Есть» с сущностью «Пользователь».
--	---

4.2.4 Контекст данных web-приложения

Для связи созданной модели данных в приложении определен специальный класс **FundContext**, наследуемый от стандартного класса Entity Framework **DbContext**. Этот класс используется повсеместно в разработанном приложении для доступа к данным, хранящимся в базе данных. Объект класса **FundContext** обладает рядом специальных методов для извлечения, сохранения новых и измененных данных.

4.2.5 Представления web-приложения

Представление (View) – файлы, состоящие из HTML-разметки с вставками программного кода на языке C#, другими словами это то, что пользователь видит в окне своего браузера. В ASP.NET MVC 4 для того чтобы осуществить вставку кода в разметку представления необходимо использовать один из двух движков ASPX и Razor. На самом деле разница невелика и зачастую выбор зависит от предпочтения самого разработчика. Был выбран движок Razor, потому что он подробнее описан в литературе по MVC и на форумах разработчиков.

В разработанном web-приложении получилось достаточно большое число представлений, поэтому при описании они будут разбиты по принципу «контроллер и его представления».

1. Контроллер **Account** имеет единственное представление в виде мастер-страницы (_masterPage.cshtml). Это обуславливается тем, что форма авторизации находится на всех страницах web-приложения.
2. Контроллер **Achievement** определяет функциональность представления AchievementsPage.cshtml, где отображаются достижения экологического фонда.

3. Контроллер **Ecologists** определяет функциональность представления EcologistsPage.cshtml, где отображаются сведения об экологах-специалистах фонда.
4. Контроллер **Home** отображает содержание главной страницы сайта Index.cshtml.
5. Контроллер **Partners** отображает содержание и определяет функциональность представления PartnersPage.cshtml.
6. Контроллер **Problems** отображает содержание страницы экологических проблем (ProblemsPage.cshtml) и определяет ее функционал.
7. Контроллер **Sections** отображает содержание страницы экологических секций (SectionsPage.cshtml) и определяет ее функционал.
8. Страницы, функциональность которых поддерживает контроллер **Registration**, показаны в таблице 4.2.5.1.

Таблица 4.2.5.1

Представления, поддерживаемые контроллером Registration

Название представления	Тип	Назначение
SignUpChoice.cshtml	Промежуточная страница	На данной странице производится выбор регистрации для разных категорий пользователей
SignUpFormEcologist.cshtml	Create ²	Регистрационная форма для эколога
SignUpFormPartner.cshtml	Create	Регистрационная форма для организации-партнера
SignUpFormRankUser.cshtml	Create	Регистрационная

² При создании нового представления ему можно задать один из шаблонов Create, Details, Empty, Delete, Edit и List. Применение такой техники построения сайта значительно сокращает время разработки.

		форма для пользователя системы
RegistrationResult.cshtml	Страница-сообщение	Страница, на которую осуществляется переход в случае успешного прохождения регистрации

9. Страницы, функциональность которых поддерживает контроллер RoomAdministrator, показаны в таблице 4.2.5.2.

Таблица 4.2.5.2

Представления, поддерживаемые контроллером RoomAdministrator

Название представления	Тип	Назначение
AdministratorRoom.cshtml	Страница-меню	Страница, где администратору системы предоставляется выбор между его дальнейшими действиями. Другими словами – меню выбора.
CreateNews.cshtml	Create	Создание новости о достижении.
CreateSection.cshtml	Create	Создание экологической секции.
EditNews.cshtml	Edit	Редактирование новости о достижении.
EditSection.cshtml	Edit	Редактирование данных экологической секции.

SystemNewsCreation.cshtml	Details	Отображение списка экологических проблем, по которым можно создать новость.
SystemSections.cshtml	Details	Отображение списка созданных секций.
SystemUsers.cshtml	Details	Отображение списка пользователей системы.

10. Страницы, функциональность которых поддерживает контроллер **RoomEcologist**, показаны в таблице 4.2.5.3.

Таблица 4.2.5.3

Представления, поддерживаемые контроллером RoomEcologist

Название представления	Тип	Назначение
EcologistRoom.cshtml	Страница-меню	Страница, где экологу-специалисту предоставляется выбор между его дальнейшими действиями. Другими словами – меню выбора.
Councils.cshtml	Details	Список экологических советов, на которые эколог может зарегистрироваться или наоборот deregистрироваться.

CreateComplaint.cshtml	Create	Создание жалобы на экологическое нарушение.
CreateProblem.cshtml	Edit	Создание экологической проблемы на основе жалобы
Complaints.cshtml	Edit	Отображение списка жалоб.

11. Страницы, функциональность которых поддерживает контроллер **RoomPartner**, показаны в таблице 4.2.5.4.

Таблица 4.2.5.4

Представления, поддерживаемые контроллером RoomPartner

Название представления	Тип	Назначение
PartnerRoom.cshtml	Страница-меню	Страница, где представителю организации-партнера предоставляется выбор между его дальнейшими действиями. Другими словами – меню выбора.
CreateComplaint.cshtml	Create	Создание жалобы на экологическое нарушение.

12. Страницы, функциональность которых поддерживает контроллер **RoomSecretary**, показаны в таблице 4.2.5.5.

Таблица 4.2.5.5

Представления, поддерживаемые контроллером RoomSecretary

Название представления	Тип	Назначение
SecretaryRoom.cshtml	Страница-меню	Страница, где секретарю предоставляется выбор между его дальнейшими действиями. Другими словами – меню выбора.
Councils.cshtml	Details	Список созданных экологических советов, на которые открыта регистрация.
Requests.cshtml	Details	Список заявок на регистрацию от представителей организаций-партнеров.
Debtors.cshtml	Details	Отображение списка организаций-штрафников.
CreateCouncil.cshtml	Create	Создание нового экологического совета.
CreateDebtor.cshtml	Create	Создание записи об организации-штрафнике на основе полученной жалобы.
EditCouncil.cshtml	Edit	Редактирование записи экологического совета.
EditDebtor.cshtml	Edit	Редактирование записи

		си об организа- ции/лице-штрафнике.
--	--	--

13. Страницы, функциональность которых поддерживает контроллер **RoomUser**, показаны в таблице 4.2.5.6.

Таблица 4.2.5.6

Представления, поддерживаемые контроллером RoomUser

Название представления	Тип	Назначение
UserRoom.cshtml	Страница-меню	Страница, где пользователю систему предоставляется выбор между его дальнейшими действиями. Другими словами – меню выбора.
Complaint.cshtml	Create	Создание жалобы на экологическое нарушение.
Sections.cshtml	Details	Список экологических секций, на которые пользователь может зарегистрироваться.

14. Страницы, функциональность которых поддерживает контроллер **Error**, показаны в таблице 4.2.5.7.

Таблица 4.2.5.4

Представления, поддерживаемые контроллером RoomPartner

Название представления	Тип	Назначение
General.cshtml	Страница-ошибки	Страница, появляющаяся в случае внутренней ошибки web-приложения.
Http404.cshtml	Страница-ошибки	Страница, появляющаяся в случае несуществования страницы, запрашиваемой пользователем.
Http403.cshtml	Страница-ошибки	Страница, появляющаяся в случае запрета доступа к определенному ресурсу.

4.3 Особенности реализации и сопровождения

Разработанная система имеет ряд важных особенностей, которые следует учитывать при ее дальнейшем сопровождении:

1. Сущность PartnershipRequests (Запрос на сотрудничество) в системе не используется. На этапе написания технического задания предполагалось хранить заявки организаций-штрафников в соответствующей этой сущности таблице. В процессе разработки было найдено другое решение, основанное на использование поля-флага для объекта сущности «Организация-партнер». При регистрации нового партнера, значение этого поля по умолчанию устанавливалось в значение false. Секретарь системы из своей «комнаты» может принять заявку, установив это значение в

true и разрешив вход в систему. В будущем планируется улучшить функционал запроса на сотрудничество, поэтому удалять сущность пока не имеет смысла.

2. Особое внимание следует обратить на то, что при малейшем изменении моделей сущностей базы данных необходимо применять механизм миграций, так как используемый подход к построению базы данных – Code First, предполагающий полностью ручное создание сущностей данных.
3. При проектировании связей типа «Многие-ко-многим» необходимо быть предельно внимательным в плане разрешения каскадного удаления записей таблиц. Например, в ходе разработки системы было потрачено много времени на установление такого типа связи между сущностями «Экологический совет» и «Эколог». При удалении «Экологического совета», система выдавала ошибку о том, что удаление объектов сущности «Эколог» может повлечь за собой непоправимые последствия. Это достаточно легко объясняется тем, что сущность «Эколог» также связана с сущностями «Экологическая секция» и «Экологическая проблема», поэтому удаление совета могло повредить и объекты этих сущностей. Решение – запрет каскадного удаления, но в некоторых особых случаях может применяться так называемое сокрытие данных при помощи специальных полей.
4. Рассмотренная в пункте 4 техника сокрытия данных была применена для случая удаления жалобы на экологическое нарушение. Это было сделано в учебных целях.

5 Руководство пользователя

5.1 Общие сведения

Разработанное web-приложение предназначено для автоматизации многих процессов работы экологического фонда. С системой могут работать 6 типов пользователей:

1. администратор;
2. секретарь;
3. специалист-эколог;
4. представитель организации-партнера;
5. авторизованный пользователь;
6. неавторизованный пользователь.

Уровень подготовки пользователей предусматривает навыки работы с ПК и с программами типа интернет-браузер.

5.2 Порядок и особенности работы

В текущей части будут описаны режимы работы с системой от лица каждого типа пользователя и подробно рассмотрен ее интерфейс.

5.2.1 Описание основных частей интерфейса системы

Первое, что видит на своем экране пользователь разработанной системы – это главная страница приложения. Она представлена на рисунке 5.2.1.1.



Рис 5.2.1.1 Главная страница web-приложения

Наиболее важная часть интерфейса располагается вверху окна – это блок, состоящий из формы авторизации и ленточного меню (представлен на рисунке 5.2.1.2).

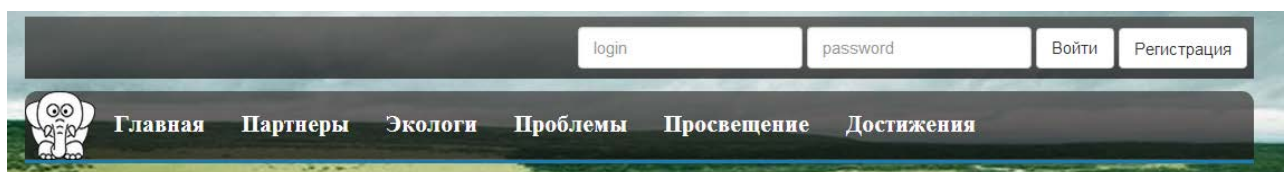


Рис 5.2.1.2 Форма авторизации и лента меню.

Если пользователь зарегистрирован, то он может осуществить вход на сайт и получить доступ к дополнительным функциям, разрешенным для его типа пользователя.

Меню сайта представляет собой группу ссылок, при нажатии на которые можно перемещаться между страницами web-приложения.

Центральная часть приложения (см. рис. 5.2.1.3) является динамической. Это означает, что при перемещении между страницами она будет меняться.

Об экологическом фонде "Слон"



Экологический фонд является внебюджетным фондом, средства которого направляются на решение экологических проблем окружающей среды. Учредителями экологических фондов являются краевые, областные и республиканские комитеты по экологии и природоохране. Организации, как правило, являются самостоятельными юридическими лицами, имеют собственный баланс. Основными задачами работы экологических фондов являются проведение мероприятий и разработка программ по следующим направлениям:

- создание информационной системы сбора, хранения, систематизации и обработки экологической информации;
- проведение мероприятий с целью решения экологических проблем;
- ведение базы сбора информации об экологических нарушениях. Нарушение может быть как со стороны организации (сброс химически опасных веществ заводом в реку), так и со стороны физического лица (браконьерство);
- организация экологического образования и воспитания, пропаганда экологических знаний.

Экологический фонд является неотъемлемой частью механизма регулирования природопользования и образуется за счет поступлений средств от предприятий и физических лиц нарушителей. Таким образом, в основном денежный баланс фонда формируется из следующих платежей:

- штрафы за загрязнение окружающей среды;
- сверхнормативное использование природных ресурсов;
- штрафы за нарушение природоохранного законодательства.

Рис 5.2.1.3 Центральная часть страницы

5.2.2 Неавторизованный пользователь системы

Неавторизованным пользователем информационной системы экологического фонда может быть как случайный посетитель, так и человек, который планирует зарегистрироваться и посещать экологические кружки, работать в экологическом фонде как специалист-эколог (при наличии соответствующего образования), или быть представителем организации, которая хочет наладить сотрудничество с фондом.

Первое, что, по мнению автора, должен сделать неавторизованный пользователь, - ознакомиться с представленной на сайте информацией. Перемещаясь между ссылками ленточного меню, пользователь может посетить следующие страницы:

1. «Главная страница» (см. рис. 5.2.1.1).
2. Партнеры – на странице (см. рис. 5.2.2.1) отображена основная информация об организациях-партнерах, сотрудничающих с экологическим фондом.

Организации-партнеры

<input type="text" value="Поиск..."/>					<input type="button" value="Искать"/>
Представитель ФИО	Компания	Адрес	Email	О компании	
Голобокова Анна Андреевна	AnnCompanyGroup	Ульяновск	golobokova_ann@gmail.ru	Мы компания, занимающаяся помощью природе.	

Рис 5.2.2.1 Информация об организациях-партнерах

3. «Экологи» – на странице (рис. 5.2.2.2) отражена основная информация о специалистах, работающих на экологический фонд. Также есть поисковая строка, при пользовании которой можно составить «запрос» и получить интересующую информацию (рис. 5.2.2.3).

