Министерство транспорта Российской Федерации  
Федеральное агентство железнодорожного транспорта  
ФГБОУ ВО «Дальневосточный государственный университет путей   
сообщения»

Кафедра «Информационные технологии  
и системы»

Курсовой проект

по дисциплине «Методы проектирования информационных систем»  
Тема: «Курьерская служба»

Выполнил: Козеко М. В.  
студент гр.СО251КОБ

Проверил: Анисимов В.В.

Хабаровск

2020 г.

Оглавление

[1. Описание предметной области 3](#_Toc51612163)

[2. Модель вариантов использования 4](#_Toc51612164)

[2.1 Диаграммы вариантов использования 5](#_Toc51612165)

[2.2 Диаграммы автоматов 10](#_Toc51612166)

[3. Модель анализа 14](#_Toc51612167)

[3.1 Диаграмма классов анализа 14](#_Toc51612168)

[3.2 Диаграммы последовательности 18](#_Toc51612169)

[3.3 Диаграммы коммуникации 21](#_Toc51612170)

[4. Модель проектирования 24](#_Toc51612171)

[4.1 Диаграммы классов 24](#_Toc51612172)

[4.2 Диаграммы деятельности 30](#_Toc51612173)

[5. Модель реализации 35](#_Toc51612174)

[5.1 Диаграммы компонентов 35](#_Toc51612175)

[5.2 Диаграмма развертывания 39](#_Toc51612176)

[6. Сгенерированный программный код 40](#_Toc51612177)

[7. Руководство пользователя 41](#_Toc51612178)

[7.1 Авторизация 41](#_Toc51612179)

[7.2 Регистрация 42](#_Toc51612180)

[7.3 Главная страница 43](#_Toc51612181)

[7.4 Справочные материалы 45](#_Toc51612182)

[7.5 Построение схем 45](#_Toc51612183)

[7.6 Просмотр статистики 47](#_Toc51612184)

[7.7 Администрирование 47](#_Toc51612185)

[Заключение 49](#_Toc51612186)

[Список используемых источников 50](#_Toc51612187)

# 1. Описание предметной области

В связи с переходом к информационному обществу, в котором в результате процессов информатизации и компьютеризации информационные технологии играют важнейшую роль во всех сферах человеческой деятельности, проблема развития информационных технологий становится одной из наиболее актуальных в современном мире.

Целью данного курсового проекта является разработка информационной системы «Социальная сеть».

В свою очередь основной целью создания системы является:  онлайн-платформа, которая используется для общения, знакомств, создания социальных отношений между людьми, которые имеют схожие интересы или офлайн-связи, также для развлечения.

Требования заказчика к информационной системе:

* Пользователи должны иметь доступ к приложению средствами браузера (Google Chrome, Mozilla Firefox, Microsoft Edge и другие.);
* Кроссплатформенность (macOS, Windows, Linux);
* Простота в развертывании и администрировании системы;
* Возможность взаимодействия между пользователями системы путем сообщений;
* Отдельные учётные записи и права доступа для каждого пользователя, исходя из его должности.

На основе требований заказчика платформой для информационной системы было выбрано веб-приложение с использованием архитектуры MVC. Model-View-Controller (MVC, «Модель-Представление-Контроллер», «Модель-Вид-Контроллер») — схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

Технологии задействованы в работе информационной системы:

* ADO.NET;
* MS SQL Server.

Проектирование информационной системы выполняется на основе объектно-ориентированного подхода и языка UML (Unified modeling language).

# 2. Модель вариантов использования

Модель вариантов использования — это модель, описывающая взаимодействие пользователей и системы между собой, для решения поставленных задач. Данная модель описывает цели пользователей, поведение системы, а также взаимодействие пользователей с системой или между собой.

Главной целью при разработке этой модели является достижение максимального взаимопонимания между разработчиками и заказчиками по вопросам назначения, возможностям и технологиям использования ИС.

Достижение этой цели, в первую очередь, достигается за счет разработки диаграмм UML, которые являются основными факторами технологического процесса «Формирование требований».

Унифицированный язык моделирования (UML) в настоящий момент является стандартом де-факто при описании (документирования) результатов проектирования и разработки объектно-ориентированных систем.

Диаграммы, которые включает в себя модель вариантов использования:

* диаграмма вариантов использования;
* диаграмма автоматов.

## 2.1 Диаграммы вариантов использования

Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Диаграмма вариантов использования включает в себя три основных элемента: актеров, варианты использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Актер — это любой объект, субъект или система, взаимодействующая с моделируемой системой извне.

Вариант использования — это спецификация сервисов (функций), которые система предоставляет актеру.

В процессе анализа проектируемой информационной системы было определено три актера:

1. Пользователь — лицо, которое использует действующую систему для общения с другими пользователями онлайн. Если пользователь не зарегистрирован на сайте, то ему доступна лишь форма регистрации. В случае, если пользователь зарегистрирован и авторизован ему доступны: редактирование профиля, личной страницы и сообществ, создателем которых он является. Так же пользователь может добавлять других пользователей в друзья и общаться с данными пользователями посредством личных сообщений. Если у пользователя возникают какие либо вопросы по использованию системы, он может обратиться за помощью в техническую поддержку.
2. Специалист технической поддержки — сотрудник, который обрабатывает обращения от пользователей. Обращения создаются пользователями и включают в себя вопросы, которые возникают у пользователя при работе с системой.
3. Администратор — сотрудник отдела программного и технического обеспечения, который следит за работоспособностью системы, управлением техническими настройками, занимается непосредственной работой с информационной системой. Регистрацией новых специалистов технической поддержки так же занимается администратор.

Так же определены следующие варианты использования:

1. Регистрация пользователя;
2. Редактирование профиля;
3. Редактирование личной страницы;
4. Управление сообществом;
5. Просмотр обращений пользователя
6. Создание обращения в техническую поддержку;
7. Отправка сообщения другому пользователю;
8. Создание сообщества;
9. Поиск друзей;
10. Авторизация;
11. Обработка обращения в техническую поддержку;
12. Просмотр сведений о системе;
13. Просмотр статистики посещений сайта;
14. Просмотр сведений о системе.

На основе перечисленных данных была построена контекстная диаграмма, описывающая общую схему взаимодействия актеров в пределах ИС (рис. 2.1.1).

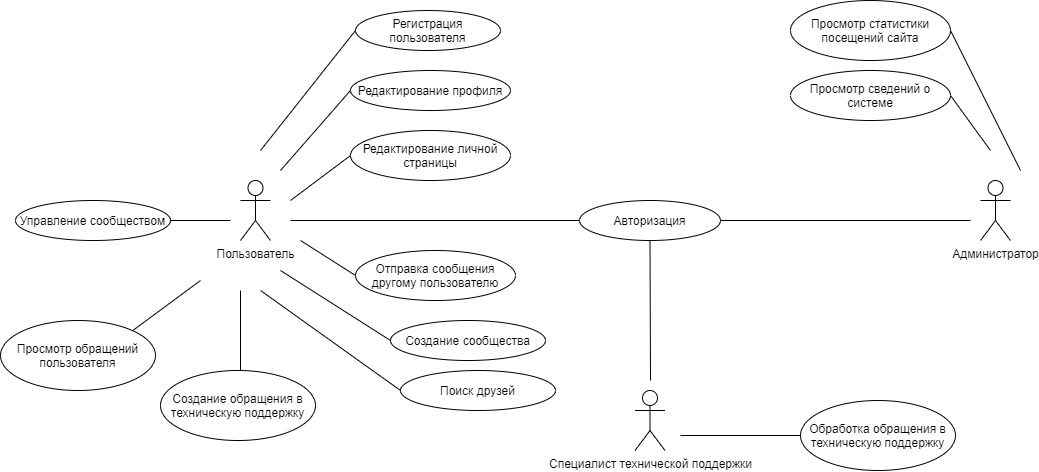


Рисунок 2.1.1. Контекстная диаграмма вариантов использования

Основываясь на контекстную диаграмму, были построены три диаграммы декомпозиций.

Обычно в центре диаграммы декомпозиции располагается декомпозируемый вариант использования, а вокруг – входящие в него обязательные (««include»») или расширяющие (««extend»») составные части.

Рассмотрим декомпозицию варианта использования «Регистрация пользователя» (рис. 2.1.2), на которой в качестве «центрального» актера выступает «Пользователь». В данном разделе пользователь может выполнять следующие действия:

1. Ввод логина;
2. Ввод пароля;
3. Ввод ФИО;
4. Ввод адреса эл. почты;
5. Добавление фото профиля;
6. Ввод данных о месте проживания;
7. Ввод данных о месте рождения;
8. Ввод адреса;
   1. Выбор страны;
   2. Выбор города;
9. Ввод даты рождения;
   1. Ввод даты рождения вручную;
   2. Выбор даты рождения из календаря;
10. Ввод номера телефона;
    1. Ввод рабочего номера;
    2. Ввод домашнего номера.

Для выполнения каких-либо манипуляций в рамках данной диаграммы пользователю необходимо перейти к пункту “Регистрация пользователя”.

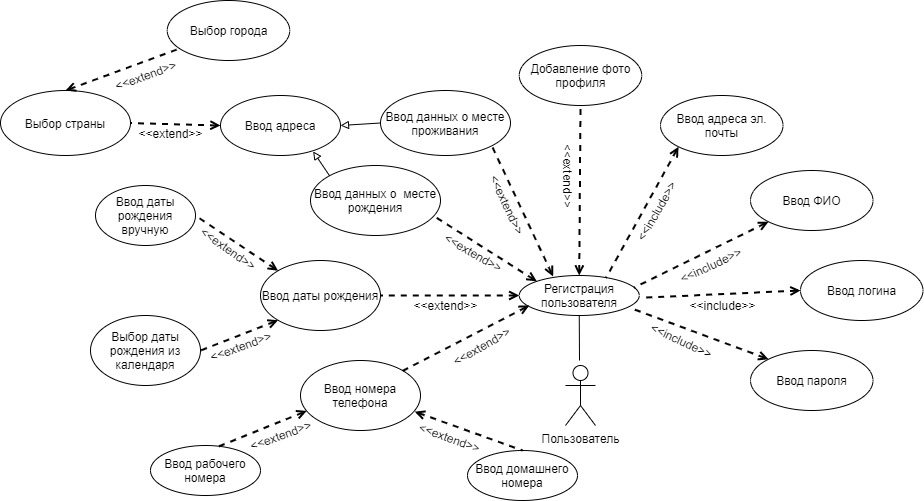


Рисунок 2.1.2. Диаграмма декомпозиции "Регистрация пользователя"

На диаграмме декомпозиции варианта использования «Создание сообщества» (рис. 2.1.3) показаны основные возможности пользователя в данном разделе:

1. Ввод названия сообщества;
2. Выбор темы сообщества;
   1. Выбор темы из списка;
   2. Ввод темы вручную;
3. Ввод описания сообщества;
4. Изменение URL страницы сообщества;
5. Добавление фото сообщества;
6. Выбор модераторов сообщества;
   1. Добавление модератора сообщества из списка всех пользователей;
   2. Добавление модератора сообщества из списка друзей.