Экологи нашего фонда

<div>Поиск...</div> <div>Искать</div>						
ФИО	Дата регистрации	Email	Область интересов	Образование	Регион базирования	Публикации
Харитонов Алексей Иванович	20.05.2014	demarvel@yandex.ru	Животные и растения. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	УлГТУ. Факультет информационных систем и технологий. Высшее	Калининградская область	Загрязнение реки Амур
Муравьев Игорь Игоревич	20.05.2014	muravey@yandex.ru	Океан, море и озера. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	УлГТУ. Факультет информационных систем и технологий. Высшее	Краснодарский край	Нет публикаций
Смеречинский Сергей Орестович	20.05.2014	smerechinskiy@yandex.ru	Интересуюсь многими сферами биологии и экологии. Lorem Ipsum - это текст-"рыба", часто	УлГТУ. Факультет информационных систем и технологий. Высшее	Смоленская область	Лесные пожары в Сибири

Рис 5.2.2.2 Информация об экологах-специалистах

Харитонов

Искать

ФИО	Дата регистрации	Email	Область интересов	Образование	Регион базирования	Публикации
Харитонов Алексей Иванович	20.05.2014	demarvel@yandex.ru	Животные и растения. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	УлГТУ. Факультет информационных систем и технологий. Высшее	Калининградская область	Загрязнение реки Амур

Рис 5.2.2.3 Пример использования поисковой строки

4. Чтобы ознакомиться с экологическими проблемами, которые в данный момент разрабатываются фондом необходимо перейти на страницу «Проблемы» (см. рис. 5.2.2.4).

Экологические проблемы

<div> <input type="text" value="Поиск..."/> <input type="button" value="Искать"/> </div>			
Название	Описание	Сумма	Ход решения
Загрязнение реки Амур	Проблема загрязнения реки Амур. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	2000000,00	Не решена
Загрязнение озера Байкал	Проблема загрязнения озера Байкал. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	2000000,00	Не решена
Лесные пожары на Алтае	Проблема лесных пожаров. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	2000000,00	Решена
Лесные пожары в Сибири	Лесной пожар в Сибири. Срочно требуется помощи фонда "Слон". Идем на помощь	100000000,00	Решена

Рис 5.2.2.4 Страница «Проблемы» web-приложения

5. На странице «Просвещение» (см. рис. 5.2.2.5) можно узнать, какие курсы преподаются, а также получить полную информацию о них.

Секции

<div> <input type="text" value="Эколог..."/> <input type="text" value="Другое..."/> <input type="button" value="Искать"/> </div>					
Название	Описание	Начало занятий	Число занятий	Свободные места	Преподаватель
Общий курс экологии. Часть 1.	Очень интересный курс. Подойдет для тех кто только начинается знакомиться с экологией. Ведет просто замечательный преподаватель	25.05.2014	15	39	Загайчук Иван Анатольевич
Общий курс экологии. Часть 2.	Очень интересный курс. Продолжение предыдущего курса. Подойдет для продвинутых пользователей. Ведет также просто замечательный преподаватель	30.05.2014	10	20	Харитонов Алексей Иванович
Экология и человек	Интересный курс как для детей, так и для взрослых	18.07.2014	10	20	Загайчук Иван Анатольевич
Экология и природа	Интереснейший курс	20.09.2014	10	15	Загайчук Иван Анатольевич

Рис 5.2.2.5 Страница «Просвещение» web-приложения

6. «Достижения» – это своеобразная новостная лента (см. рис. 5.2.2.6), рассказывающая о достижениях экологического фонда.

Достижения


Название	Описание	О проблеме	Фото	Опубликовано
Река Амур!	При непосредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительственной награды.	Загрязнение реки Амур	Нет фото	Моисеев Владислав Валерьевич
Пожар на Алтае потушен!	При непосредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительственной награды.	Лесные пожары на Алтае		Моисеев Владислав Валерьевич
Пожар в Сибири потушен!	При непосредственной поддержке фонда был потушен большой пожар в Сибирских лесах. Фонд удостоился правительственной награды.	Лесные пожары в Сибири	Нет фото	Желепов Алексей Сергеевич

Рис 5.2.2.6 Страница «Достижения» web-приложения

Неавторизованный пользователь также имеет возможность зарегистрироваться. Для этого ему необходимо осуществить переход на страницу регистрации, нажав на кнопку «Регистрация» рядом с формой авторизации (см. рис. 5.2.1.2).

После перехода открывается страница, представленная на рисунке (5.2.2.7)

Регистрация

Для пользователя

Хотите узнать побольше об экологии? Записаться на участие в экологических секциях? Вам сюда.

Для эколога

Если вы эколог-специалист и хотите сотрудничать с нашим фондом - Вам сюда.

Для организации партнера

Вы организация, заинтересованная в сотрудничестве? Представитель организации может зарегистрироваться здесь. (P.S. после регистрации придется немного подождать. Наш секретарь проверит вашу заявку)

Рис 5.2.2.7 Страница выбора регистрации web-приложения

Рассмотрим вариант регистрации «Для пользователя». Для этого нажмем на кнопку «Зарегистрироваться» – откроется форма (см. рис. 5.2.2.8) и заполним ее необходимыми данными.

Заполните предложенную регистрационную форму (для пользователя)

Введите ваше имя:	<input type="text" value="Иван"/>
Введите вашу фамилию:	<input type="text" value="Иванович"/>
Введите ваше отчество:	<input type="text" value="Иванов"/>
Укажите ваш пол:	<input type="text" value="Мужской"/>
Введите вашу дату рождения:	<input type="text" value="19.04.1994"/>
Придумайте логин:	<input type="text" value="ivan"/>
Придумайте пароль:	<input type="password" value="...."/>
Введите адрес электронной почты:	<input type="text" value="ivan@mail.ru"/>
Введите информацию себе:	<input type="text" value="Интересуюсь экологией...."/>

Рис 5.2.2.8 Страница регистрации пользователя

При нажатии на кнопку «Зарегистрироваться», в случае успешной регистрации будет произведен трансфер на страницу, представленную на рисунке 5.2.2.9, и создана новая учетная запись пользователя.

Наши поздравления - вы успешно зарегистрировались!

Рис 5.2.2.9 Сообщение о завершении регистрации

Для входа в систему необходимо ввести логин и пароль в форму авторизации и нажать на кнопку «Войти». При успешном входе отобразится приветственное сообщение (см. рис. 5.2.2.10).

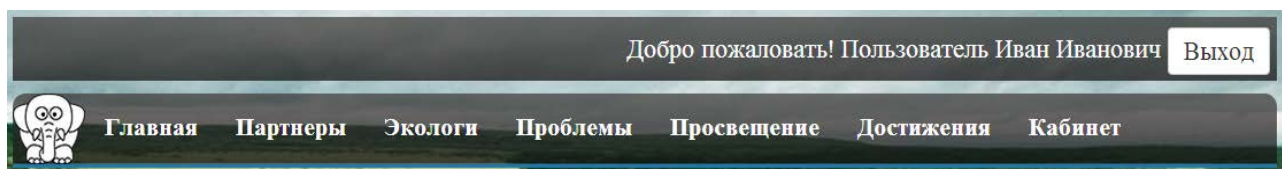


Рис 5.2.2.10 Сообщение об успешном входе в систему

Остальные виды регистрации работают аналогичным образом. Они также предусматривают заполнение регистрационной формы и ее отправку. Но в

случае регистрации организации-партнера вход в систему будет запрещен до тех пор, пока секретарь не рассмотрит и не примет заявку (подробно рассматривается в пункте 5.2.5).

5.2.3 Представитель организации-партнера

Представитель организации-партнера – это ответственное лицо, которое осуществляет взаимосвязь своей организации с фондом. В системе этому типу пользователей предлагается функция составления жалоб на экологические нарушения.

При успешной авторизации представитель организации-партнера может видеть следующую страницу (см. ри 5.2.3.1). Как видно на рисунке, появляется дополнительный пункт меню «Кабинет», который и открывает дополнительные возможности авторизованным пользователям системы.

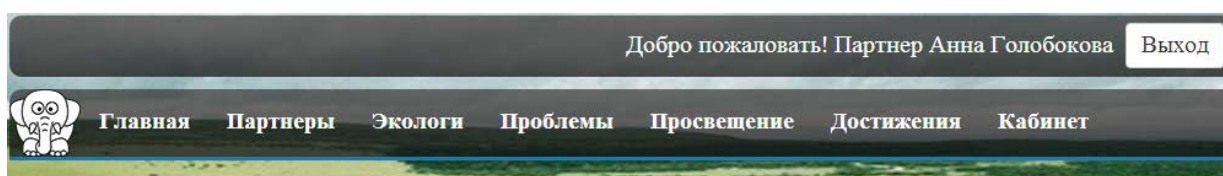


Рис. 5.2.3.1 Строка приветствия и пункт меню «Кабинет»

Кабинет представителя организации партнера представлен на рисунке 5.2.3.2. В нем доступен только один пункт меню – жалоба на экологическое нарушение.



Рис. 5.2.3.2 Меню «Кабинета» представителя-организации партнера

При переходе по ссылке открывается форма заполнения жалобы на экологическое нарушение (см. рис. 5.2.3.3)

Создание жалобы

Название жалобы

Описание жалобы

Дата публикации жалобы

Рис. 5.2.3.3 Страница заполнения формы жалобы

При успешном создании жалобы пользователя перенесет на страницу кабинета (см. 5.2.3.2). Пользователь может выйти из системы, нажав на кнопку «Выход» в верхней части web-сайта.

5.2.4 Администратор

Администратор – ключевая фигура системы, отвечающая за ее корректную работу, в некоторых случаях он сопровождает web-приложение. В разработанной системе у администратора большие полномочия – он может создавать/удалять пользователей системы, публиковать новости о достижениях фонда, открывать регистрацию на экологические секции.

На рисунке 5.2.4.1 представлена страница-меню «кабинета» администратора.

Администраторская

Пользователи

В данном блоке администратору системы предлагается возможность управления данными пользователей системы

Сейчас зарегистрированы:

Взрослые - 12

Дети - 1

[Перейти в раздел...](#)

Достижения и новости

В данном блоке администратору системы предлагается возможность добавления новостей о достижениях фонда

[Перейти в раздел...](#)

Экологические секции

В данном блоке администратору системы предлагается возможность добавления новостей о достижениях фонда

[Перейти в раздел...](#)

Рис. 5.2.4.1 Страница «кабинета» администратора

Меню состоит из трех подпунктов:

1. пользователи – при переходе по ссылке откроется возможность удаления/создания новых пользователей. Следует заметить, что рядом ссылкой перехода на страницу есть небольшая статистическая вставка, показывающая число зарегистрированных пользователей;
2. достижения и новости – при переходе по ссылке откроется функционал создания/редактирования/удаления новостей о достижениях экологического фонда;
3. экологические секции – данный пункт подменю отвечает за открытие регистрации на новые курсы для пользователей системы.

Страница «Пользователи» представлена на рисунке 5.2.4.2.

Пользователи системы

Фильтрация пользователей системы

Выберите пол: Выберите категорию:

ФИО	Дата рождения	Email	Роль	Логин	Пароль	Удаление
Евгений Прохоров Эдуардович	20.05.1995	evgeni@yandex.ru	Секретарь	evgwed	qwerty	Удаление
Алексей Харитонов Иванович	20.05.1997	demarvel@yandex.ru	Эколог	demarvel	hariton	Удаление
Игорь Муравьев Игоревич	20.05.1996	muravey@yandex.ru	Эколог	muravei	muravei	Удаление
Сергей Смеречинский Орестович	20.05.1994	smerechinskiy@yandex.ru	Эколог	sergey	sergey	Удаление
Дмитрий Ефимов Дмитриевич	20.05.1993	dimaefimov@yandex.ru	Эколог	dima	dima	Удаление
Иван Загайчук Анатолевич	20.05.1995	zagaichuk@yandex.ru	Эколог	zagaichuk	ziga	Удаление
Равиль Альмяшев Камилевич	20.05.1995	zagaichuk@yandex.ru	Пользователь	ravil	lalka	Удаление
Петр Сергеев Сергеевич	20.05.1995	zagaichuk@yandex.ru	Пользователь	serg	serg	Удаление
Анна Голобокова Андреевна	20.05.1994	golobokova_ann@gmail.ru	Партнер	ann	ann	Удаление
Иван Иванович Иванов	19.04.1994	ivan@mail.ru	Пользователь	ivan	ivan	Удаление
Алексей Сибирев Михайлович	10.10.1994	sibir@mail.ru	Партнер	sibir	12345	Удаление

Рис. 5.2.4.2 Страница отображения пользователей системы

Для удобства поиска определенных категорий пользователей администратору предоставляется алгоритм фильтрации, действия которого продемонстрировано на рисунке 5.2.4.3.

Пользователи системы

Фильтрация пользователей системы

Выберите пол: Выберите категорию:

ФИО	Дата рождения	Email	Роль	Логин	Пароль	Удаление
Алексей Харитонов Иванович	20.05.1997	demarvel@yandex.ru	Эколог	demarvel	hariton	Удаление
Игорь Муравьев Игоревич	20.05.1996	muravey@yandex.ru	Эколог	muravei	muravei	Удаление
Сергей Смеречинский Орестович	20.05.1994	smerechinskiy@yandex.ru	Эколог	sergey	sergey	Удаление
Дмитрий Ефимов Дмитриевич	20.05.1993	dimaefimov@yandex.ru	Эколог	dima	dima	Удаление
Иван Загайчук Анатолевич	20.05.1995	zagaichuk@yandex.ru	Эколог	zagaichuk	ziga	Удаление

Рис. 5.2.4.3 Отображение пользователей-экологов после применения механизма фильтрации

Также предоставляется возможность поиска конкретного пользователя. Для этого достаточно ввести какие-либо ключевые данные о нем, например Фамилию, Имя или Отчество. Результат работы поиска представлен на рисунке 5.2.4.4.

Пользователи системы

Фильтрация пользователей системы

Выберите пол: Не указан ▼ Выберите категорию: Не указан ▼ Фильтр

						Смеречинский	Искать
ФИО	Дата рождения	Email	Роль	Логин	Пароль	Удаление	
Сергей Смеречинский Орестович	20.05.1994	smerechinskiy@yandex.ru	Эколог	sergey	sergey	Удаление	

Добавить нового пользователя

Рис. 5.2.4.4 Результат работы поиска

Администратору предоставляется возможность удаления пользователей. Для этого необходимо нажать на ссылку «Удалить» в последнем столбце в строке, определяющей интересующего пользователя (см. рис. 5.2.4.4).

Также предоставляется возможность создания нового пользователя, при нажатии на кнопку «Добавить нового пользователя» будет произведен переход на страницу выбора регистрации (см. рис. 5.2.2.7). Регистрация подробно рассматривалась в пункте 5.2.1 настоящей пояснительной записки.

Вторая функция «кабинета» администратора – это публикация новостей. Эта страница «кабинета» представлена на рисунке 5.2.4.5.

Создание новостей по решенным проблемам

Фильтрация экологических проблем, согласно ходу их решения

Решенные Нерешенные Все

Проблема	Состояние	Запись о достижении	Удаление записи
Загрязнение реки Амур	В разработке	Редактировать	Удалить
Загрязнение озера Байкал	В разработке	Создать	-
Лесные пожары на Алтае	Решена	Редактировать	Удалить
Лесные пожары в Сибири	Решена	Редактировать	Удалить

Рис. 5.2.4.5 Страница создания/редактирования/удаления новостей

Для удобства работы администратора здесь также предусмотрен фильтр для отбора решенных/нерешенных фондом экологических проблем. В таблице представлены проблемы и показано (см. рис. 5.2.4.5) по каким из них можно создать новость, а по каким уже можно удалить или отредактировать созданную ранее.

Страница создания новости представлена на рисунке 5.2.4.6. Она представляет собой обыкновенную форму, после отправки которой, на странице достижений экологического фонда можно будет найти новую опубликованную заметку (см. рис. 5.2.4.7).

Создание новости

Заголовок новости

Описание новости

Фото

movement.png

Рис. 5.2.4.6 Страница создания новости

Достижения



Название	Описание	О проблеме	Фото	Опубликовано
Река Амур!	При непосредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительственной награды.	Загрязнение реки Амур	Нет фото	Моисеев Владислав Валерьевич
Очистка Байкала в самом разгаре!	На Байкале чистят воду.	Загрязнение озера Байкал		Желепов Алексей Сергеевич
Пожар на Алтае потушен!	При непосредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительственной награды.	Лесные пожары на Алтае		Моисеев Владислав Валерьевич
Пожар в Сибири потушен!	При непосредственной поддержке фонда был потушен большой пожар в Сибирских лесах. Фонд удостоился правительственной награды.	Лесные пожары в Сибири	Нет фото	Желепов Алексей Сергеевич

Рис. 5.2.4.7 Созданная новость на странице достижений

Также присутствует возможность редактирования новости. Для перехода в окно редактирования необходимо нажать на ссылку «Редактировать». При этом откроется форма, представленная на рисунке 5.2.4.8. После внесения необходимых изменений следует нажать на кнопку «Сохранить».

Редактировать новость

Заголовок новости

Река Амур!

Описание новости

При непосредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительственной награды.

Фото

Выберите файл

Файл не выбран

Сохранить

Вернуться

Рис. 5.2.4.8 Страница редактирования существующей новости

Функция удаления новости работает по схожему принципу с функцией удаления пользователя.

Администратор системы имеет возможность создания записей о новых курсах. Страница экологических секций администратора представлена на рисунке 5.2.4.9.

Экологические кружки

Название	Описание	Число участников	Число свободных мест	Время начала занятий	Преподаватель(ФИО)	Редактирование	Удаление
Общий курс экологии. Часть 1.	Очень интересный курс. Подойдет для тех кто только начинается знакомиться с экологией. Ведет просто замечательный преподаватель	1	39	25.05.2014	Загайчук Иван Анатольевич	Редактирование	Удаление
Общий курс экологии. Часть 2.	Очень интересный курс. Продолжение предыдущего курса. Подойдет для продвинутых пользователей. Ведет также просто замечательный преподаватель	0	20	30.05.2014	Харитонов Алексей Иванович	Редактирование	Удаление
Экология и человек	Интересный курс как для детей, так и для взрослых	0	20	18.07.2014	Загайчук Иван Анатольевич	Редактирование	Удаление
Экология и природа	Интереснейший курс	0	15	20.09.2014	Загайчук Иван Анатольевич	Редактирование	Удаление

Создать экологическую секцию

Рис. 5.2.4.9 Страница отображения всех экологических кружков

Страница создания экологической секции представлена на рисунке 5.2.4.10.

Создание секции

Название секции

Описание секции

Время старта занятий

Число мест

Число занятий

Преподаватель

Харитонов ▾

Харитонов
Муравьев
Смеречинский
Ефимов
Загайчук
Вернуться

Рис. 5.2.4.10 Страница создания экологической секции

Страница редактирования экологической секции представлена на рисунке 5.2.4.11.

Редактирование секции

Название секции

Общий курс экологии. Часть 1.

Описание секции

Очень интересный курс. Подойдет для тех кто только начинается знакомиться с экологией. Ведет просто замечательный преподаватель

Время старта занятий

25.05.2014 15:22:46

Число мест

40

Число занятий

15

Преподаватель

Загайчук ▼

Сохранить

Вернуться

Рис. 5.2.4.11 Страница редактирования экологической секции

Удаление экологической секции работает по схожему принципу с удалением пользователей системы.

5.2.5 Секретарь

Секретарь – важная должность в организации. Этот человек должен выполнять большой спектр повседневных задач, таких как рассмотрение заявок на сотрудничество от организаций, назначение и согласование дат проведения экологических советов, выявление организаций/лиц-штрафников и контроль хода уплаты штрафа ими.

«Кабинет» секретаря системы представлен на рисунке 5.2.5.1.

Секретарская

Рассмотрение заявок

В данном блоке секретарю предлагается рассмотреть поступающие заявки от организаций-партнеров на сотрудничество

Перейти в раздел...

Экологические советы Штрафники

В данном блоке секретарь может размещать информацию об экологических советах

Перейти в раздел...

В данном блоке секретарь может контролировать ход штрафных выплат со стороны организаций/лиц-штрафников

Перейти в раздел...

Рис. 5.2.5.1 Кабинет секретаря системы

В меню «Рассмотрение заявок» секретарю предлагается список заявок на сотрудничество от организаций. После рассмотрения секретарь может, как принять, так и отклонить заявку (см. рис. 5.2.5.2).

Заявки организаций партнеров

ФИО представителя	Email	Компания	О компании	Причина	Удаление	Принятие
Сибирев Алексей Михайлович	sibir@mail.ru	SibirCompany	Занимаемся заготовкой леса	Хотим помочь фонду	Удалить	Принять

Рис. 5.2.5.2 Страница рассмотрения заявок на сотрудничество

Меню «Экологические советы» представлено на рисунке 5.2.5.3.

Экологические советы

В настоящее не намечается ни одного совета.

Список проблем

Название	Описание	Дата публикации
Загрязнение реки Амур	Проблема загрязнения реки Амур. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Загрязнение озера Байкал	Проблема загрязнения озера Байкал. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Лесные пожары на Алтае	Проблема лесных пожаров. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Лесные пожары в Сибири	Лесной пожар в Сибири. Срочно требуется помощи фонда "Слон". Идем на помощь	20.05.2013

Рис. 5.2.5.3 Страница экологических советов кабинета секретаря

Как видно на рисунке 5.2.5.3 список созданных советов отображается в верхней части окна. В нижней части сосредоточен список проблем, которые должны быть рассмотрены. Также можно проводить запрос на выявление проблем в радиусе указанного числа дней. Для этого необходимо ввести в поле ввода необходимое число дней и нажать кнопку «Запрос». Данный поиск был создан с целью определения секретарем давно опубликованных проблем.

Страница создания совета представлена на рисунке 5.2.5.4.

Создание совета

Название совета

Дата проведения

Результат совета



Описание

Проблема

Рис. 5.2.5.4 Страница создания экологического совета

После создания совета по одной из проблем в верхней части экрана отобразятся сведения о созданных советах (см. рис. 5.2.3.5). При этом число специалистов, равное 0 означает, что на совет не зарегистрировалось ни одного эколога-специалиста.

Экологические советы

Название	Описание	Дата проведения	Проблема	Результат	Число специалистов	Удаление	Редактирование
Пожары в Сибири	Описание совета	30.09.2014	Лесные пожары в Сибири	Не решена	0	Удалить	Редактировать

Список проблем

Название	Описание	Дата публикации
Загрязнение реки Амур	Проблема загрязнения реки Амур. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Загрязнение озера Байкал	Проблема загрязнения озера Байкал. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Лесные пожары на Алтае	Проблема лесных пожаров. Lorem Ipsum - это текст-"рыба", часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.	20.05.2011
Лесные пожары в Сибири	Лесной пожар в Сибири. Срочно требуется помощи фонда "Слон". Идем на помощь	20.05.2013

Рис. 5.2.3.5 Созданный экологический совет

Страница редактирования совета представлена на рисунке 5.2.3.6.

Редактирование совета

Название совета

Пожары в Сибири

Дата проведения

30.09.2014 0:00:00

Результат совета

☐

Описание

Описание совета

Сохранить

Вернуться

Рис. 5.2.3.6 Страница редактирования экологического совета

Удаление совета производится аналогичным образом.

Меню «Штрафники» представлено на рисунке 5.2.3.7. В верхней части экрана представлен список жалоб, поданных пользователями системы и экологами. В нижней части список штрафников. У таблицы организаций/лиц-должников есть фильтр, позволяющий сортировать выплатившие штраф и не сделавшие это организации. Для выявления должников, оставшийся срок уплаты штрафа которых составляет менее трех дней, был создан специальный запрос.

Жалобы

Заголовок	Описание	Дата опубликования	Автор	Организация-Должник
Загрязнение Амура	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014	Харитонов Алексей Иванович	Создать
Загрязнение озера Байкал	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014	Харитонов Алексей Иванович	Создать
Лесные пожары на Алтае	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014	Харитонов Алексей Иванович	Создать
Выбросы в атмосферу на Кавказе	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014	Харитонов Алексей Иванович	Создать
Лесные пожары в Сибири	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014	Харитонов Алексей Иванович	Петр Петрович
Жалоба1	description1111	20.05.2014	Харитонов Алексей Иванович	Создать
Мусорка рядом с деревней Рублевка	В Рублевке невозможно дышать	21.05.2014	Голобокова Анна Андреевна	Создать

Организации/лица-штрафники

Фильтрация должников по состоянию уплаты штрафных санкций

Заплатившие

Незаплатившие

Все

Запрос на должников (срок выплаты, менее 3 дней)

Запрос

ФИО/Наименование организации	Причина	Срок	Сумма штрафа	Ход уплаты	Редактирование	Удаление
Петр Петрович	Поджог леса в Сибири	28.08.2014 15:22:46	10000000,00	Штраф не выплачен	Редактировать	Удаление

Рис. 5.2.3.7 Меню «Штрафники» секции секретаря.

Страница создания должника представлена на рисунке 5.2.3.8.

Создать должника

ФИО/Название организации

Причина

Сумма платежа

Сроки платежа

Ход уплаты штрафа



Email

Рис. 5.2.3.8 Страница создания нового должника

Страница редактирования информации о должнике представлена на рисунке 5.2.3.9.

Редактирование должника

ФИО/Название организации

Петр Петрович

Причина

Поджог леса в Сибири

Сумма платежа

10000000,00

Сроки платежа

28.08.2014 15:22:46

Ход уплаты штрафа



Email

petr@email.ru

Сохранить

Вернуться

Рис. 5.2.3.9 Страница редактирования информации о должнике

Удаление информации о должниках проводится ранее описанным способом.

5.2.6 Эколог-специалист

Экологи-специалисты проводят мониторинг на наличие экологических проблем, публикуют информацию о них, рассматривают жалобы, поступающие от пользователей системы, регистрируют свое участие в экологических советах, проводимых фондом.

Страница «кабинета» эколога представлена на рисунке 5.2.6.1.

Эколог

Экологические советы

В данном блоке экологу-специлисту предлагается возможность регистрации своего участия в экологических советах.

[Перейти в раздел...](#)

Жалобы на экологические нарушения

В данном блоке администратору системы предлагается возможность рассмотрения жалоб пользователей на экологические нарушения, создания экологических проблем на основе проведенного мониторинга.

[Перейти в раздел...](#)

Рис. 5.2.6.1 Страница «кабинета» эколога

В разделе «Экологические советы» экологу предлагается зарегистрировать/ deregистировать свое участие в экологических советах. Страница представлена на рисунке 5.2.6.2.

Экологические советы

Название	Описание	Экологическая проблема	Дата проведения	Регистрация
Пожары в Сибири	Описание совета	Лесные пожары в Сибири	30.09.2014	Зарегистрироваться

Рис. 5.2.6.1 Страница регистрации/дерегистрации специалиста на экологические советы

В разделе «Жалобы на экологические нарушения» (см. рис. 5.2.6.2) эколог просматривает жалобы на экологические нарушения, поступающие от пользователей системы. Если эколог считает жалобу недействительной, он может ее отклонить, в противном случае при принятии жалобы – эколог создает экологическую проблему (см. рис. 5.2.6.3).

Эколог проводит свой собственный мониторинг, на основе которого он также может создавать экологические проблемы. Для этого ему необходимо сначала (как и пользователю системы) создать жалобу на экологическое нарушение (см. рис. 5.2.6.4), а затем повторить действия, описанные в абзаце выше.