Для выполнения каких-либо манипуляций в рамках данной диаграммы пользователю необходимо успешно авторизоваться в информационной системе.

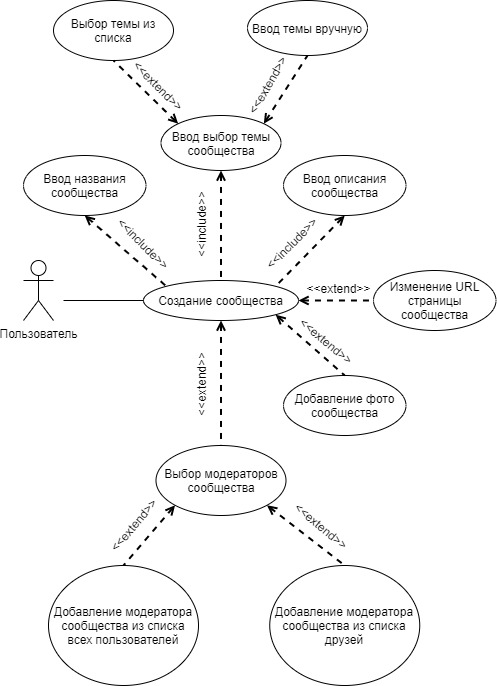


Рисунок 2.1.3. Диаграмма декомпозиции "Создание сообщества"

На диаграмме декомпозиции варианта использования «Поиск друзей» (рис. 2.1.4) показаны основные возможности пользователя в данном разделе:

1. Выбор страны поиска;
   1. Ввод страны вручную;
   2. Выбор страны из списка;
2. Выбор региона поиска;
   1. Ввод региона вручную;
   2. Выбор региона из списка;
3. Ввод ФИО;
4. Ввод возраста;
5. Отправка запроса на добавление в друзья;
6. Выбор пользователя из списка.

Для выполнения каких-либо манипуляций в рамках данной диаграммы пользователю необходимо успешно авторизоваться в информационной системе.

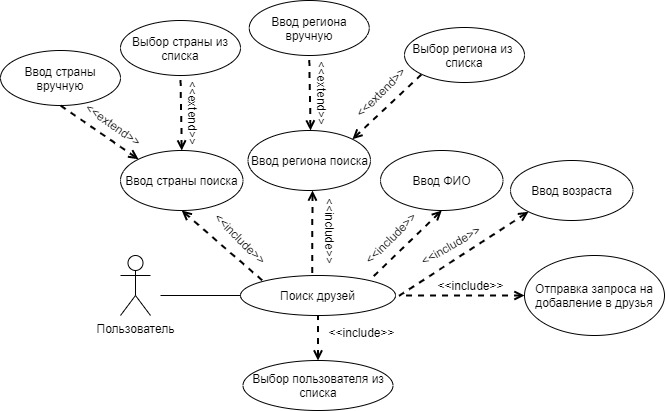


Рисунок 2.1.4. Диаграмма декомпозиции "Поиск друзей"

## 2.2 Диаграммы автоматов

Диаграммы автоматов (англ. state machine) используются для описания поведения, реализуемого в рамках варианта использования, или поведения экземпляра сущности (класса, объекта, компонента, узла или системы в целом). Поведение моделируется через описание возможных состояний экземпляра сущности и переходов между ними на протяжении его жизненного цикла, начиная от создания и заканчивая уничтожением. Диаграмма автоматов представляет собой связный ориентированный граф, вершинами которого являются состояния, а дуги служат для обозначения переходов из состояния в состояние.

Под состоянием (англ. state) понимается ситуация в ходе жизни экземпляра сущности, когда эта ситуация удовлетворяет некоторому условию, экземпляр выполняет некоторые операции или ждет наступления некоторого события.

Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы автоматов могут быть вло­жены друг в друга, образуя вложенные диаграммы более детального пред­ставления отдельных элементов модели.

В связи с тем, что к данной системе будут иметь доступ разные пользователи (администраторы, пользователи и специалисты технической поддержки), которые соответственно обладают разными правами, то у нас появляется необходимость для каждого типа пользователей организовать индивидуальный интерфейс с определёнными правами именно для данного типа. Эта ситуация отражена на контекстной диаграмме автоматов (рис. 2.2.1).

В данном случае, мы видим состояния системы, при ее использовании пользователями, имеющими разные роли, и как следствие, типы доступа к компонентам системы.

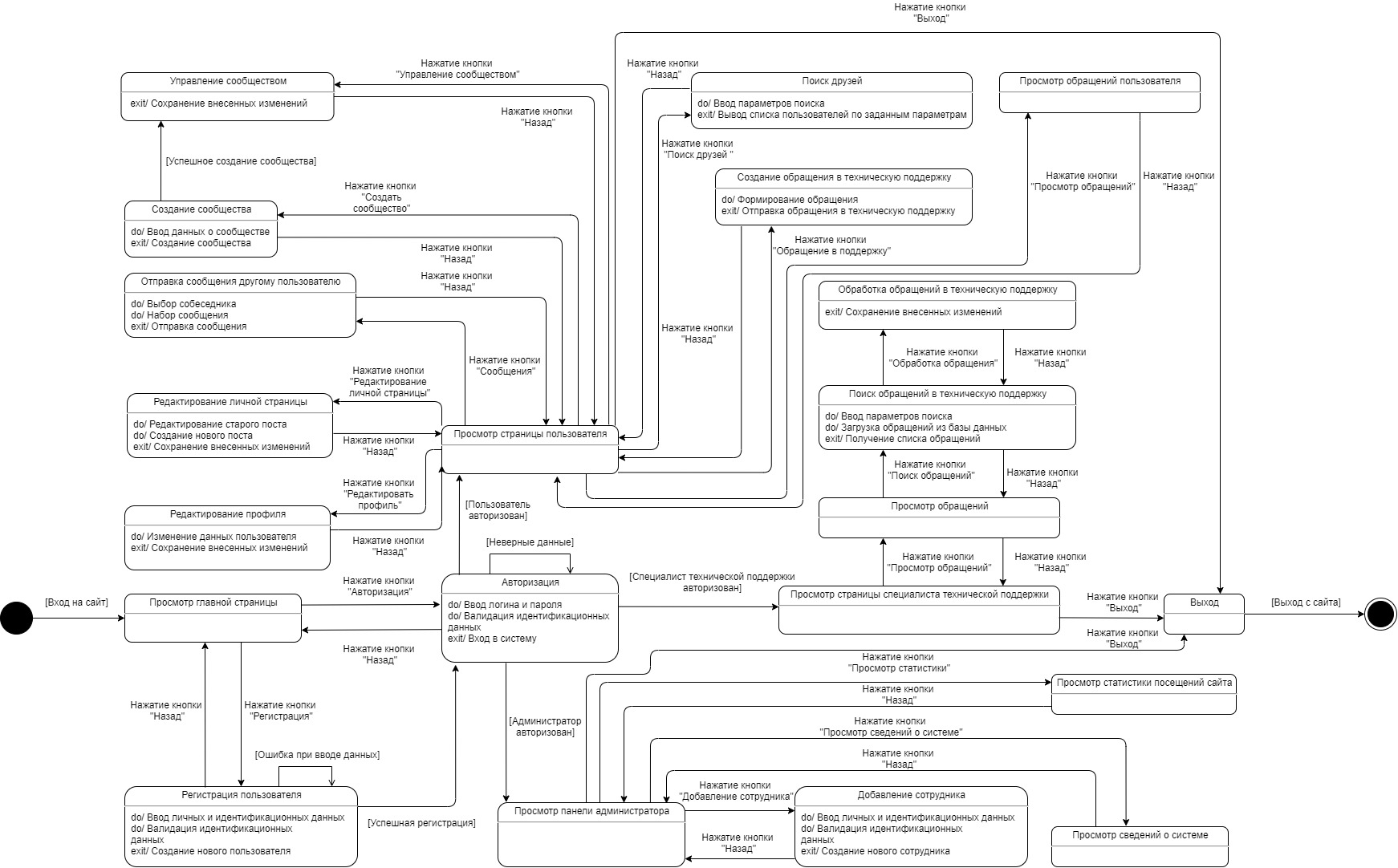


Рисунок 2.2.1. Контекстная диаграмма автоматов.

На диаграмме автоматов подсистемы «Регистрация пользователя» (рис. 2.2.2) отображены состояния системы при регистрации пользователя в системе. Диаграмма более подробно детализирует вариант использования «Регистрация пользователя». Сценарий «Регистрация пользователя» следующий: пользователь переходит на форму регистрации, после чего он вводит логин, пароль и адрес эл. почты. Далее на сервер отправляется запрос о проверке введенных данных пользователя на уникальность и сложность(поле “Пароль”). Если сервер выявил ошибку, пользователь должен исправить введенные данные. Если данные введены успешно, пользователь вводит свои ФИО. Далее пользователь может нажать на кнопку “Дополнительная информация” и приступить к заполнению следующих полей: фото профиля, данные о месте проживания, данные о месте рождения, дата рождения, номер телефона. В не зависимости от того, нажал пользователь кнопку “Дополнительная информация” или нет, он может нажать кнопку “Зарегистрироваться”. После этого система регистрирует пользователя и он используя свои логин и пароль в окне “Авторизация” может войти в систему. В момент регистрации пользователь может нажать кнопку “Назад”, для выхода из окна регистрации.