Жалобы пользователей

Название	Описание	Дата публикации	Создатель	Отклонение	Принятие
Выбросы в атмосферу на Кавказе	Часто используемый в печати и веб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века.	20.05.2014 15:22:45	Харитонов Алексей Иванович	Отклонить	Принять
Жалоба1	description1111	20.05.2014 15:22:46	Харитонов Алексей Иванович	Отклонить	Принять
Мусорка рядом с деревней Рублевка	В Рублевке невозможно дышать	21.05.2014 0:00:00	Голобокова Анна Андреевна	Отклонить	Принять

Создать жалобу

Рис. 5.2.6.2 Жалобы пользователей на экологические нарушения

Создание проблемы

Название проблемы

Описание проблемы

Требуемая сумма

Ход решения

☐

Дата публикации проблемы

Создать

Вернуться

Рис. 5.2.6.3 Создание экологической проблемы после принятия жалобы

Создание жалобы

Название жалобы

Описание жалобы

Дата публикации жалобы

Рис. 5.2.6.4 Создание жалобы экологом на основе мониторинга

5.2.7 Авторизованный пользователь системы

Пользователи системы – это заинтересованные люди в сохранении окружающей среды. Они могут публиковать свои наблюдения в виде жалоб на экологические нарушения, записывать на экологические тренинги и курсы, проводимые специалистами-экологами фонда.

Страница «кабинета» авторизованного пользователя представлена на рисунке 5.2.7.1.

Кабинет пользователя

Создание жалобы

В данном блоке пользователю предлагается оформить жалобу на экологическое нарушение.

Запись на курсы

В данном блоке пользователь может зарегистрироваться на курсы по экологическому воспитанию.

Рис. 5.2.7.1 «Кабинет» пользователя системы

В разделе «Создание жалобы» пользователь может сформировать и отправить жалобу на экологическое нарушение (см. рис. 5.2.7.2).

Создать жалобу

Название жалобы

Описание жалобы

Дата публикации жалобы

Рис. 5.2.7.2 Страница создания жалобы на экологическое нарушение

В разделе «Запись на курсы» пользователь имеет возможность регистрации своего участия в экологических тренингах и курсах, проводимых специалистами фонда. Страница с доступными курсами и регистрацией/дерегистацией на них, представлена на рисунке 5.2.7.3.

Экологические секции

Название	Описание	Преподаватель	Начало занятий	Число занятий	Число св. мест	Регистрация
Общий курс экологии. Часть 1.	Очень интересный курс. Подойдет для тех кто только начинается знакомиться с экологией. Ведет просто замечательный преподаватель	Загайчук Иван Анатольевич	25.05.2014	15	38	Дерегистрироваться
Общий курс экологии. Часть 2.	Очень интересный курс. Продолжение предыдущего курса. Подойдет для продвинутых пользователей. Ведет также просто замечательный преподаватель	Харитонов Алексей Иванович	30.05.2014	10	19	Дерегистрироваться
Экология и человек	Интересный курс как для детей, так и для взрослых	Загайчук Иван Анатольевич	18.07.2014	10	19	Дерегистрироваться
Экология и природа	Интереснейший курс	Загайчук Иван Анатольевич	20.09.2014	10	15	Зарегистрироваться

Рис. 5.2.7.3 Страница, отображающая проводимые фондом курсы и тренинги

5.3 Исключительные ситуации

Разработанное web-приложение оснащено глобальной обработкой исключительных ситуаций. Это означает, что при возникновении ошибки пользователь

будет перенаправлен на одну из так называемых страниц-заглушек, на которой будет отображена причина ошибки и вариант дальнейших действий пользователя.

Возможные исключительные ситуации и их обработка:

1. Ввод неверного логина или пароля при входе в систему.

Система отреагирует отображением соответствующего сообщения рядом с формой авторизации (см. рис. 5.3.1).

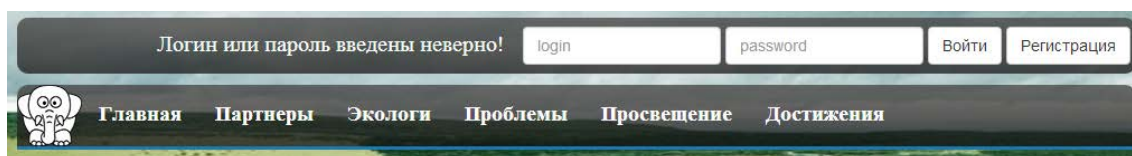


Рис. 5.3.1 Ошибка при неверном вводе логина или пароля

2. При регистрации представитель организации-партнера составляет заявку, которая после принимается или отклоняется секретарем системы. До момента принятия представителю будет запрещен вход в систему (см. рис. 5.3.2).

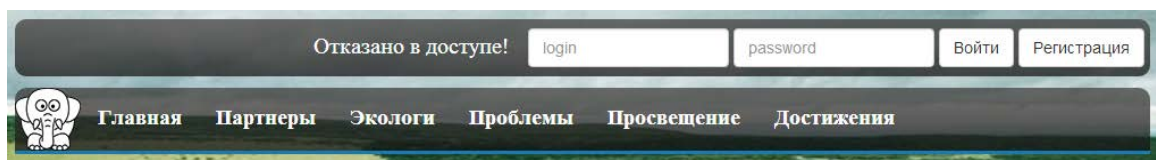


Рис. 5.3.2 Ошибка отказа доступа представителю организации-партнера

3. При создании/редактировании данных также имеют место ошибки. Например, при создании об экологической проблеме секретарь может забыть ввести строку даты, тогда система отреагирует, как показано на рисунке 5.3.3.

Создание проблемы

Название проблемы

Описание проблемы

Требуемая сумма

Ход решения

☐

Дата публикации проблемы

Требуется поле Дата публикации проблемы.

Рис. 5.3.2 Реакция системы на не введенные поля

4. Пользователь может запросить несуществующую страницу. Тогда отобразится ошибка с кодом 404 (рис. 5.3.3 и рис. 5.3.4).

 localhost:32131/RoomEcologist/nonexistent_page

Рис. 5.3.3 Запрос несуществующей страницы

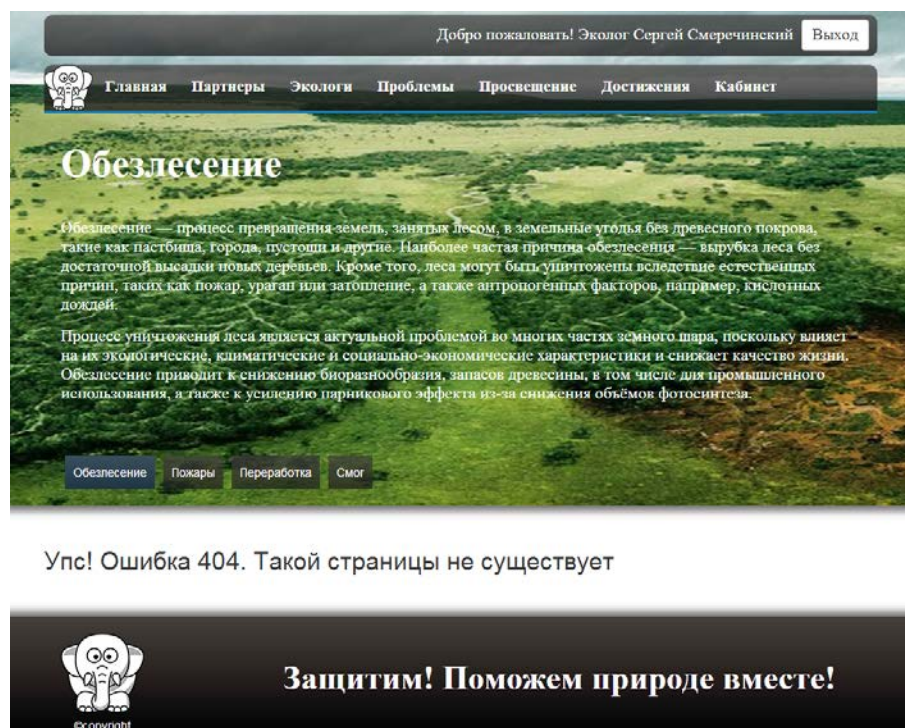


Рис. 5.3.3 Отображение ошибки запроса несуществующей страницы

5. Пользователь может ввести неверные данные в ходе заполнения заявки жалобы. Например, поля даты. В этом случае система отобразит ошибку, показанную на рисунке 5.3.4.

**Произошла ошибка. Перейдите на главную или другую страницу
страницу**

Информация об ошибке направлена администратору системы

Данные об ошибке:

An error occurred while saving entities that do not expose foreign key properties for their relationships. The EntityEntries property will return null because a single entity cannot be identified as the source of the exception. Handling of exceptions while saving can be made easier by exposing foreign key properties in your entity types. See the InnerException for details.

Рис. 5.3.4. Отображение страницы ошибка системы

Заключение

Реализованный программный продукт – web-приложение автоматизации работы процессов экологического фонда соответствует утвержденному техническому заданию, за исключением некоторых особенностей, которые были указаны в части 4.3 настоящей пояснительной записки.

Основными трудностями, с которыми столкнулся автор курсовой работы, являлись незнание технологий ASP.NET MVC и Entity Framework 5-ой версии.

Список использованных источников

1. Бизнес-словарь «Ведомости». Экологические фонды [Электронный ресурс] // Режим доступа:
<http://www.vedomosti.ru/glossary/%D0%AD%D0%BA%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5%20%D1%84%D0%BE%D0%BD%D0%B4%D1%8B> – Загл. с экрана.
2. Фонд имени Вернадского. Информация о фонде [Электронный ресурс] // Режим доступа: <http://www.rsci.ru/grants/fonds/141.php> - Загл. с экрана.
3. Jess Chadwick, Todd Snyder, and Hrusikesh Panda. Programming ASP.NET MVC 4 [Текст] – Published by O'Reilly Media, Inc., 1005 – 492 с.
4. Jeffrey Palermo, Jimmy Bogard, Eric Hexter, Matthew Hinze, Jeremy Skinner. ASP.NET MVC 4 in Action [Текст] – Published by Manning Publications Co. – 440 с.
5. Jon Galloway, Phil Haack, Brad Wilson, K. Scott Allen. Professional ASP.NET MVC 4 [Текст] – Published by Wiley Publishing Inc. – 507 с.

Приложение А. Исходные тексты программных модулей

Программный код контроллера AccountController.cs:

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class AccountController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult Login()
        {
            return View();
        }

        //ЛОГИН --- ПРИНЯТИЕ ДАННЫХ
        [HttpPost]
        public ActionResult Login(string login, string password)
        {
            //Проверяем пусты ли введенные пользователем данные
            if (string.IsNullOrEmpty(login) || string.IsNullOrEmpty(password))
            {
                ModelState.AddModelError(string.Empty, "Введите все данные!");
            }

            //если валидация прошла успешно, то логинимся
            if (ModelState.IsValid)
            {
                var user = db.Users.FirstOrDefault(n => n.Login == login && n.Password ==
password);

                if (user != null)
                {
                    Session.Add("SystemUserID", user.ID);

                    //Проверим кто пытается войти в систему
                    if (user is Administrator)
                    {
                        Session.Add("Role", "Administrator");
                        Session.Add("Greeting", "Добро пожаловать! Администратор " + us-
er.Name + " " + user.Surname);
                    }
                    if (user is RankUser)
                    {
                        Session.Add("Role", "User");
                        Session.Add("Greeting", "Добро пожаловать! Пользователь " + us-
er.Name + " " + user.Surname);
                    }
                    if (user is Ecologist)
                    {
                        Session.Add("Role", "Ecologist");
                        Session.Add("Greeting", "Добро пожаловать! Эколог " + user.Name + "
" + user.Surname);
                    }
                }
            }
        }
    }
}
```

```

    }
    if (user is Secretary)
    {
        Session.Add("Role", "Secretary");
        Session.Add("Greeting", "Добро пожаловать! Секретарь " + user.Name +
" " + user.Surname);
    }

    if (user is Partner)
    {
        if ((user as Partner).IsSolved)
        {
            Session.Add("Role", "Partner");
            Session.Add("Greeting", "Добро пожаловать! Партнер " + user.Name
+ " " + user.Surname);
        }
        else
        {
            Session.Add("Role", "NotAllowed");
        }
    }

    Session.Add("ErrorLogin", false);
}
else
{
    //ModelState.AddModelError("Error", "Данные неверны!");
    Session.Add("ErrorLogIn", true);
}
}

if (!string.IsNullOrEmpty(Request.UrlReferrer.AbsolutePath))
{
    return Redirect(Request.UrlReferrer.AbsolutePath);
}

return RedirectToAction("Index", "Home");
}

//ВЫХОД
public ActionResult Logout()
{
    Session.Clear();

    return RedirectToAction("Index", "Home");
}
}
}

```

Программный код контроллера AchievementController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class AchievementsController : Controller
    {
        FundContext db = new FundContext();
    }
}

```



```

public ActionResult AchievementsPage()
{
    return View(db.Achivements.ToList());
}

[HttpGet]
public ActionResult SearchAchievement(string searchString)
{
    if (string.IsNullOrEmpty(searchString))
    {
        return View("AchievementsPage", db.Achivements.ToList());
    }

    List<Achievement> achievements = new List<Achievement>();

    if (Session["Role"] != null && Session["Role"].ToString() == "Administrator")
    {
        achievements = db.Achivements.Where(n => (n.Title.Contains(searchString) ||
n.Description.Contains(searchString) ||
n.EcologicalProblem.Title.Contains(searchString) ||
n.Administrator.Name.Contains(searchString) ||
n.Administrator.Surname.Contains(searchString) ||
n.Administrator.FatherName.Contains(searchString))).ToList();
    }
    else
    {
        achievements = db.Achivements.Where(n => (n.Title.Contains(searchString) ||
n.Description.Contains(searchString) ||
n.EcologicalProblem.Title.Contains(searchString))).ToList();
    }

    return View("AchievementsPage", achievements);
}

//Забираем картинку
public FileContentResult GetPicture(int achievementID)
{
    var achievement = db.Achivements.Find(achievementID);

    if (achievement != null)
    {
        return File(achievement.PhotoFile, achievement.PhotoType);
    }

    return null;
}
}

```

Программный код контроллера EcologistsController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Diagnostics;
using FundApp.Models;

```

```

namespace FundApp.Controllers
{
    public class EcologistsController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult EcologistsPage()
        {
            List<Ecologist> ecologists = db.Ecologists.ToList();

            return View(ecologists);
        }

        //Поиск по экологам
        [HttpGet]
        public ActionResult SearchEcologists(string searchString)
        {
            if (string.IsNullOrEmpty(searchString)) {
                return View("EcologistsPage", db.Ecologists);
            }

            return View("EcologistsPage", db.Ecologists.Where(n =>
(n.Name.Contains(searchString) || n.Surname.Contains(searchString)
|| n.InterestsSphere.Contains(searchString) ||
n.FatherName.Contains(searchString)
|| n.DistrictLocation.Contains(searchString) ||
n.Education.Contains(searchString) || n.Email.Contains(searchString) ||
n.EcologicalProblems.Any(p => p.Title.Contains(searchString)))).ToList());
        }
    }
}

```

Программный код контроллера ErrorController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace FundApp.Controllers
{
    public class ErrorController : Controller
    {
        public ActionResult General(Exception exception)
        {
            return View(exception);
        }

        public ActionResult Http404()
        {
            return View();
        }

        public ActionResult Http403()
        {
            return View();
        }
    }
}

```

Программный код котроллера HomeController.cs:

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

Программный код котроллера PartnersController.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class PartnersController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult PartnersPage()
        {
            return View(db.Partners.ToList());
        }

        [HttpGet]
        public ActionResult SearchPartner(string searchString)
        {
            if (string.IsNullOrEmpty(searchString))
                return View("PartnersPage", db.Partners.ToList());

            List<Partner> partners = db.Partners.Where(n => (n.Name.Contains(searchString)
            || n.Surname.Contains(searchString) || n.FatherName.Contains(searchString) ||
            n.CompanyName.Contains(searchString) || n.Address.Contains(searchString) ||
            n.Email.Contains(searchString) || n.Description.Contains(searchString))).ToList();

            return View("PartnersPage", partners);
        }
    }
}
```

Программный код котроллера ProblemsController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class ProblemsController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult ProblemsPage()
        {
            List<EcologicalProblem> problems = db.EcologicalProblems.ToList();

            return View(problems);
        }

        [HttpGet]
        public ActionResult SearchProblem(string searchString)
        {
            if (string.IsNullOrEmpty(searchString))
            {
                return View("ProblemsPage", db.EcologicalProblems.ToList());
            }

            List<EcologicalProblem> requestedProblems = db.EcologicalProblems.Where(n =>
(n.Title.Contains(searchString) || n.Description.Contains(searchString))).ToList();

            return View("ProblemsPage", requestedProblems);
        }
    }
}

```

Программный код котроллера RegistrationController.cs:

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class RegistrationController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult SignUpChoice()
        {
            return View();
        }

        //Проверяем есть ли уже пользователь под таким же логином
        public bool IsNewLoginValid(string login)
        {
            if (db.Users.Count(n => n.Login == login) == 0)
                return true;
            else

```

```

        return false;
    }

    #region Для пользователя
    public ActionResult SignUpFormRankUser()
    {
        return View();
    }

    [HttpPost]
    public ActionResult SignUpFormRankUser(RegistrationViewModel registeredUser)
    {
        if (ModelState.IsValid)
        {
            if (!IsNewLoginValid(registeredUser.ItemRankUser.Login))
            {
                ViewBag.login = registeredUser.ItemRankUser.Login;
                return View(registeredUser);
            }
            else
            {
                registeredUser.ItemRankUser.RegistrationDate = DateTime.Now;

                db.RankUsers.Add(registeredUser.ItemRankUser);
                db.SaveChanges();

                return View("RegistrationResult");
            }
        }
        else
        {
            //при возникновении ошибки возвращаем данные пользователя
            return View(registeredUser);
        }
    }
    #endregion

    #region Для эколога
    public ActionResult SignUpFormEcologist()
    {
        return View();
    }

    [HttpPost]
    public ActionResult SignUpFormEcologist(RegistrationViewModel registeredUser)
    {
        if (ModelState.IsValid)
        {
            if (!IsNewLoginValid(registeredUser.ItemEcologist.Login))
            {
                ViewBag.login = registeredUser.ItemEcologist.Login;
                return View(registeredUser);
            }

            registeredUser.ItemEcologist.RegistrationDate = DateTime.Now;

            db.Ecologists.Add(registeredUser.ItemEcologist);
            db.SaveChanges();

            return View("RegistrationResult");
        }
        else
        {
            //при возникновении ошибки возвращаем данные пользователя

```

```

        return View(registeredUser);
    }
}
#endregion

#region Для партнера
public ActionResult SignUpFormPartner()
{
    return View();
}

[HttpPost]
public ActionResult SignUpFormPartner(Partner registeredPartner)
{
    if (!IsNewLoginValid(registeredPartner.Login))
    {
        ViewBag.login = registeredPartner.Login;
        return View(registeredPartner);
    }

    registeredPartner.RegistrationDate = DateTime.Now;
    registeredPartner.IsSolved = false; //секретарь еще не принял заявку

    registeredPartner.Secretary = db.Secretaries.FirstOrDefault();

    try
    {
        db.Partners.Add(registeredPartner);
        db.SaveChanges();

        ViewBag.regResult = true;

        return View("RegistrationResult");
    }
    catch
    {
        ViewBag.regResult = false;
        return View("RegistrationResult");
    }
}
#endregion
}
}

```

Программный код котроллера RoomAdministratorController.cs:

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Helpers;
using System.Web.Mvc;
using FundApp.Models;
using FundApp.Models.ViewModels;

namespace FundApp.Controllers
{
    public class RoomAdministratorController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult AdministratorRoom()

```

```

    {
        var adults = db.Database.SqlQuery<int>("SELECT [dbo].[GetAdultsQuantity]
()").FirstOrDefault();
        var children = db.Database.SqlQuery<int>("SELECT [dbo].[GetChildrenQuantity]
()").FirstOrDefault();

        ViewBag.adultsQuantity = adults;
        ViewBag.childrenQuantity = children;

        return View();
    }

#region Работа с пользователями

//Отображение страницы со списком пользователей
public ActionResult SystemUsers()
{
    //Вернем всех пользователей, кроме администратора, а то не Дай Бог еще себя лю-
бимых удалим
    List<User> users = db.Users.Where(n => !(n is Administrator)).ToList();

    return View(users);
}

//Удаление пользователя
public ActionResult DeleteUser(int id)
{
    var user = db.Users.Find(id);

    if (user != null)
    {
        try
        {
            db.Users.Remove(user);
            db.SaveChanges();
        }
        catch { }
    }

    return RedirectToAction("SystemUsers");
}

//Добавление нового пользователя
public ActionResult AddNewUser() {
    return RedirectToAction("SignUpChoice", "Registration", null);
}

//Поиск по фамилии/имени/отчеству
[HttpGet]
public ActionResult SearchUser(string searchString)
{
    if (!string.IsNullOrEmpty(searchString))
    {
        var users = (from n in db.Users
                     where (!(n is Administrator) && (n.Name.Contains(searchString)
|| n.Surname.Contains(searchString) || n.FatherName.Contains(searchString)))
                     select n).ToList();

        return View("SystemUsers", users);
    }
    else
    {
        var users = (from n in db.Users

```

```

        where (!(n is Administrator))
        select n).ToList();

        return View("SystemUsers", users);
    }
}

//Фильтрация
[HttpGet]
public ActionResult FilterUsers(string genderResult, string userType)
{
    if (!string.IsNullOrEmpty(genderResult))
    {
        bool gender = Convert.ToBoolean(genderResult);

        if (string.IsNullOrEmpty(userType))
        {
            return View("SystemUsers", db.Users.Where(n => n.Sex == gender && !(n is
Administrator))).ToList();
        }
        else
        {
            if (userType == "Partners")
            {
                return View("SystemUsers", db.Users.Where(n => (n.Sex == gender &&
(n is Partner))).ToList();
            }
            else if (userType == "Ecologists")
            {
                return View("SystemUsers", db.Users.Where(n => (n.Sex == gender &&
(n is Ecologist))).ToList();
            }
            else if (userType == "RankUsers")
            {
                return View("SystemUsers", db.Users.Where(n => (n.Sex == gender &&
(n is RankUser))).ToList();
            }
            else if (userType == "Secretaries")
            {
                return View("SystemUsers", db.Users.Where(n => (n.Sex == gender &&
(n is Secretary))).ToList();
            }
        }
    }
    else
    {
        if (string.IsNullOrEmpty(userType))
        {
            return View("SystemUsers", db.Users.Where(n => !(n is Administra-
tor))).ToList();
        }
        else
        {
            if (userType == "Partners")
            {
                return View("SystemUsers", db.Users.Where(n => n is Part-
ner).ToList());
            }
            else if (userType == "Ecologists")
            {
                return View("SystemUsers", db.Users.Where(n => n is Ecol-
ogist).ToList());
            }
            else if (userType == "RankUsers")

```



```

        {
            return View("SystemUsers", db.Users.Where(n => n is
RankUser).ToList());
        }
        else if (userType == "Secretaries")
        {
            return View("SystemUsers", db.Users.Where(n => (n is Secre-
tary)).ToList());
        }
    }
    }
    //не доберется никогда, но мало ли)
    return View("SystemUsers", db.Users.Where(n => !(n is Administrator)).ToList());
}

/*[HttpGet]
public ActionResult FilterPartners(string GenderResult)
{
    if (!string.IsNullOrEmpty(GenderResult))
        return View("SystemUsers", db.Users.Where(n => (n is Partner) && n.Sex ==
Convert.ToBoolean(GenderResult)).ToList());
    else
        return View("SystemUsers", db.Users.Where(n => (n is Partner)).ToList());
}*/

#endregion

#region Новости

//Отображение
public ActionResult CreateNews(int id)
{
    ViewBag.problemID = id;

    return View();
}

public ActionResult SystemNewsCreation()
{
    return View(db.EcologicalProblems.ToList());
}

//Создаем новость
[HttpPost]
public ActionResult CreateNews(AddAchievement achievement, int ecologicalProblemID)
{
    ViewBag.problemID = ecologicalProblemID;
    int currentAdminId;

    if (int.TryParse(Session["SystemUserId"].ToString(), out currentAdminId))
    {
        try
        {
            achievement.AchievementItem.Administrator =
db.Administrators.Find(currentAdminId);
            achievement.AchievementItem.EcologicalProblem =
db.EcologicalProblems.Find(ecologicalProblemID);

            if (achievement.ImageFile != null)
            {
                var image = new WebImage(achievement.ImageFile.InputStream);
                image.Resize(200, 133);
            }
        }
        catch { }
    }
}

```

```

        achievement.AchievementItem.PhotoType = achievement-
ment.ImageFile.ContentType;
        achievement.AchievementItem.PhotoFile = image.GetBytes();
    }

    db.Achivements.Add(achievement.AchievementItem);
    db.SaveChanges();

    return RedirectToAction("SystemNewsCreation");
}
catch { }
}

return View();
}

//Редактирование новости
[HttpGet]
public ActionResult EditNews(int id)
{
    AddAchievement item = new AddAchievement();
    item.AchievementItem = db.Achivements.Find(id);

    return View(item);
}

[HttpPost]
public ActionResult EditNews(AddAchievement a)
{
    var achievement = db.Achivements.Find(a.AchievementItem.AchievementID);

    if (a.ImageFile != null)
    {
        var image = new WebImage(a.ImageFile.InputStream);
        image.Resize(200, 133);

        achievement.PhotoType = a.ImageFile.ContentType;
        achievement.PhotoFile = image.GetBytes();
    }

    achievement.Title = a.AchievementItem.Title;
    achievement.Description = a.AchievementItem.Description;

    TryUpdateModel<Achievement>(achievement);
    db.Entry<Achievement>(achievement).State = System.Data.EntityState.Modified;
    db.SaveChanges();

    return RedirectToAction("SystemNewsCreation");
}

//Удаление новости
public ActionResult DeleteNews(int achievementID)
{
    var achievement = db.Achivements.Find(achievementID);

    if (achievement != null)
    {
        db.Achivements.Remove(achievement);
        db.SaveChanges();
    }

    return RedirectToAction("SystemNewsCreation");
}

```

```

//Фильтрация
[HttpGet]
public ActionResult SolvedProblemFilter()
{
    var filteredProblems = (from problem in db.EcologicalProblems
                            where problem.IsSolved == true
                            select problem).ToList();

    return View("SystemNewsCreation", filteredProblems);
}

[HttpGet]
public ActionResult UnsolvedProblemFilter()
{
    var filteredProblems = (from problem in db.EcologicalProblems
                            where problem.IsSolved == false
                            select problem).ToList();

    return View("SystemNewsCreation", filteredProblems);
}

#endregion

#region Экологические кружки

//Отображение страницы с секциями
[HttpGet]
public ActionResult SystemSections()
{
    return View(db.Sections.ToList());
}

private SelectListItem[] GetEcologistsSurnames(int id)
{
    List<Ecologist> availableEcologists = db.Ecologists.ToList();
    SelectListItem[] list = new SelectListItem[availableEcologists.Count];

    for (int i = 0; i < list.Length; i++)
    {
        if (availableEcologists[i].ID == id)
            list[i] = new SelectListItem() { Text = availableEcologists[i].Surname,
            Selected = true, Value = availableEcologists[i].ID.ToString() };
        else
            list[i] = new SelectListItem() { Text = availableEcologists[i].Surname,
            Selected = false, Value = availableEcologists[i].ID.ToString() };
    }

    return list;
}

//Создание экологической секции
[HttpGet]
public ActionResult CreateSection()
{
    ViewBag.ecologists = GetEcologistsSurnames(1); //при создании первый эколог бу-
дет выбран по умолчанию

    return View();
}

[HttpPost]
public ActionResult CreateSection(Section s, int ecologistID)
{