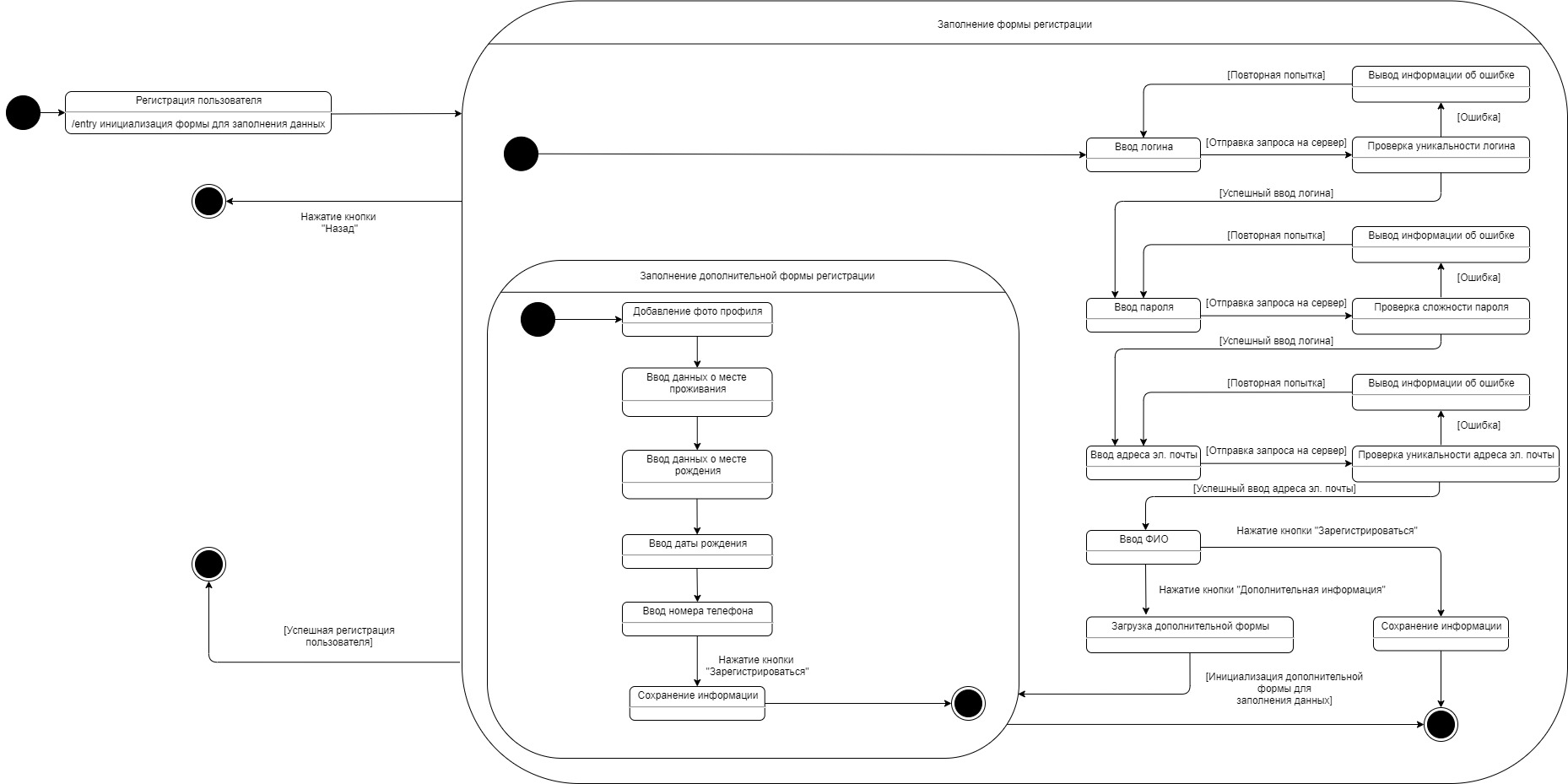


Рисунок 2.2.2 Диаграмма автоматов «Регистрация пользователя»

На диаграмме автоматов подсистемы «Поиск друзей» (рис. 2.2.3) отображены состояния системы при поиске пользователем друзей в системе.. Диаграмма более подробно детализирует вариант использования «Поиск друзей». Сценарий «Поиск друзей» следующий: пользователь открывает окно поиска друзей и приступает к заполнению формы поиска. Пользователь производит заполнение следующих полей: страна поиска, регион поиска, город/населенный пункт, ФИО пользователя, возраст пользователя. После заполнения полей система выводит список пользователей, удовлетворяющих условию поиска. Пользователь выбирает из списка другого пользователя и нажимает кнопку “Добавить в друзья”. Второму пользователю приходит запрос дружбы. Если второй пользователь подтверждает дружбу, текущему пользователю системы приходит уведомление о том, что запрос дружбы подтвержден. В обратном случае текущий пользователь видит сообщение о том, что второй пользователь отклонил запрос.

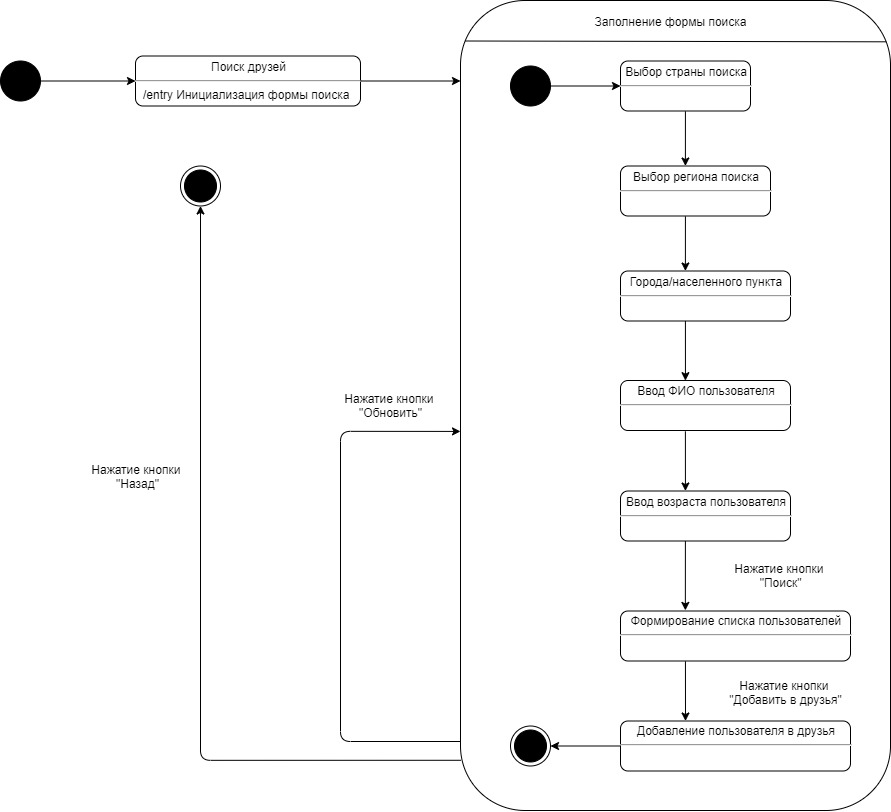


Рисунок 2.2.3 Диаграмма автоматов "Поиск друзей"

# 3. Модель анализа

Главная цель построения модели анализа заключается в уточнении вариантов использования с учетом внутренней архитектуры проектируемой системы.

Построение этой модели необходимо:

* для того чтобы выявить внутренние архитектуры, то есть определить основные классы и подсистемы;
* для поиска альтернативных вариантов реализации системы (подсистем) и выбора основного;
* для уточнения всех требований (функциональных и нефункциональных).

При разработке модели анализа строятся следующие диаграммы:

* классов анализа;
* последовательности;
* коммуникации.

## 3.1 Диаграмма классов анализа

Класс анализа – это укрупненная абстракция, которая на концептуальном уровне (без точного определения атрибутов и операций) описывает некоторый фрагмент системы.

Существует три вида классов анализа:

* граничный;
* управляющий;
* сущности.

Диаграмма классов анализа является прообразом классической диаграммы классов. Элементами, отображаемыми на диаграмме, являются классы и отношения между ними.

Назначение классов анализа:

* граничный класс – используется для моделирования взаимодействия между системой и актерами (пользователями, внешними системами или устройствами);
* управляющий класс – отвечает за координацию, взаимодействие и управление другими объектами, выполняет сложные вычисления, управляет безопасностью, транзакциями и т. п.;
* класс сущности – используется для моделирования долгоживущей, нередко сохраняемой информации. Классы сущности являются абстракциями основных понятий предметной области – людей, объектов, документов и т. д., как правило, хранимых в табличном или ином виде.

Связи между классами анализа отображаются с использованием отношений пяти видов:

* ассоциаций – показывает, что объекты одного класса содержат информацию о существовании (наличии в памяти) объектов другого класса и между ними имеется некоторая логическая или семантическая связь;
* агрегаций – указывает на отношение «часть–целое» и отображается сплошной линией с не закрашенным ромбиком со стороны «целого»;
* композиций – аналогично агрегации, в которой «части» не могут существовать отдельно от «целого»;
* обобщения – является обычным таксонометрическим отношением между более общим (абстрактным) классом (родителем или предком) и его частным случаем (дочерним классом или потомком);
* зависимостей – означает, что в спецификации или теле методов объектов одного класса (зависимого) выполняется обращение к атрибутам, методам или непосредственно к объектам другого класса.

Диаграмма классов анализа изображена на рисунке 3.1.1.

На диаграмме в виде классов отображено клиент-серверное взаимодействие между клиентским приложением и сервером. Соединение обеспечивает некоторый управляющий класс, который позволяет всем граничным классам приложения получать информацию, которая содержится на сервере, и наоборот, отправлять информацию на сервер для дальнейшей ее обработки. Все граничные классы представляют собой View-компоненты, которые отображаются на странице. В свою очередь все сущности представляют собой таблицы в базе данных.

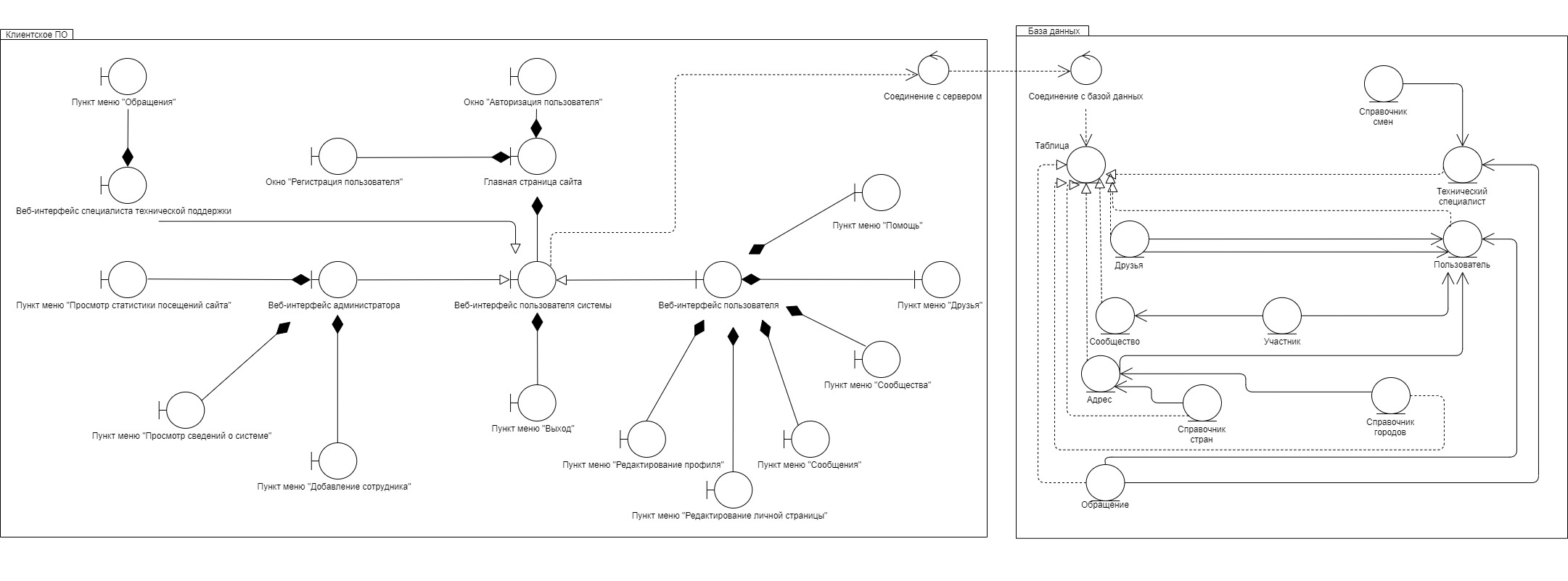


Рисунок 3.1.1 Диаграмма классов анализа

## 3.2 Диаграммы последовательности

Диаграмма последовательности – это одна из разновидностей диаграмм взаимодействия, её назначение заключается в моделирование взаимодействий различных объектов системы во времени, а также в обмене сообщениями между этими объектами.

На диаграмме последовательности изображаются объекты, которые в основном представляют экземпляры класса или сущности, обладающие поведением. Объектами на диаграмме последовательности могут быть пользователи, классы, программные компоненты, а иногда и системы в целом.

Диаграмма последовательности наглядно отображает временной аспект взаимодействия. Она имеет два измерения. Одно измерение (слева-направо) указывает на порядок вовлечения экземпляров сущностей во взаимодействие. Крайним слева на диаграмме отображается экземпляр актера или объект, который является инициатором взаимодействия. Правее отображается другой экземпляр сущности, который непосредственно взаимодействует с первым и т.д. Второе измерение (сверху-вниз) указывает на порядок обмена сообщениями. Начальному моменту времени соответствует самая верхняя часть диаграммы. Масштаб на оси времени не указывается, поскольку диаграмма отображает лишь временную упорядоченность взаимодействия типа «раньше-позже».

Для проектируемой информационной системы были построены две диаграммы последовательностей.

На рисунке 3.2.1 представлена диаграмма последовательности «Регистрация пользователя». Для того чтобы попасть на страницу «Регистрация пользователя» пользователю необходимо выбрать советующий пункт. После чего произойдет получение записей из БД, формирование страницы на сервере и отправка её пользователю.

Далее пользователь вводит логин, пароль и адрес эл. почты. После этого на сервер отправляется запрос о проверке введенных данных пользователя на уникальность(поля “Логин” и “Адрес эл. почты”) и сложность(поле “Пароль”). Если сервер выявил ошибку, пользователь должен исправить введенные данные. Если данные введены успешно, пользователь вводит свои ФИО. Далее пользователь может нажать на кнопку “Дополнительная информация” и приступить к заполнению следующих полей: фото профиля, данные о месте проживания, данные о месте рождения, дата рождения, номер телефона. В не зависимости от того, нажал пользователь кнопку “Дополнительная информация” или нет, он может нажать кнопку “Зарегистрироваться”. После этого система регистрирует пользователя и он, используя свои логин и пароль в окне “Авторизация”, может войти в систему.

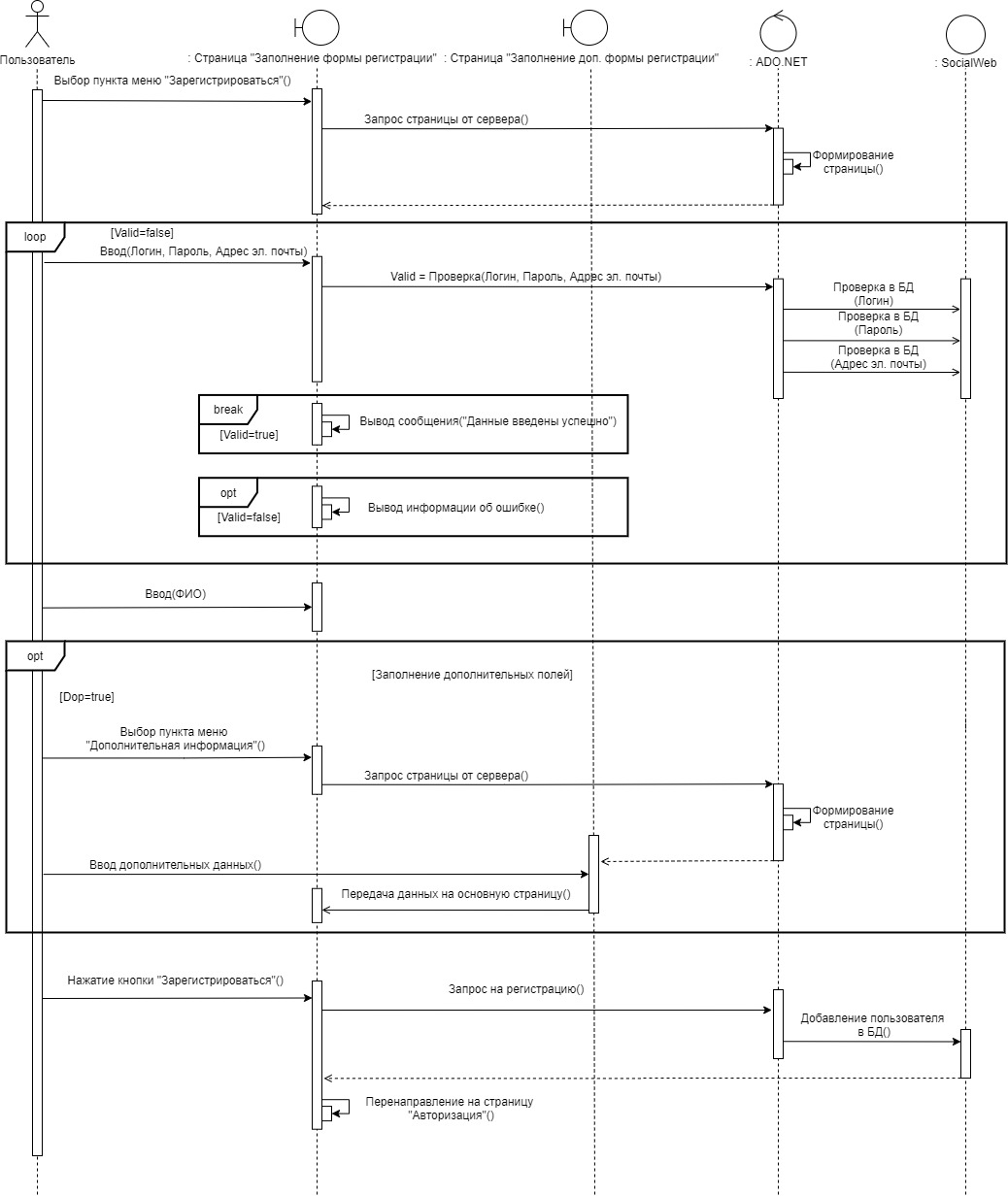


Рисунок 3.2.1. Диаграмма последовательности «Регистрация пользователя»

На рисунке 3.2.2 представлена диаграмма последовательности «Поиск друзей». Чтобы пользователю приступить к поиску друзей, ему необходимо перейти на страницу «Поиск друзей», после чего произойдет формирование страницы на сервере и отправка её клиенту. После этого пользователь приступает к заполнению формы поиска. Пользователь производит заполнение следующих полей: страна поиска, регион поиска, город/населенный пункт, ФИО пользователя, возраст пользователя. После заполнения полей система выводит список пользователей, удовлетворяющих условию поиска. Пользователь выбирает из списка другого пользователя и нажимает кнопку “Добавить в друзья”. Второму пользователю приходит запрос дружбы. Если второй пользователь подтверждает дружбу, текущему пользователю системы приходит уведомление о том, что запрос дружбы подтвержден. В обратном случае текущий пользователь видит сообщение о том, что второй пользователь отклонил запрос. Списки стран и регионов загружаются автоматически при выборе пользователем соответствующих полей.

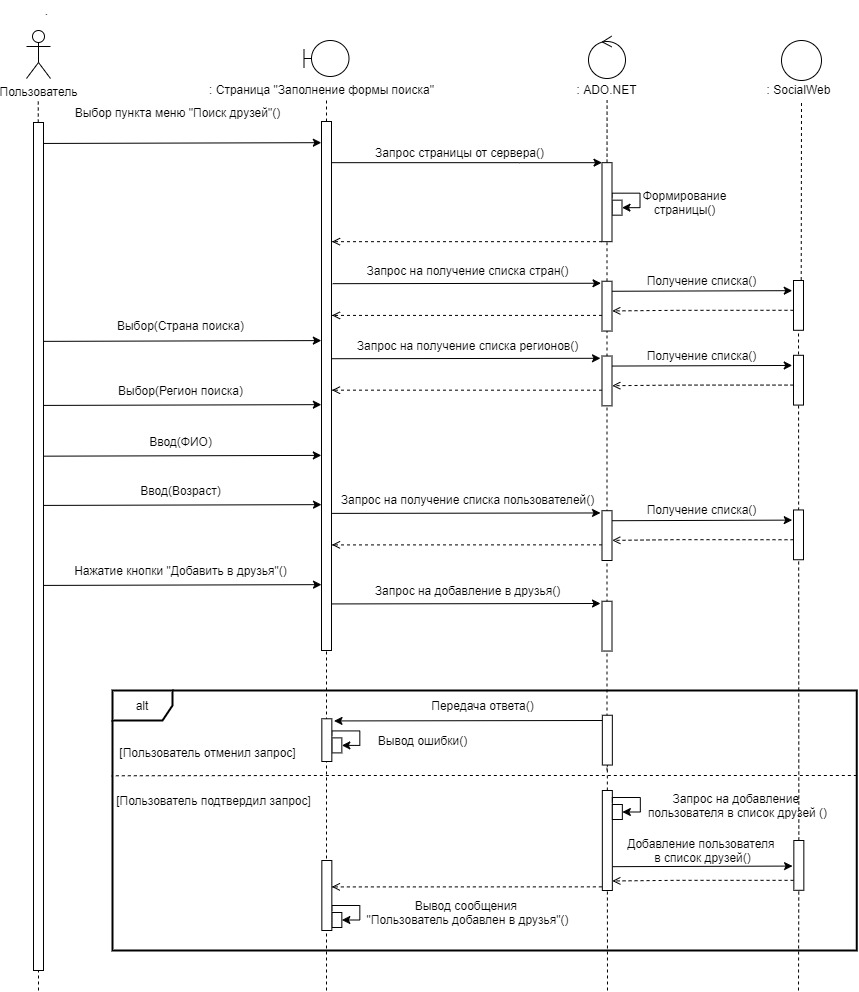


Рисунок 3.2.2. Диаграмма последовательности "Поиск друзей"

## 3.3 Диаграммы коммуникации

В отличие от диаграммы последовательности на диаграмме коммуникации основное внимание уделяется структуре взаимодействия. Помимо общих элементов (экземпляров актеров, объектов и сообщений) между участниками взаимодействия отображаются ненаправленные ассоциации, над которыми указываются передаваемые ими сообщениями. Другой отличительной особенностью является использование в спецификации сообщений нумерации, отражающей порядок их выполнения.