```

```

        ViewBag.ecologists = GetEcologistsSurnames(ecologistID);
        //Debug.WriteLine(ecologistID + " " + ecologistID.GetType());
        try
        {
            s.Ecologist = db.Ecologists.Find(ecologistID);

            //s.CalculateParticipantsCount();
            s.CalculateFreeSpots();

            db.Sections.Add(s);
            db.SaveChanges();

            return RedirectToAction("SystemSections");
        }
        catch
        {
            return View();
        }
    }

    //Удаление информации о секции
    public ActionResult DeleteSection(int sectionID)
    {
        var section = db.Sections.Find(sectionID);

        if (section != null)
        {
            db.Sections.Remove(section);
            db.SaveChanges();
        }

        return RedirectToAction("SystemSections");
    }

    //Редактирование секции
    [HttpGet]
    public ActionResult EditSection(int sectionID)
    {
        int ecologistID = db.Sections.Find(sectionID).Ecologist.ID;
        ViewBag.ecologists = GetEcologistsSurnames(ecologistID);

        return View(db.Sections.Find(sectionID));
    }

    [HttpPost]
    public ActionResult EditSection(Section s, int ecologistID)
    {
        ViewBag.ecologists = GetEcologistsSurnames(ecologistID); //учитываем dropdown
list

        var section = db.Sections.Find(s.SectionID);
        section.Ecologist = db.Ecologists.Find(ecologistID);

        TryUpdateModel<Section>(section);

        //расчет двух "столбцов"
        section.CalculateParticipantsCount();
        section.CalculateFreeSpots();

        db.Entry<Section>(section).State = System.Data.EntityState.Modified;
        db.SaveChanges();

        return RedirectToAction("SystemSections");
    }
}

```

```

        #endregion
    }
}

```

Программный код контроллера RoomEcologistController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;
using System.Diagnostics;

namespace FundApp.Controllers
{
    public class RoomEcologistController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult EcologistRoom()
        {
            return View();
        }

        #region Жалобы пользователей

        public ActionResult Complaints()
        {
            List<Complaint> complaints = db.Complaints.Where(n => n.IsHidden ==
false).ToList(); //вернем только те жалобы, которым еще не были рассмотрены
            return View(complaints);
        }

        //Создание проблемы на базе жалобы
        [HttpGet]
        public ActionResult CreateProblem(int complaintID)
        {
            ViewBag.complaintID = complaintID;
            return View();
        }

        [HttpPost]
        public ActionResult CreateProblem(EcologicalProblem problem, int complaintID)
        {
            Debug.WriteLine(complaintID);
            try
            {
                //одновременно "отклоняем" жалобу
                var complaint = db.Complaints.Find(complaintID);
                complaint.IsHidden = true;
                TryUpdateModel<Complaint>(complaint);
                db.Entry<Complaint>(complaint).State = System.Data.EntityState.Modified;
                db.SaveChanges();

                //сохраняем проблему
                problem.Creator = db.Ecologists.Find(Session["SystemUserID"]);
                problem.Complaint = db.Complaints.Find(complaintID);
                db.EcologicalProblems.Add(problem);
                db.SaveChanges();

                return RedirectToAction("Complaints");
            }
            catch { }
        }
    }
}

```

```

    }
    catch {
        return View();
    }
}

//Отклонение жалобы
public ActionResult DeclineComplaint(int complaintID)
{
    Debug.WriteLine(complaintID);
    var complaint = db.Complaints.Find(complaintID);

    if (complaint != null)
    {
        complaint.IsHidden = true;
        TryUpdateModel<Complaint>(complaint);
        db.Entry<Complaint>(complaint).State = System.Data.EntityState.Modified;
        db.SaveChanges();
    }

    return RedirectToAction("Complaints");
}

//Создание жалобы
[HttpGet]
public ActionResult CreateComplaint()
{
    return View();
}

[HttpPost]
public ActionResult CreateComplaint(Complaint complaint)
{
    if (complaint != null)
    {
        complaint.IsHidden = false; //открываем проблему
        complaint.Creator = db.Users.Find(Session["SystemUserID"]);
        db.Complaints.Add(complaint);
        db.SaveChanges();
    }

    return RedirectToAction("Complaints");
}

#endregion

#region Экологические советы

public ActionResult Councils()
{
    List<Council> councils = db.Councils.ToList();
    ViewBag.ecologist = db.Ecologists.Find(Session["SystemUserID"]);

    return View(councils);
}

//регистрация на совет
public ActionResult RegisterOnCouncil(int councilID)
{
    var council = db.Councils.Find(councilID);
    var ecologist = db.Ecologists.Find(Session["SystemUserID"]);

    council.Ecologists.Add(ecologist);
    TryUpdateModel<Council>(council);
}

```

```

        db.Entry<Council>(council).State = System.Data.EntityState.Modified;
        db.SaveChanges();

        return RedirectToAction("Councils");
    }

    //дерегистрация с совета
    public ActionResult DeregisterFromCouncil(int councilID)
    {
        var council = db.Councils.Find(councilID);
        var ecologist = db.Ecologists.Find(Session["SystemUserID"]);

        council.Ecologists.Remove(ecologist);
        TryUpdateModel<Council>(council);
        db.Entry<Council>(council).State = System.Data.EntityState.Modified;
        db.SaveChanges();

        return RedirectToAction("Councils");
    }

    #endregion
}
}

```

Программный код котроллера RoomPartnerController.cs:

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class RoomPartnerController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult PartnerRoom()
        {
            return View();
        }

        //Создание жалобы
        [HttpGet]
        public ActionResult CreateComplaint()
        {
            return View();
        }

        [HttpPost]
        public ActionResult CreateComplaint(Complaint complaint)
        {
            if (complaint != null)
            {
                complaint.Creator = db.Users.Find(Session["SystemUserID"]);
                db.Complaints.Add(complaint);
                db.SaveChanges();
            }
            return RedirectToAction("PartnerRoom");
        }
    }
}

```

```
}  
}
```

Программный код контроллера RoomSecretaryController.cs:

```
using System;  
using System.Collections.Generic;  
using System.Data.SqlClient;  
using System.Diagnostics;  
using System.Linq;  
using System.Web;  
using System.Web.Mvc;  
using FundApp.Models;  
using FundApp.Models.ViewModels;  
  
namespace FundApp.Controllers  
{  
    public class RoomSecretaryController : Controller  
    {  
        FundContext db = new FundContext();  
  
        public ActionResult SecretaryRoom()  
        {  
            return View();  
        }  
  
        #region Заявки  
  
        //Отображение списка заявок  
        public ActionResult Requests()  
        {  
            return View(db.Partners.Where(n => n.IsSolved == false).ToList());  
        }  
  
        //Удаление заявки  
        public ActionResult DeleteRequest(int requestID)  
        {  
            var request = db.Partners.Find(requestID);  
  
            if (request != null)  
            {  
                db.Partners.Remove(request);  
                db.SaveChanges();  
            }  
  
            return View("Requests", db.Partners.Where(n => n.IsSolved == false).ToList());  
        }  
  
        //Принятие заявки  
        public ActionResult AcceptRequest(int requestID)  
        {  
            var request = db.Partners.Find(requestID);  
            Debug.WriteLine(Session["SystemUserID"]);  
  
            if (request != null)  
            {  
                request.IsSolved = true;  
                request.Secretary = db.Secretaries.Find(Session["SystemUserID"]);  
                db.SaveChanges();  
            }  
  
            return View("Requests", db.Partners.Where(n => n.IsSolved == false).ToList());  
        }  
    }  
}
```



```

#endregion

#region Экологические советы

//Получим список проблем для View
private SelectListItem[] GetProblemsList(int id)
{
    List<EcologicalProblem> problems = db.EcologicalProblems.ToList();
    SelectListItem[] list = new SelectListItem[problems.Count];

    for (int i = 0; i < list.Length; i++)
    {
        if (problems[i].ProblemID == id)
            list[i] = new SelectListItem() { Text = problems[i].Title, Selected =
true, Value = problems[i].ProblemID.ToString() };
        else
            list[i] = new SelectListItem() { Text = problems[i].Title, Selected =
false, Value = problems[i].ProblemID.ToString() };
    }

    return list;
}

//Отображение экологических советов
public ActionResult Councils()
{
    //Скрывать ли ссылку создания совета или нет
    if (db.Councils.ToList().Count == db.EcologicalProblems.ToList().Count)
        ViewBag.showCreateLink = false;
    else
        ViewBag.showCreateLink = true;

    CouncilsProblems list = new CouncilsProblems();
    list.listCouncils = db.Councils.ToList();
    list.listProblems = (from problem in db.EcologicalProblems
                        select problem).ToList();

    return View(list);
}

//Запрос на проблемы
[HttpGet]
public ActionResult GetCrucialProblems(string daysRange)
{
    CouncilsProblems list = new CouncilsProblems();
    list.listCouncils = db.Councils.ToList();
    int days;

    if (int.TryParse(daysRange, out days))
    {
        try
        {
            list.listProblems =
db.Database.SqlQuery<EcologicalProblem>("GetEcologicalProblems @days_range", new
SqlParameter("days_range", days)).ToList();
            int l = list.listProblems.Count;
        }
        catch {
            list.listProblems = (from problem in db.EcologicalProblems
                                select problem).ToList();
        }
    }
    else

```

```

    {
        list.listProblems = (from problem in db.EcologicalProblems
                             select problem).ToList();
    }

    //Скрывать ли ссылку создания совета или нет
    if (db.Councils.ToList().Count == db.EcologicalProblems.ToList().Count)
        ViewBag.showCreateLink = false;
    else
        ViewBag.showCreateLink = true;

    return View("Councils", list);
}

//Удаление совета
public ActionResult DeleteCouncil(int councilID)
{
    var council = db.Councils.Find(councilID);

    if (council != null)
    {
        council.Ecologists.Clear();
        db.Councils.Remove(council);
        db.SaveChanges();
    }

    return RedirectToAction("Councils");
}

//Редактирование совета
[HttpGet]
public ActionResult EditCouncil(int councilID)
{
    int problemID = db.Councils.Find(councilID).Problem.ProblemID;
    //ViewBag.problems = GetProblemsList(problemID);
    ViewBag.problemID = problemID;

    return View(db.Councils.Find(councilID));
}

[HttpPost]
public ActionResult EditCouncil(Council c, int problemID)
{
    ViewBag.problems = GetProblemsList(problemID); //отсылка в View
    var council = db.Councils.Find(c.CouncilID); //редактируемый совет

    council.Problem = db.EcologicalProblems.Find(problemID);

    TryUpdateModel<Council>(council);
    db.Entry<Council>(council).State = System.Data.EntityState.Modified;

    db.SaveChanges();

    //Debug.WriteLine(c.Problem.ProblemID);
    var problem = db.EcologicalProblems.Find(problemID);
    bool result = db.Councils.Find(c.CouncilID).CouncilResult; ;
    string name = problem.Title;

    problem.IsSolved = result;

    TryUpdateModel<EcologicalProblem>(problem);
    db.Entry<EcologicalProblem>(problem).State = System.Data.EntityState.Modified;
}

```

```

        db.SaveChanges();

        problem.Title = name;
        db.SaveChanges();

        return RedirectToAction("Councils");
    }

    //СОЗДАНИЕ ЭКОЛОГИЧЕСКОГО СОВЕТА

    //При создании необходимо учесть, что проблемы, которые будут уже рассматриваются не
    включать
    private SelectListItem[] GetNonOverviewedProblems()
    {
        List<EcologicalProblem> problems = db.EcologicalProblems.ToList();
        List<Council> councils = db.Councils.ToList();

        for (int i = 0; i < councils.Count; i++)
        {
            int j = 0;
            while (j < problems.Count)
            {
                if (councils[i].Problem.ProblemID == problems[j].ProblemID)
                {
                    problems.Remove(problems[j]);
                }
                else
                {
                    j++;
                }
            }
        }

        if (problems.Count != 0)
        {
            SelectListItem[] list = new SelectListItem[problems.Count];

            for (int i = 0; i < list.Length; i++)
            {
                //if (i == 0)
                list[i] = new SelectListItem() { Text = problems[i].Title, Selected
= true, Value = problems[i].ProblemID.ToString() };
                //else
                // list[i] = new SelectListItem() { Text = problems[i].Title, Select-
ed = false, Value = problems[i].ProblemID.ToString() };
            }

            return list;
        }
        else
        {
            return null;
        }
    }

    [HttpGet]
    public ActionResult CreateCouncil()
    {
        SelectListItem[] list = GetNonOverviewedProblems();

        if (list != null)
        {
            ViewBag.problems = GetNonOverviewedProblems();
            return View();
        }
        else
        {
            return RedirectToAction("Councils");
        }
    }

```

```

    }
}

[HttpPost]
public ActionResult CreateCouncil(Council c, int problemID)
{
    var problem = db.EcologicalProblems.Find(problemID);
    try
    {
        c.Problem = problem;
        db.Councils.Add(c);
        db.SaveChanges();

        return RedirectToAction("Councils");
    }
    catch
    {
        return View();
    }
}
#endregion

#region Организации-штрафники

//Формирование страницы
[HttpGet]
public ActionResult Debtors()
{
    //возвращаем в View нужные данные для формирования двух таблиц
    DebtorComplaint list = new DebtorComplaint();
    list.listComplaints = db.Complaints.ToList();
    list.listDebtors = db.OrganisationDebtors.ToList();

    return View(list);
}

//Удаление должника
public ActionResult DeleteDebtor(int debtorID)
{
    var debtor = db.OrganisationDebtors.Find(debtorID);

    if (debtor != null)
    {
        db.OrganisationDebtors.Remove(debtor);
        db.SaveChanges();
    }

    return RedirectToAction("Debtors");
}

//Создание должника
[HttpGet]
public ActionResult CreateDebtor(int complaintID)
{
    ViewBag.complaintID = complaintID;
    return View();
}

[HttpPost]
public ActionResult CreateDebtor(OrganisationDebtor d, int complaintID)
{
    Debug.WriteLine(d.Name);
    try
    {