Проектировщикам диаграмма коммуникации может дать богатый материал о распределении обязанностей между объектами. Так, например, если диаграмма напоминает форму звезды, то можно сделать вывод, что система сильно зависит от центрального объекта. В этом случае стоит подумать о более равномерном распределении обязанностей между участниками взаимодействия. Или, наоборот, если в системе хранится и обрабатывается конфиденциальная информация, то большинство сообщений должно проходить через ядро безопасности – классы, отвечающие за идентификацию, аутентификацию и, возможно, шифрование / расшифрование данных.

Таким образом, цель самой коммуникации состоит в том, чтобы специфицировать особенности реализации отдельных наиболее значимых операций в системе. Коммуникация определяет структуру поведения системы.

На рисунке 3.3.1 показана диаграмма коммуникации «Регистрация пользователя». Данная диаграмма сформирована на основе диаграммы последовательности, изображенной на рисунке 3.2.1.

На рисунке 3.3.2 показана диаграмма коммуникации «Поиск друзей». Данная диаграмма сформирована на основе диаграммы последовательности, изображенной на рисунке 3.2.2.



Рисунок 3.3.1. Диаграмма коммуникации «Регистрация пользователя»

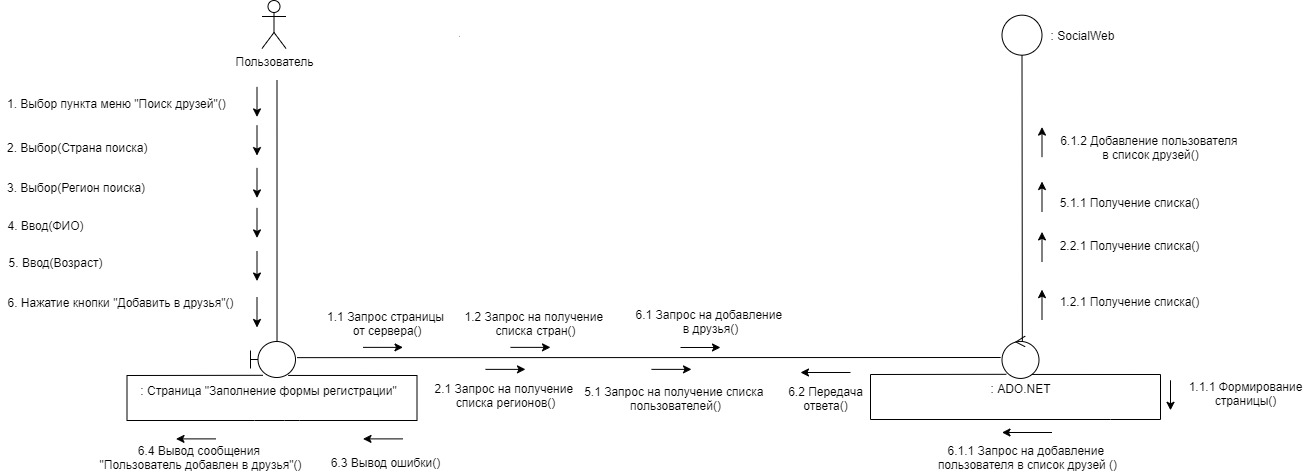


Рисунок 3.3.1. Диаграмма коммуникации «Поиск друзей»

# 4. Модель проектирования

В процессе проектирования создается архитектура системы, которая позволит реализовать и затем поддерживать все функции информационной системы.

Назначение модели проектирования заключается в создании полного детализированного описания внутренней архитектуры и алгоритмов работы системы.

Рекомендуется разрабатывать данную модель без привязки к конкретным языкам программирования, с помощью которых будет создаваться программный продукт, т. е. разрабатывать логическую модель.

Стоит оговориться, что создать модель без оглядки на используемые языки программирования невозможно, но, по крайней мере, необходимо стремиться к этому.

Построение модели проектирования необходимо:

* для уточнения внутренней архитектуры и вариантов использования системы;
* уточнения требований;
* определения детализированных алгоритмов работы системы в целом и ее отдельных элементов.

Модель проектирования представляется диаграммами классов и диаграммами деятельности.

## 4.1 Диаграммы классов

Диаграммы классов используются при моделировании информационных систем наиболее часто. Они являются одной из форм статического описания системы с точки зрения ее проектирования, показывая ее структуру. Диаграмма классов не отображает динамическое поведение объектов, изображенных на ней классов.

На диаграммах классов показываются классы, интерфейсы и отношения между ними.

Диаграмма классов представляет собой граф, вершинами которого являются элементы типа «классификатор», связанные различными типами структурных отношений.

Существуют разные точки зрения на построение диаграмм классов в зависимости от целей их применения:

1. концептуальная точка зрения – диаграмма классов описывает модель предметной области, в ней присутствуют только классы прикладных объектов;
2. точка зрения спецификации – диаграмма классов применяется при проектировании информационных систем;
3. точка зрения реализации – диаграмма классов содержит классы, используемые непосредственно в программном коде.

Классы могут иметь логическую и физическую реализации. Логические диаграммы классов в отличие от физических, строятся без привязки к языкам программирования.

В ходе выполнения курсового проекта было разработано два типа диаграмм классов: для клиентского приложения и для серверного приложения.

Прежде чем перейти к разработке этих двух моделей, опишем структуру таблиц базы данных:

Таблица 1. Адрес (Address)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер адреса | AddressID | Int | int | Уникальный идентификатор адреса |
| Страна | Country | Directory\_Country | int | Страна проживания |
| Город | City | Directory\_City | int | Город проживания |

Таблица 2. Участник (Member)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер участника | MemberID | int | int | Уникальный идентификатор участника |
| Пользователь | User | User | int | Пользователь, участник сообщества |
| Сообщество | Community | Community | int | Сообщество |
| Роль | Role | string | nvarchar(15) | Роль пользователя в сообществе |

Таблица 3. Сообщество(Community)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер сообщества | CommunityID | int | int | Уникальный идентификатор сообщества |
| Название | Name | string | Nvarchar(30) | Название сообщества |
| Тема | Theme | string | Nvarchar(30) | Тема сообщества |
| Описание | Discription | string | Text | Описание сообщества |
| Фото | Photo | image | image | Фотография сообщества |

Таблица 4. Пользователь(User)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер пользователя | UserID | int | int | Уникальный идентификатор пользователя |
| Логин | Login | string | Nvarchar(20) | Логин пользователя |
| Пароль | Password | string | Nvarchar(30) | Пароль пользователя |
| Фамилия | Surname | String | Nvarchar(20) | Фамилия пользователя |
| Имя | Name | String | Nvarchar(20) | Имя пользователя |
| Отчество | Patronymic | String | Nvarchar(20) | Отчество пользователя |
| Эл. Почта | Email | String | Nvarchar(20) | Электронная почта пользователя |
| Фото профиля | UserPhoto | Image | Image | Фото профиля пользователя |

Таблица 5. Технический специалист(Specialist)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер технического специалиста | SpecialistID | int | int | Уникальный идентификатор технического специалиста |
| Логин | Login | string | Nvarchar(20) | Логин технического специалиста |
| Пароль | Password | string | Nvarchar(30) | Пароль технического специалиста |
| Фамилия | Surname | String | Nvarchar(20) | Фамилия технического специалиста |
| Имя | Name | String | Nvarchar(20) | Имя технического специалиста |
| Отчество | Patronymic | String | Nvarchar(20) | Отчество технического специалиста |
| Должность | Position | String | Nvarchar(20) | Должность технического специалиста |
| Номер смены | WorkTime | Directory\_WorkTime | int | Номер рабочей смены специалиста |

Таблица 6. Друзья(Friends)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер записи | RecordID | int | int | Уникальный идентификатор записи о дружбе пользователей |
| Пользователь 1 | User1 | User | Int | Пользователь 1 |
| Пользователь 2 | User2 | User | Int | Пользователь 2 |

Таблица 7. Обращение (Request)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер обращения | RequestID | int | int | Уникальный идентификатор обращения |
| Пользователь | User | User | Int | Пользователь системы |
| Технический специалист | Specialist | Specialist | int | Специалист технической поддержки |
| Дата создания | CreationDate | DateTime | DateTime | Дата создания обращения |
| Дата закрытия | ClosingDate | DateTime | DateTime | Дата закрытия обращения |
| Текст обращения | Message | String | Text | Текст обращения пользователя |

Таблица 8. Справочник смен (Directory\_WorkTime)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер смены | WorkTimeID | int | int | Номер рабочей смены специалиста |
| Смена | WorkTime | string | Nvarchar(20) | Смена специалиста(дневная, вечерняя, ночная) |

Таблица 9. Справочник городов (Directory\_City)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер города | CityID | int | int | Номер города |
| Город | City | string | Nvarchar(20) | Название города |

Таблица 10. Справочник стран (Directory\_Country)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название поля | Название поля на английском | Тип данных в с# | Тип данных в SQL Server | Описание |
| Номер страны | CountryID | int | int | Номер страны |
| Страна | Country | string | Nvarchar(20) | Название страны |

Далее приведены диаграммы классов для БД (рис. 4.1.1 и рис. 4.1.2) и диаграммы классов для приложения (рис. 4.1.3 и рисунок 4.1.4). Каждый вид диаграмм представлен в двух экземплярах: логическом (на русском языке) и физическом (с учетом языка программирования).

Целевой СУБД для описания диаграмм классов БД, является MS SQL Server.

Целевым языком программирования для описания диаграмм классов приложения является C# (ADO.NET).

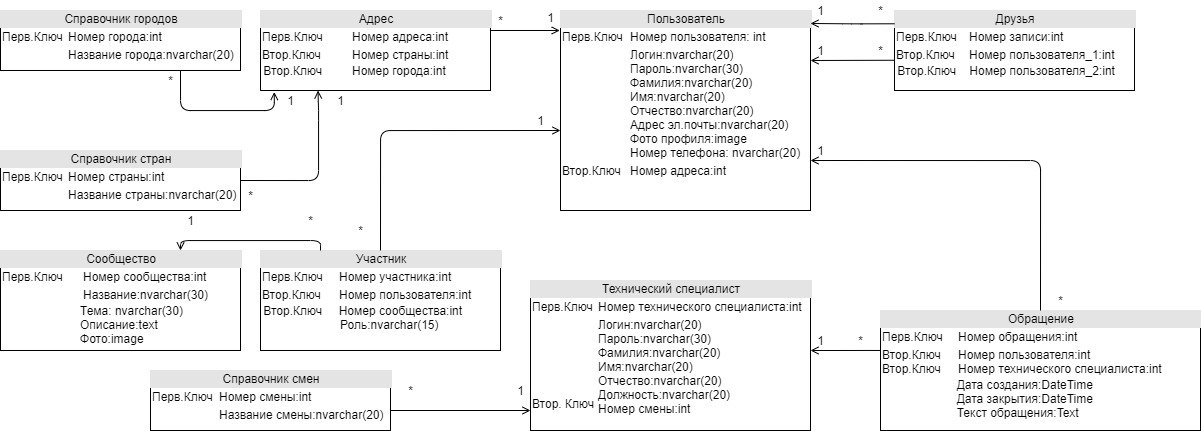


Рисунок 4.1.1. Логическая диаграмма классов базы данных

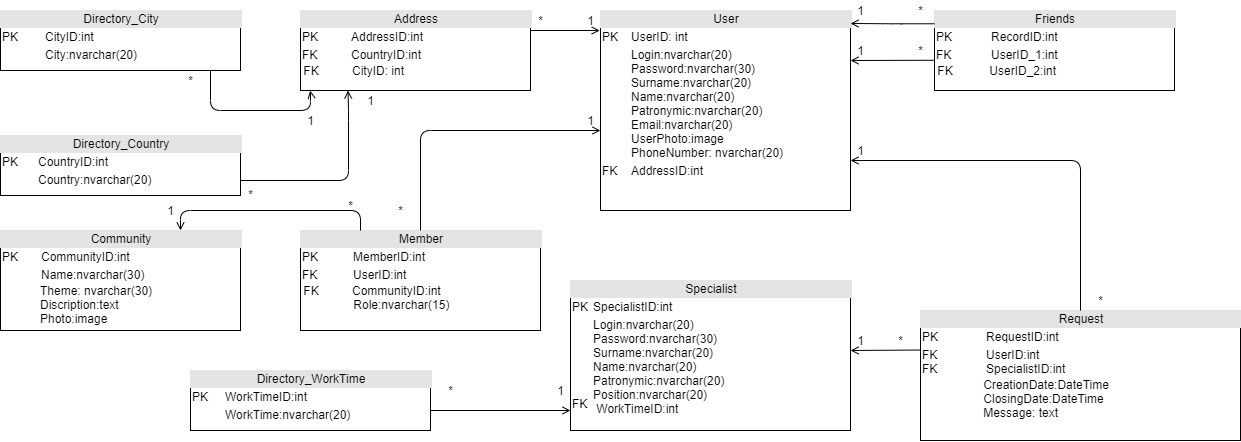


Рисунок 4.1.2. Физическая диаграмма классов базы данных

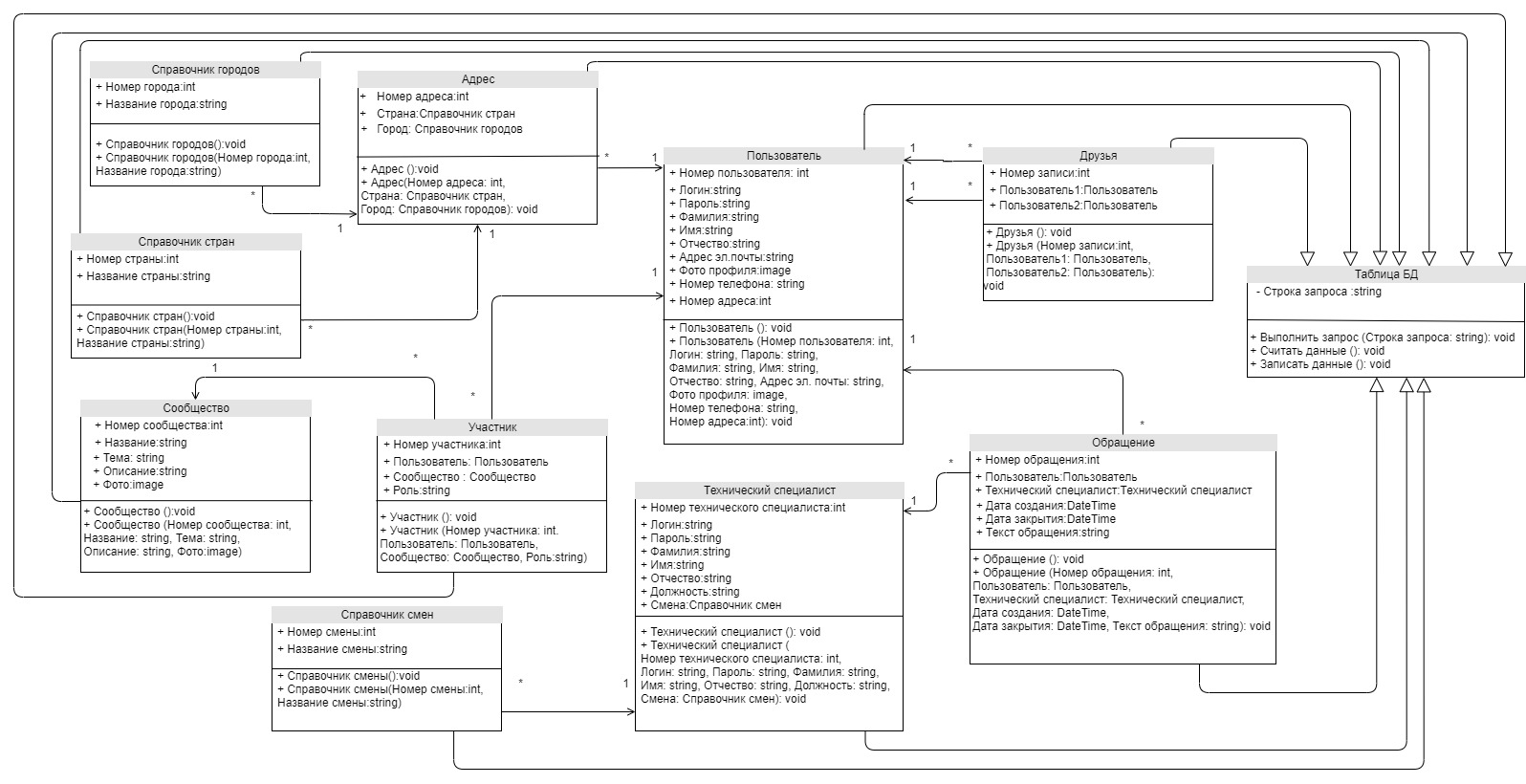


Рисунок 4.1.3. Логическая диаграмма классов приложения

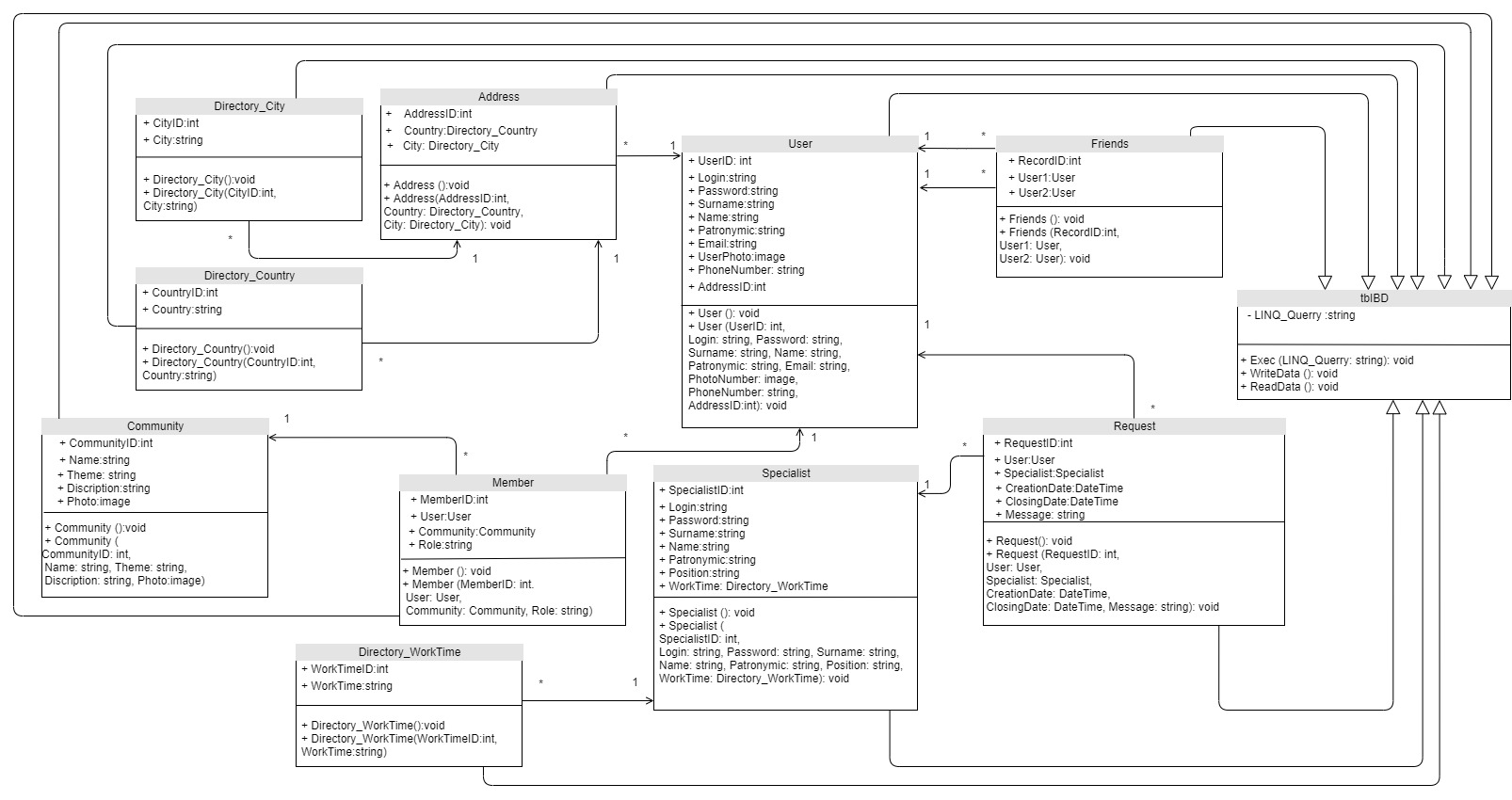


Рисунок 4.1.4. Физическая диаграмма классов приложения

## 4.2 Диаграммы деятельности

При моделировании поведения системы возникает необходимость не только представить процесс изменения ее состояний, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций.

Для описания поведения системы и ее отдельных элементов (поведенческих моделей) в UML предусмотрено четыре вида диаграмм:

* диаграммы автоматов;
* диаграммы последовательности;
* диаграммы коммуникации;
* диаграммы деятельности.

Несмотря на то, что первые три вида диаграмм, так или иначе, отображают динамические аспекты системы, они недостаточно формальны для детального описания алгоритмов работы. В структурном подходе для этого применяются блок-схемы, диаграммы EPC и BPMN. В UML аналогом блок-схем являются диаграммы деятельности (активности), схожие с ними по своей семантике и выразительным средствам (набору элементов).

Каждая диаграмма деятельности акцентирует внимание на последовательности выполнения определенных действий, которые в совокупности приводят к получению желаемого результата. Они могут быть построены для отдельного варианта использования, кооперации, метода и т. д. Диаграммы деятельности являются разновидностью диаграмм автоматов, но если на второй основное внимание уделяется статическим состояниям, то на первой – действиям.

Графически диаграмма деятельности, как и диаграмма автоматов, представляется в виде ориентированного графа, вершинами которого являются действия или деятельности, а дугами – переходы между ними. При этом в UML действие – это атомарная операция, выполнение которой не может быть прервано, а деятельность – составная операция, с возможностью ее прерывания. Переход к следующему действию или деятельности срабатывает сразу по их завершении.