```

```

        d.Complaint = db.Complaints.Find(complaintID);
        db.OrganisationDeptors.Add(d);
        db.SaveChanges();

        return RedirectToAction("Debtors");
    }
    catch {
        ViewBag.complaintID = complaintID;
        return View();
    }
}

//Редактирование должника
[HttpGet]
public ActionResult EditDebtor(int debtorID)
{
    var debtor = db.OrganisationDeptors.Find(debtorID);
    return View(debtor);
}

[HttpPost]
public ActionResult EditDebtor(OrganisationDebtor d)
{
    var debtor = db.OrganisationDeptors.Find(d.OrganisationDebtorID);
    TryUpdateModel<OrganisationDebtor>(debtor);
    db.Entry<OrganisationDebtor>(debtor).State = System.Data.EntityState.Modified;
    db.SaveChanges();

    return RedirectToAction("Debtors");
}

//Фильтрация должников
[HttpGet]
public ActionResult PayedDebtorFilter()
{
    DebtorComplaint list = new DebtorComplaint();
    list.listComplaints = db.Complaints.ToList();
    list.listDebtors = (from debtor in db.OrganisationDeptors
                        where debtor.IsPayed == true
                        select debtor).ToList();

    return View("Debtors", list);
}

[HttpGet]
public ActionResult UnpayedDebtorFilter()
{
    DebtorComplaint list = new DebtorComplaint();
    list.listComplaints = db.Complaints.ToList();
    list.listDebtors = (from debtor in db.OrganisationDeptors
                        where debtor.IsPayed == false
                        select debtor).ToList();

    return View("Debtors", list);
}

//Запрос на должника
public ActionResult QueryForCrucialDebtors()
{
    DebtorComplaint list = new DebtorComplaint();

    list.listDebtors = db.Database.SqlQuery<OrganisationDebtor>("GetCrucialDebtor
@day_count", new SqlParameter("day_count", 3)).ToList();

```

```

        list.listComplaints = db.Complaints.ToList();

        return View("Debtors", list);
    }
    #endregion
}
}

```

Программный код котроллера RoomUserController.cs:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class RoomUserController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult UserRoom()
        {
            return View();
        }

        #region Жалоба

        [HttpGet]
        public ActionResult Complaint()
        {
            return View();
        }

        //создание и отправка жалобы
        [HttpPost]
        public ActionResult Complaint(Complaint complaint)
        {
            if (complaint != null)
            {
                complaint.Creator = db.Users.Find(Session["SystemUserID"]);
                db.Complaints.Add(complaint);
                db.SaveChanges();
            }
            return RedirectToAction("UserRoom");
        }

        #endregion

        #region Курсы

        public ActionResult Sections()
        {
            List<Section> sections = db.Sections.ToList();
            ViewBag.participant = db.RankUsers.Find(Session["SystemUserID"]);

            return View(sections);
        }

        //регистрация на курсы
        public ActionResult RegisterOnSection(int sectionID)

```

```

{
    var section = db.Sections.Find(sectionID);

    //Если свободные места еще есть
    if (section.FreeSpotsCount > 0)
    {
        var participant = db.RankUsers.Find(Session["SystemUserID"]);

        section.Participants.Add(participant);
        //section.CalculateParticipantsCount();
        //section.CalculateFreeSpots();
        TryUpdateModel<Section>(section);
        db.Entry<Section>(section).State = System.Data.EntityState.Modified;

        db.SaveChanges();
    }

    return RedirectToAction("Sections");
}

//дерегистрация
public ActionResult DeregisterFromSection(int sectionID)
{
    var section = db.Sections.Find(sectionID);
    var participant = db.RankUsers.Find(Session["SystemUserID"]);

    section.Participants.Remove(participant);
    //section.CalculateParticipantsCount();
    //section.CalculateFreeSpots();
    TryUpdateModel<Section>(section);
    db.Entry<Section>(section).State = System.Data.EntityState.Modified;

    db.SaveChanges();

    return RedirectToAction("Sections");
}

#endregion
}
}

```

Программный код контроллера SectionsController.cs:

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using FundApp.Models;

namespace FundApp.Controllers
{
    public class SectionsController : Controller
    {
        FundContext db = new FundContext();

        public ActionResult SectionsPage()
        {
            return View(db.Sections.ToList());
        }

        //Запрос(поиск) по сущностям "Эколог" и "Секция"
    }
}

```

```

[HttpGet]
public ActionResult SearchSection(string searchString, string ecologistName)
{
    if (string.IsNullOrEmpty(searchString) &&
string.IsNullOrEmpty(ecologistName))
    {
        return View("SectionsPage", db.Sections.ToList());
    }

    int lessonsCount;
    int freeSpots;
    DateTime date;

    Int32.TryParse(searchString, out lessonsCount);
    Int32.TryParse(searchString, out freeSpots);
    DateTime.TryParse(searchString, out date);

    List<Section> sections = db.Ecologists.Where(e =>
(e.Surname.Contains(ecologistName) || e.Name.Contains(ecologistName) ||
e.FatherName.Contains(ecologistName)))
        .SelectMany(e => db.Sections.Where(s =>
(s.Ecologist == e && (s.Title.Contains(searchString) || s.Description.Contains(searchString)
|| s.FreeSpotsCount == freeSpots || s.LessonsCount == lessonsCount
|| (s.StartLessonsTime.Year == date.Year && s.StartLessonsTime.Month == date.Month &&
s.StartLessonsTime.Day == date.Day))))))
        .ToList();

    /*
        List<Section> sections = db.Sections.Where(n => (n.Title.Contains(searchString)
|| n.Description.Contains(searchString)
||
n.Ecologist.Name.Contains(searchString) || n.Ecologist.Surname.Contains(searchString)
||
n.Ecologist.FatherName.Contains(searchString) || n.LessonsCount == lessonsCount
|| n.FreeSpotsCount == freeSpots ||
(n.StartLessonsTime.Year == d.Year && n.StartLessonsTime.Month == d.Month &&
n.StartLessonsTime.Day == d.Day))).ToList();*/

    return View("SectionsPage", sections);
}
}
}

```

Программный код Configuration.cs:

```

namespace FundApp.Migrations
{
    using System;
    using System.Data.Common;
    using System.Data.Entity;
    using System.Data.Entity.Migrations;
    using System.Linq;
    using FundApp.Models;

    internal sealed class Configuration :
DbMigrationsConfiguration<FundApp.Models.FundContext>
    {
        public Configuration()
        {
            AutomaticMigrationsEnabled = false;

```



```

}

protected override void Seed(FundApp.Models.FundContext db)
{
    // This method will be called after migrating to the latest version.

    // You can use the DbSet<T>.AddOrUpdate() helper extension method
    // to avoid creating duplicate seed data. E.g.
    //
    // context.People.AddOrUpdate(
    //     p => p.FullName,
    //     new Person { FullName = "Andrew Peters" },
    //     new Person { FullName = "Brice Lambson" },
    //     new Person { FullName = "Rowan Miller" }
    // );
    //
    db.Administrators.AddOrUpdate(
        n => n.Surname,
        new Administrator { Name = "Алексей", Surname = "Желепов", FatherName =
"Сергеевич", Sex = true, BirthDate = DateTime.Now.AddYears(-20), RegistrationDate =
DateTime.Now, Login = "alexey", Password = "0000", Email = "a.zhelepov@yandex.ru",
PhoneNumber = "44-67-77"},
        new Administrator { Name = "Владислав", Surname = "Моисеев", FatherName =
"Валерьевич", Sex = true, BirthDate = DateTime.Now.AddYears(-19), RegistrationDate =
DateTime.Now, Login = "vladdy", Password = "moses", Email = "vladdy@yandex.ru", PhoneNumber
= "34-20-23"}
    );

    db.SaveChanges();

    db.Secretaries.AddOrUpdate(
        n => n.Surname,
        new Secretary { Name = "Евгений", Surname = "Прохоров", FatherName = "Эдуар-
дович", Sex = true, BirthDate = DateTime.Now.AddYears(-19), RegistrationDate = DateTime.Now,
Login = "evgwed", Password = "qwerty", Email = "evgeni@yandex.ru", IndividualTaxNumber =
"9089605302943"}
    );

    db.SaveChanges();

    db.Ecologists.AddOrUpdate(
        n => n.Surname,
        new Ecologist { Name = "Алексей", Surname = "Харитонов", FatherName = "Ива-
нович", Sex = true, BirthDate = DateTime.Now.AddYears(-17), RegistrationDate = DateTime.Now,
Login = "demarvel", Password = "hariton", Email = "demarvel@yandex.ru", InterestsSphere =
"Животные и растения. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и вэб-
дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В
то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, исполь-
зуя Lorem Ipsum для распечатки образцов.", Education = "УлГТУ. Факультет информационных сис-
тем и технологий. Высшее", DistrictLocation = "Калининградская область"},
        new Ecologist { Name = "Игорь", Surname = "Муравьев", FatherName = "Игоре-
вич", Sex = true, BirthDate = DateTime.Now.AddYears(-18), RegistrationDate = DateTime.Now,
Login = "muravei", Password = "muravei", Email = "muravey@yandex.ru", InterestsSphere =
"Океан, море и озера. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и вэб-
дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В
то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов, исполь-
зуя Lorem Ipsum для распечатки образцов.", Education = "УлГТУ. Факультет информационных сис-
тем и технологий. Высшее", DistrictLocation = "Краснодарский край"},
        new Ecologist { Name = "Сергей", Surname = "Смеречинский", FatherName =
"Орестович", Sex = true, BirthDate = DateTime.Now.AddYears(-20), RegistrationDate =
DateTime.Now, Login = "sergey", Password = "sergey", Email = "smerechinskiy@yandex.ru",
InterestsSphere = "Интересуюсь многими сферами биологии и экологии. Lorem Ipsum - это текст-
\\"рыба\\"", часто используемый в печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой
для текстов на латинице с начала XVI века. В то время некий безымянный печатник создал боль-

```

```

шую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для распечатки образцов.",
Education = "УлГТУ. Факультет информационных систем и технологий. Высшее", DistrictLocation
= "Смоленская область"},
    new Ecologist { Name = "Дмитрий", Surname = "Ефимов", FatherName = "Дмитрие-
вич", Sex = true, BirthDate = DateTime.Now.AddYears(-21), RegistrationDate = DateTime.Now,
Login = "dima", Password = "dima", Email = "dimaefimov@yandex.ru", InterestsSphere = "Горы.
Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и вэб-дизайне. Lorem Ipsum
является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий безы-
мянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum для
распечатки образцов.", Education = "УлГТУ. Факультет информационных систем и технологий.
Высшее", DistrictLocation = "Кавказ"},
    new Ecologist { Name = "Иван", Surname = "Загайчук", FatherName = "Анатолие-
вич", Sex = true, BirthDate = DateTime.Now.AddYears(-19), RegistrationDate = DateTime.Now,
Login = "zagaichuk", Password = "ziga", Email = "zagaichuk@yandex.ru", InterestsSphere =
"Все. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и вэб-дизайне. Lorem
Ipsum является стандартной рыбой для текстов на латинице с начала XVI века. В то время некий
безымянный печатник создал большую коллекцию размеров и форм шрифтов, используя Lorem Ipsum
для распечатки образцов.", Education = "УлГТУ. Факультет информационных систем и технологий.
Высшее", DistrictLocation = "Урал"}
);

db.SaveChanges();

db.RankUsers.AddOrUpdate(
    n => n.Surname,
    new RankUser { Name = "Равиль", Surname = "Альмяшев", FatherName =
"Камилевич", Sex = true, BirthDate = DateTime.Now.AddYears(-19), RegistrationDate =
DateTime.Now, Login = "ravil", Password = "lalka", Email = "zagaichuk@yandex.ru", Infor-
mation = "Я простой юзер!" },
    new RankUser { Name = "Петр", Surname = "Сергеев", FatherName = "Сергеевич",
Sex = true, BirthDate = DateTime.Now.AddYears(-19), RegistrationDate = DateTime.Now, Login =
"serg", Password = "serg", Email = "zagaichuk@yandex.ru", Information = "Я простой юзер!" }
);

db.SaveChanges();

db.Partners.AddOrUpdate(
    n => n.CompanyName,
    new Partner { Name = "Анна", Surname = "Голобокова", FatherName =
"Андреевна", Sex = false, Email = "golobokova_ann@gmail.ru", Login = "ann", Password =
"ann", BirthDate = DateTime.Now.AddYears(-20), RegistrationDate = DateTime.Now, CompanyName
= "AnnCompanyGroup", Address = "Ульяновск", Description = "Мы компания, занимающаяся помощью
природе.", Reason = "Хотим сделать мир лучше. Мы очень богатые", IsSolved = true, Secretary
= db.Secretaries.First(n => n.Surname == "Прохоров") }
);

db.SaveChanges();

db.Complaints.AddOrUpdate(
    n => n.Title,
    new Complaint { Title = "Загрязнение Амура", Creator =
db.Ecologists.First(), AppearingDate = DateTime.Now, Description = "Часто используемый в
печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века.", IsHidden = true },
    new Complaint { Title = "Загрязнение озера Байкал", Creator =
db.Ecologists.First(), AppearingDate = DateTime.Now, Description = "Часто используемый в
печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века.", IsHidden = true },
    new Complaint { Title = "Лесные пожары на Алтае", Creator =
db.Ecologists.First(), AppearingDate = DateTime.Now, Description = "Часто используемый в
печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века.", IsHidden = true },
    new Complaint { Title = "Выбросы в атмосферу на Кавказе", Creator =
db.Ecologists.First(), AppearingDate = DateTime.Now, Description = "Часто используемый в