Основными элементами диаграммы являются:

* исполняемые узлы – к исполняемым узлам (англ. executable nodes) относятся действия (англ. action) и деятельности (англ. activity);
* объекты – к объектам относятся непосредственно объекты (англ. object) в традиционном понимании UML, отправка сигнала (англ. send signal), прием сигнала (англ. accept signal) и событие времени (англ. time event);
* переходы – переход (англ. transition или activity edge), как и на диаграмме автоматов, отображается ассоциацией. На диаграммах деятельности различают следующие виды переходов:
  + поток управление;
  + объектный поток;
  + поток прерывания;
  + поток исключения.
* управляющие узлы – управляющим узлам (англ. control nodes) на диаграмме деятельности соответствуют псевдосостояния на диаграмме автоматов;
* коннекторы – коннекторы (англ. connectors) выступают в качестве соединителей, применяемых на блок-схемах;
* группирующие элементы – к группирующим элементам (англ. activity groups) относятся разделы деятельности (англ. activity partitions) и прерываемые регионы (англ. interruptible activity regions).

На рисунке 4.2.1 изображена диаграмма деятельности, описывающая процесс регистрации пользователя.

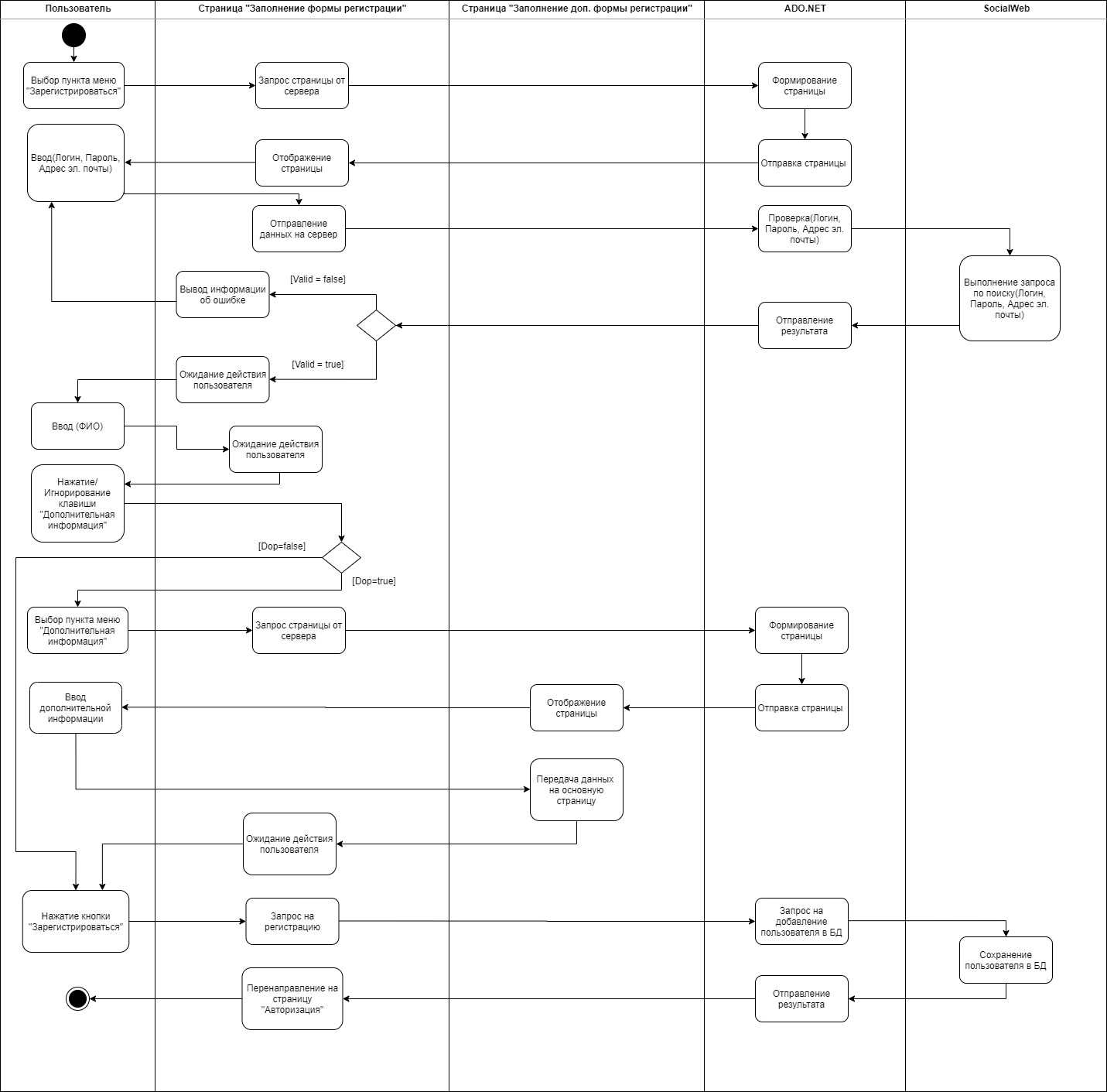
**

Рисунок 4.2.1. Диаграмма деятельности, описывающая процесс регистрации пользователя

На рисунке 4.2.2 изображена диаграмма деятельности, описывающая процесс поиска друзей.

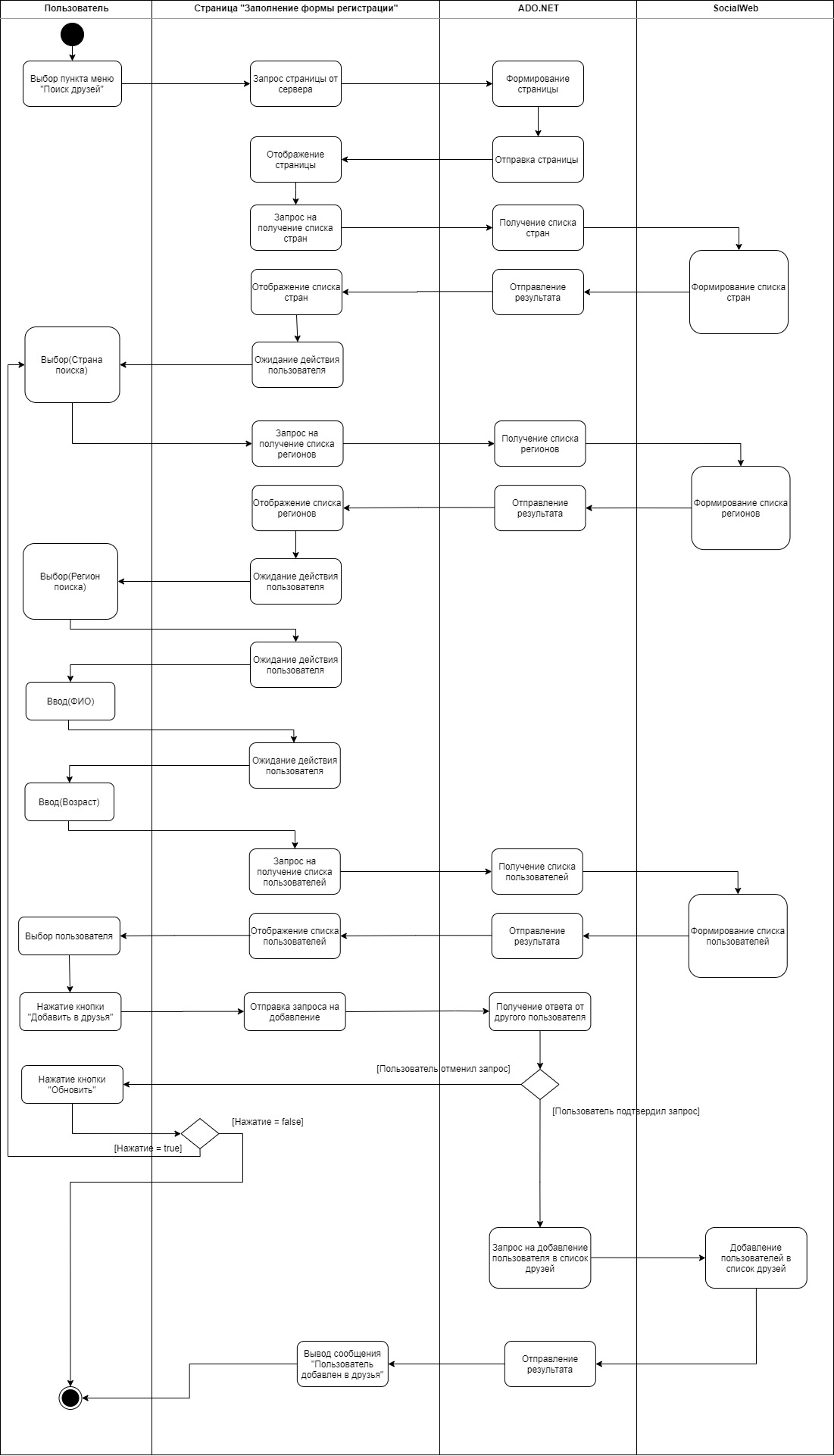
**

Рисунок 4.2.2. Диаграмма деятельности, описывающая процесс поиска друзей

# 5. Модель реализации

Модель реализации представляет физическую структуру реализации в терминах подсистем реализации и элементов реализации (каталогов и фалов, включая файлы исходного кода, файлы данных и исполняемые файлы и т.д).

Модель реализации идентифицирует физические компоненты реализации с целью облегчения их восприятия и управления ними.

Модель реализации определяет основные блоки интеграции, вокруг которых организованы рабочие группы, а также блоки, которым можно по отдельности присваивать версии, отдельно развертывать и заменять.

Модель реализации представляется диаграммами компонентов и развертывания.

Таким образом, при разработке модели преследуются цели:

* определение окончательного состава, структуры и кода классов;
* распределение классов по компонентам и подсистемам;
* определение топологии распределенной системы и распределение подсистем по узлам сети;
* планирование итераций (версий) сборки системы;
* сборка версий системы.

При разработке модели реализации рекомендуется построить диаграммы:

* компонентов;
* развертывания.

## 5.1 Диаграммы компонентов

Диаграмма компонентов описывает особенности физического представления системы.

Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный и исполняемый код.

Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними.

Диаграмма компонентов разрабатывается для следующих целей:

* визуализации общей структуры исходного кода программной системы;
* спецификации исполняемого варианта программной системы;
* обеспечения многократного использования отдельных фрагментов программного кода;
* представления концептуальной и физической схем баз данных.

В языке UML для компонентов определены следующие стереотипы:

* «file» – любой файл, кроме таблицы;
* «executable» – программа (исполняемый файл);
* «library» – статическая или динамическая библиотека;
* «source» – файл с исходным текстом программы;
* «document» – остальные файлы (например, файл справки);
* «table» – таблица базы данных.