```

```

печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века.", IsHidden = false },
    new Complaint { Title = "Лесные пожары в Сибири", Creator =
db.Ecologists.First(), AppearingDate = DateTime.Now, Description = "Часто используемый в
печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века.", IsHidden = true }
);

db.SaveChanges();

db.EcologicalProblems.AddOrUpdate(
    n => n.Title,
    new EcologicalProblem { Title = "Загрязнение реки Амур", Description = "Про-
блема загрязнения реки Амур. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и
вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI
века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов,
используя Lorem Ipsum для распечатки образцов.", RequiredSum = 2000000, IsSolved = false,
PublicationDate = DateTime.Now.AddYears(-3), PhotoFile = null, PhotoType = string.Empty,
Creator = db.Ecologists.First(n => n.Surname == "Харитонов"), Complaint =
db.Complaints.First(n => n.Title == "Загрязнение Амура")},
    new EcologicalProblem { Title = "Загрязнение озера Байкал", Description =
"Проблема загрязнения озера Байкал. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в
печати и вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с нача-
ла XVI века. В то время некий безымянный печатник создал большую коллекцию размеров и форм
шрифтов, используя Lorem Ipsum для распечатки образцов.", RequiredSum = 2000000, IsSolved =
false, PublicationDate = DateTime.Now.AddYears(-3), PhotoFile = null, PhotoType =
string.Empty, Creator = db.Ecologists.First(n => n.Surname == "Ефимов"), Complaint =
db.Complaints.First(n => n.Title == "Загрязнение озера Байкал") },
    new EcologicalProblem { Title = "Лесные пожары на Алтае", Description =
"Проблема лесных пожаров. Lorem Ipsum - это текст-\\"рыба\\"", часто используемый в печати и
вэб-дизайне. Lorem Ipsum является стандартной рыбой для текстов на латинице с начала XVI
века. В то время некий безымянный печатник создал большую коллекцию размеров и форм шрифтов,
используя Lorem Ipsum для распечатки образцов.", RequiredSum = 2000000, IsSolved = true,
PublicationDate = DateTime.Now.AddYears(-3), PhotoFile = null, PhotoType = string.Empty,
Creator = db.Ecologists.First(n => n.Surname == "Ефимов"), Complaint = db.Complaints.First(n
=> n.Title == "Лесные пожары на Алтае") },
    new EcologicalProblem { Title = "Лесные пожары в Сибири", Description =
"Лесной пожар в Сибири. Срочно требуется помощи фонда \\"Слон\\"", Идем на помощь", RequiredSum
= 100000000, IsSolved = true, PublicationDate = DateTime.Now.AddYears(-1), PhotoFile = null,
PhotoType = string.Empty, Creator = db.Ecologists.First(n => n.Surname == "Смеречинский"),
Complaint = db.Complaints.First(n => n.Title == "Лесные пожары в Сибири")}
);

db.SaveChanges();

db.Sections.AddOrUpdate(
    n => n.Title,
    new Section { Title = "Общий курс экологии. Часть 1.", SpotsCount = 40,
StartLessonsTime = DateTime.Now.AddDays(5), Ecologist = db.Ecologists.First(n => n.Surname
== "Загайчук"), Description = "Очень интересеный курс. Подойдет для тех кто только начинает-
ся знакомиться с экологией. Ведет просто замечательный преподаватель", LessonsCount = 15,
Participants = db.RankUsers.ToList() },
    new Section { Title = "Общий курс экологии. Часть 2.", SpotsCount = 20,
StartLessonsTime = DateTime.Now.AddDays(10), Ecologist = db.Ecologists.First(n => n.Surname
== "Харитонов"), Description = "Очень интересеный курс. Продолжение предыдущего курса. По-
дойдет для продвинутых пользователей. Ведет также просто замечательный преподаватель",
LessonsCount = 10 },
    new Section { Title = "Общий курс экологии. Часть 3.", SpotsCount = 15,
StartLessonsTime = DateTime.Now.AddDays(15), Ecologist = db.Ecologists.First(), Description
= "Очень интересеный курс. Продолжение предыдущего курса. Подойдет для супер-продвинутых
пользователей. Ведет также просто замечательный преподаватель", LessonsCount = 5 }
);

db.SaveChanges();

```

```

foreach (var s in db.Sections)
{
    //s.CalculateParticipantsCount();
    s.CalculateFreeSpots();
}

db.SaveChanges();

db.Achivements.AddOrUpdate(
    n => n.Title,
    new Achievement { Title = "Пожар в Сибири потушен!", Description = "При не-
посредственной поддержке фонда был потушен большой пожар в Сибирских лесах. Фонд удостоился
правительственной награды.", EcologicalProblem = db.EcologicalProblems.First(n => n.Title ==
"Лесные пожары в Сибири"), Administrator = db.Administrators.First(), PhotoFile = null,
PhotoType = string.Empty },
    new Achievement { Title = "Пожар на Алтае потушен!", Description = "При не-
посредственной поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился
правительственной награды.", EcologicalProblem = db.EcologicalProblems.First(n => n.Title ==
"Лесные пожары на Алтае"), Administrator = db.Administrators.First(n => n.Surname == "Моисе-
ев"), PhotoFile = null, PhotoType = string.Empty },
    new Achievement { Title = "Река Амур!", Description = "При непосредственной
поддержке фонда был потушен большой пожар в Алтайских лесах. Фонд удостоился правительствен-
ной награды.", EcologicalProblem = db.EcologicalProblems.First(n => n.Title == "Загрязнение
реки Амур"), Administrator = db.Administrators.First(n => n.Surname == "Моисеев"), PhotoFile
= null, PhotoType = string.Empty }
);

db.SaveChanges();

db.Complaints.AddOrUpdate(
    n => n.Title,
    new Complaint { Title = "Жалоба1", AppearingDate = DateTime.Now, Creator =
db.Ecologists.First(), Description = "description1111", IsHidden = false }
);

db.SaveChanges();

db.Councils.AddOrUpdate(
    n => n.Title,
    new Council { Title = "Разбор пожара в Сибири", AssignmentDate =
DateTime.Now.AddMonths(-10), Description = "На данном совете будет рассматриваться проблема
пожара в Сибири", CouncilResult = true, Ecologists = db.Ecologists.ToList(), Problem =
db.EcologicalProblems.First(n => n.Title == "Лесные пожары в Сибири") }
);

db.SaveChanges();

//Организации-штрафники
db.OrganisationDeptors.AddOrUpdate(
    n => n.Name,
    new OrganisationDeptor { Name = "Петр Петрович", Reason = "Поджог леса в
Сибири", PayTime = DateTime.Now.AddDays(100), FineAmount = 10000000, IsPaid = false, Com-
plaint = db.Complaints.First(n => n.Title == "Лесные пожары в Сибири"), Email =
"petr@email.ru", ResponsiblePerson = db.Secretaries.First() }
);

db.SaveChanges();

DbCommand cmd = null;
db.Database.Connection.Open();
cmd = db.Database.Connection.CreateCommand();

```

//1. Хранимая процедура - возвращает должников, до срока выплаты которых остается @day_count дней

```
cmd.CommandText = @"IF EXISTS
    (SELECT * FROM sys.objects
      WHERE object_id = OBJECT_ID(N'[dbo].[GetCrucialDebtor]'))
AND type in (N'P', N'PC'))
    DROP PROCEDURE [dbo].[GetCrucialDebtor]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"
CREATE PROCEDURE [dbo].[GetCrucialDebtor]
    @day_count int
AS
BEGIN
    BEGIN TRANSACTION;
    SET NOCOUNT ON;
    SELECT * FROM [dbo].[OrganisationDebtors] WHERE (PayTime
<= DATEADD(DAY, @day_count, GETDATE())) AND IsPaid = 0)
    COMMIT TRANSACTION;
END
";
cmd.ExecuteNonQuery();
```

//2. Хранимая процедура - возвращает экологические проблемы, опубликованные в течение указанного числа дней

```
cmd.CommandText = @"IF EXISTS
    (SELECT * FROM sys.objects
      WHERE object_id = OBJECT_ID(N'[dbo].[GetEcologicalProblems]'))
AND type in (N'P', N'PC'))
    DROP PROCEDURE [dbo].[GetEcologicalProblems]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"CREATE PROCEDURE [dbo].[GetEcologicalProblems]
    @days_range int
AS
BEGIN
    BEGIN TRANSACTION;
    SET NOCOUNT ON;
    SELECT * FROM [dbo].[EcologicalProblems] WHERE
(PublicationDate >= DATEADD(DAY, -@days_range, GETDATE()))
    COMMIT TRANSACTION;
END
";
cmd.ExecuteNonQuery();
```

//3. Скалярная функция (возвращает число совершеннолетних пользователей)

```
cmd.CommandText = @"IF EXISTS (SELECT *
    FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[GetAdultsQuantity]')
    AND type IN ( N'FN', N'IF', N'TF', N'FS', N'FT' ))
    DROP FUNCTION [dbo].[GetAdultsQuantity]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"CREATE FUNCTION [dbo].[GetAdultsQuantity]
(
)
RETURNS int
AS
BEGIN
    DECLARE @adults int;
    SELECT @adults = COUNT(*)
    FROM [dbo].[Users] WHERE BirthDate <= DATEADD(YEAR, -
18, GETDATE())

    RETURN @adults
```

```

        END";

cmd.ExecuteNonQuery();

//4. Скалярная функция (возвращает число несовершеннолетних пользователей)
cmd.CommandText = @"IF EXISTS (SELECT *
    FROM sys.objects
    WHERE object_id = OBJECT_ID(N'[dbo].[GetChildrenQuantity]')
    AND type IN ( N'FN', N'IF', N'TF', N'FS', N'FT' ))
    DROP FUNCTION [dbo].[GetChildrenQuantity]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"CREATE FUNCTION [dbo].[GetChildrenQuantity]
(
)
RETURNS int
AS
BEGIN
    DECLARE @children int;
    SELECT @children = COUNT(*)
    FROM [dbo].[Users] WHERE BirthDate >= DATEADD(YEAR, -
18, GETDATE())

    RETURN @children
END";

cmd.ExecuteNonQuery();

//Триггеры
//1 Рассчитывает число участников секции, количество свободных мест после реги-
страции
cmd.CommandText = @"IF EXISTS (SELECT * FROM sys.triggers
    WHERE parent_class = 1 AND name = 'CalculateSpotsForInsert')
    DROP TRIGGER [dbo].[CalculateSpotsForInsert]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"CREATE TRIGGER [dbo].[CalculateSpotsForInsert]
    ON [dbo].[SectionRankUsers]
    AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE dbo.Sections SET ParticipantsCount = (SELECT
COUNT(*) FROM [dbo].[SectionRankUsers] WHERE [dbo].[SectionRankUsers].Section_SectionID =
SectionId) WHERE SectionId IN (SELECT i.Section_SectionId FROM inserted AS i)
    UPDATE dbo.Sections SET FreeSpotsCount = SpotsCount -
(SELECT COUNT(*) FROM [dbo].[SectionRankUsers] WHERE
[dbo].[SectionRankUsers].Section_SectionID = SectionId) WHERE SectionId IN (SELECT
i.Section_SectionId FROM inserted AS i)
END";
cmd.ExecuteNonQuery();

//2 Рассчитывает число участников секции, число свободных мест после дерегистра-
ции
cmd.CommandText = @"IF EXISTS (SELECT * FROM sys.triggers
    WHERE parent_class = 1 AND name = 'CalculateSpotsForDelete')
    DROP TRIGGER [dbo].[CalculateSpotsForDelete]";
cmd.ExecuteNonQuery();

cmd.CommandText = @"CREATE TRIGGER [dbo].[CalculateSpotsForDelete]
    ON [dbo].[SectionRankUsers]
    AFTER DELETE
AS

```

```

        BEGIN
            SET NOCOUNT ON;
            UPDATE dbo.Sections SET ParticipantsCount = (SELECT
COUNT(*) FROM [dbo].[SectionRankUsers] WHERE [dbo].[SectionRankUsers].Section_SectionID =
SectionId) WHERE SectionId IN (SELECT i.Section_SectionId FROM deleted AS i)
            UPDATE dbo.Sections SET FreeSpotsCount = SpotsCount -
(SELECT COUNT(*) FROM [dbo].[SectionRankUsers] WHERE
[dbo].[SectionRankUsers].Section_SectionID = SectionId) WHERE SectionId IN (SELECT
i.Section_SectionId FROM deleted AS i)
            END";
        cmd.ExecuteNonQuery();
    }
}

```

Программный код FundContext.cs:

```

using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;

namespace FundApp.Models
{
    public class FundContext : DbContext
    {
        public FundContext()
            : base("DefaultConnection")
        {
        }

        public DbSet<User> Users { get; set; }
        public DbSet<RankUser> RankUsers { get; set; }
        public DbSet<Secretary> Secretaries { get; set; }
        public DbSet<Administrator> Administrators { get; set; }
        public DbSet<Ecologist> Ecologists { get; set; }

        public DbSet<Achievement> Achivements { get; set; }
        public DbSet<Complaint> Complaints { get; set; }
        public DbSet<Council> Councils { get; set; }
        public DbSet<EcologicalProblem> EcologicalProblems { get; set; }
        public DbSet<OrganisationDeptor> OrganisationDeptors { get; set; }
        public DbSet<Partner> Partners { get; set; }
        public DbSet<PartnershipRequest> PartnershipRequests { get; set; }
        public DbSet<Section> Sections { get; set; }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            base.OnModelCreating(modelBuilder);
        }
    }
}

```