На рисунке 5.1.1 показана диаграмма компонентов. На рисунке 5.1.2 показана диаграмма компонентов серверного приложения.

На данной диаграмме отражены основные элементы разрабатываемой информационной системы, а также основной исполняемый файл, файлы компонентов клиентского приложения, служебные файлы с дополнительной информацией, подключаемые библиотеки.

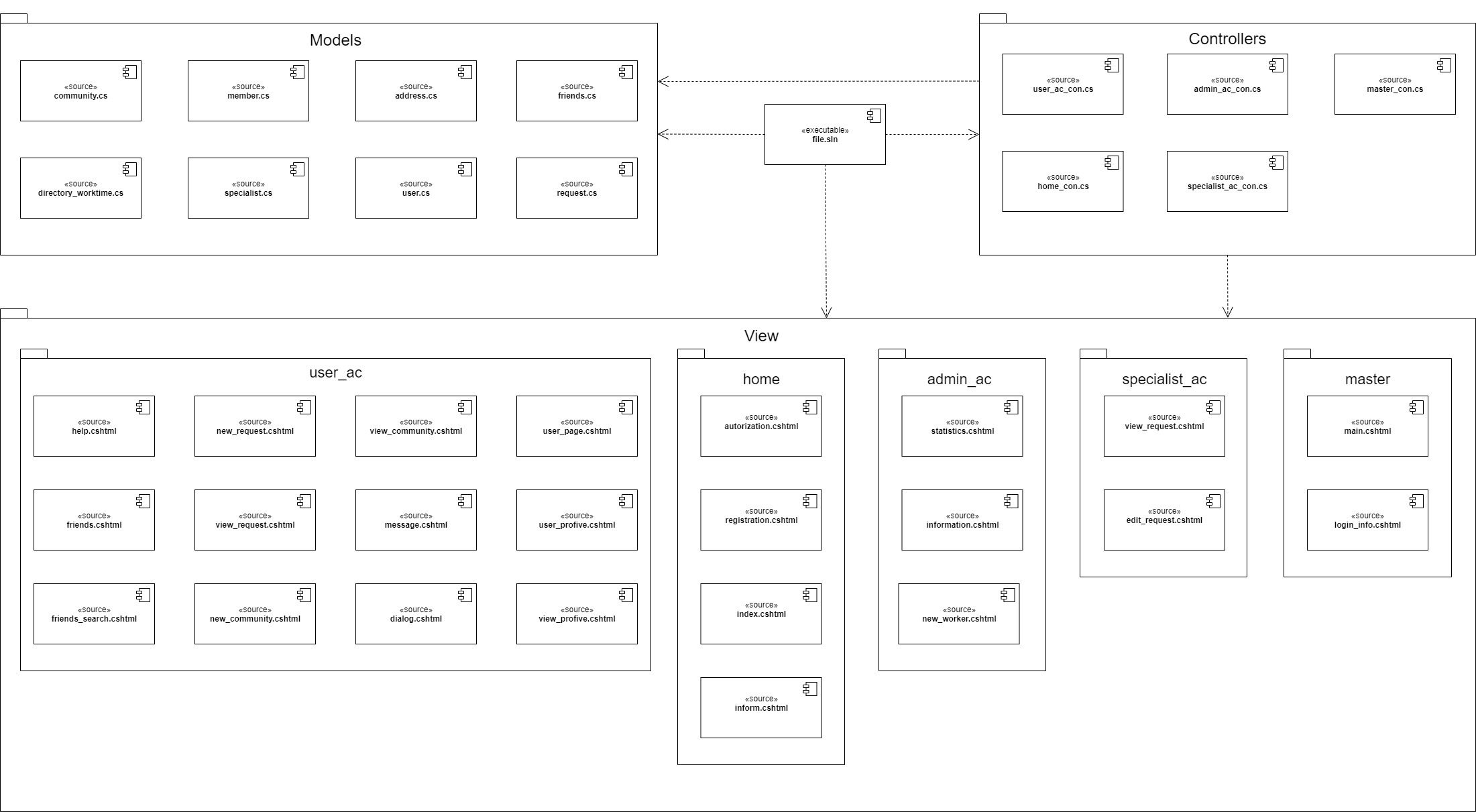


Рисунок 5.1.1. Диаграмма компонентов клиентского приложения

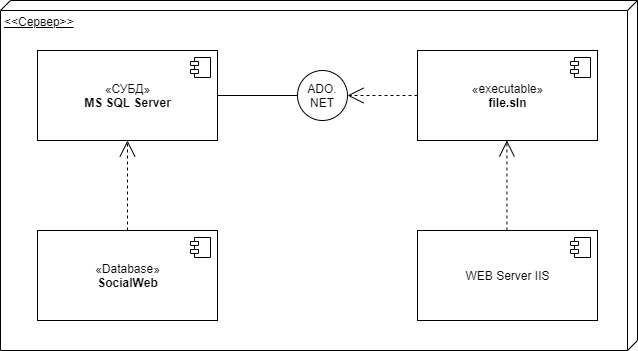


Рисунок 5.1.2. Диаграмма компонентов клиентского серверного приложения

## 5.2 Диаграмма развертывания

Целью диаграммы развертывания является представление физического расположения системы, взаимодействия физического оборудования, на котором запускается та или иная составляющая программного обеспечения.

Основные цели, преследуемые при разработке диаграммы развертывания:

* распределение компонентов системы по ее физическим узлам;
* отображение физических связей между узлами системы на этапе исполнения;
* выявление узких мест системы и реконфигурация ее топологии для достижения требуемой производительности.

Элементами диаграммы реализации являются узлы, компоненты и связи между ними.

На рисунке 5.2.1 представлена диаграмма развертывания, которая отражает физические компоненты информационной системы.

Для работы с информационной системой, на машинах пользователей должен быть установлен браузер. Само приложение и база данных устанавливаются на сервер.

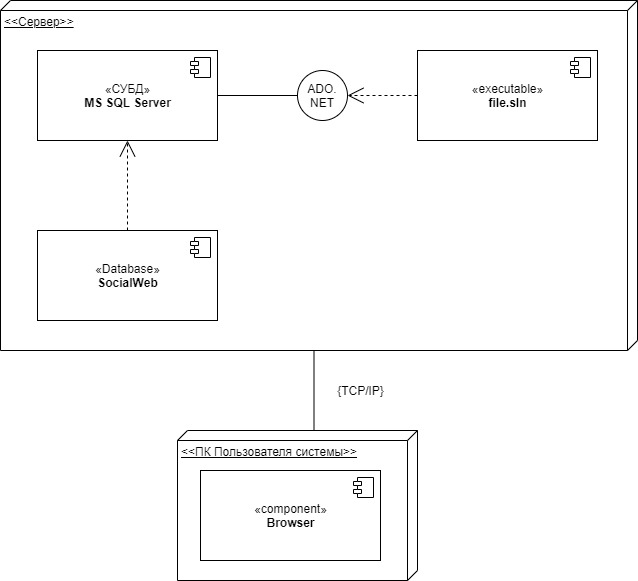


Рисунок 5.2.1 Диаграмма развертывания ИС

# 6. Сгенерированный программный код

Средствами программного продукта АSTAH-PROFESSIONAL (C#) для сформированной статической модели классов был автоматически сгенерирован программный код.

public class Address : tblBD

{

int AddressID;

Directory\_Country Country;

Directory\_City City;

public Address()

{ }

public Address(int AddressID, Directory\_Country Country, Directory\_City City) { }

}

Листинг 6.1 Класс Address (Адрес)

public class Member : tblBD

{

int MemberId;

User User;

Community Community;

string Role;

public Member()

{ }

public Member(int MemberId, User User, Community Community, string Role) { }

}

Листинг 6.2 Класс Member (Участник)

public class Community : tblBD

{

int CommunityID;

string Name;

string Theme;

string Discription;

image Photo;

public Community()

{ }

public Community(int CommunityID, string Name, string Theme, string Discription, image Photo) { }

}

Листинг 6.3 Класс Community (Сообщество)

public class User : tblBD

{

int UserID;

string Login;

string Password;

string Surname;

string Name;

string Patronymic;

string Email;

image UserPhoto;

public User()

{ }

public User(int UserID, string Login, string Password, string Surname, string Name, string Patronymic, string Email, image UserPhoto) { }

}

Листинг 6.4 Класс User (Пользователь)

public class Specialist : tblBD

{

int SpecialistID;

string Login;

string Password;

string Surname;

string Name;

string Patronymic;

string Position;

Directory\_WorkTime WorkTime;

public Specialist()

{ }

public Specialist(int SpecialistID, string Login, string Password, string Surname, string Name, string Patronymic, string Position, Directory\_WorkTime WorkTime) { }

}

Листинг 6.5 Класс Specialist (Технический специалист)

public class Friends : tblBD

{

int RecordID;

User User1;

User User2;

public Friends()

{ }

public Friends(int RecordID, User User1, User User2) { }

}

Листинг 6.6 Класс Friends (Друзья)

public class Request : tblBD

{

int RequestID;

User User;

Specialist Specialist;

DateTime CreationDate;

DateTime ClosingDate;

string Message;

public Request()

{ }

public Request(int RequestID, User User, Specialist Specialist, DateTime CreationDate, DateTime ClosingDate, string Message) { }

}

Листинг 6.7 Класс Request (Обращение)

public class Directory\_WorkTime : tblBD

{

int WorkTimeID;

string WorkTime;

public Directory\_WorkTime()

{ }

public Directory\_WorkTime(int WorkTimeID, string WorkTime) { }

}

Листинг 6.8 Класс Directory\_WorkTime (Справочник смен)

public class Directory\_City : tblBD

{

int CityID;

string City;

public Directory\_City()

{ }

public Directory\_City(int CityID, string City) { }

}

Листинг 6.9 Класс Directory\_City (Справочник городов)

public class Directory\_Country : tblBD

{

int CountryID;

string Country;

public Directory\_Country()

{ }

public Directory\_Country(int CountryID, string Country) { }

}

Листинг 6.10 Класс Directory\_Country (Справочник стран)

# 7. Руководство пользователя

## 7.1 Авторизация

После открытия сайта в браузере, сперва пользователь попадает на страницу Авторизации (рис 7.1.1), если пользователь не был авторизован до этого в системе. Пользователю необходимо указать логин и пароль.

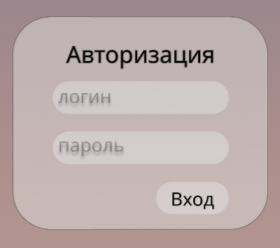


Рисунок 7.1.1. Форма авторизации

После ввода пароля, пользователю необходимо нажать кнопку «Вход». В случае если логин и пароль верный, пользователь увидит главную страницу (рис. 7.3.1), иначе пользователя уведомят о неправильности введенных данных.

## 7.2 Главная страница

Функционал программы зависит от того, кто был авторизирован. Так, например, если процесс авторизации прошел менеджер, он увидит главную страницу в следующем формате (рис 7.2.1). Тут мы видим функционал, который доступен менеджеру.

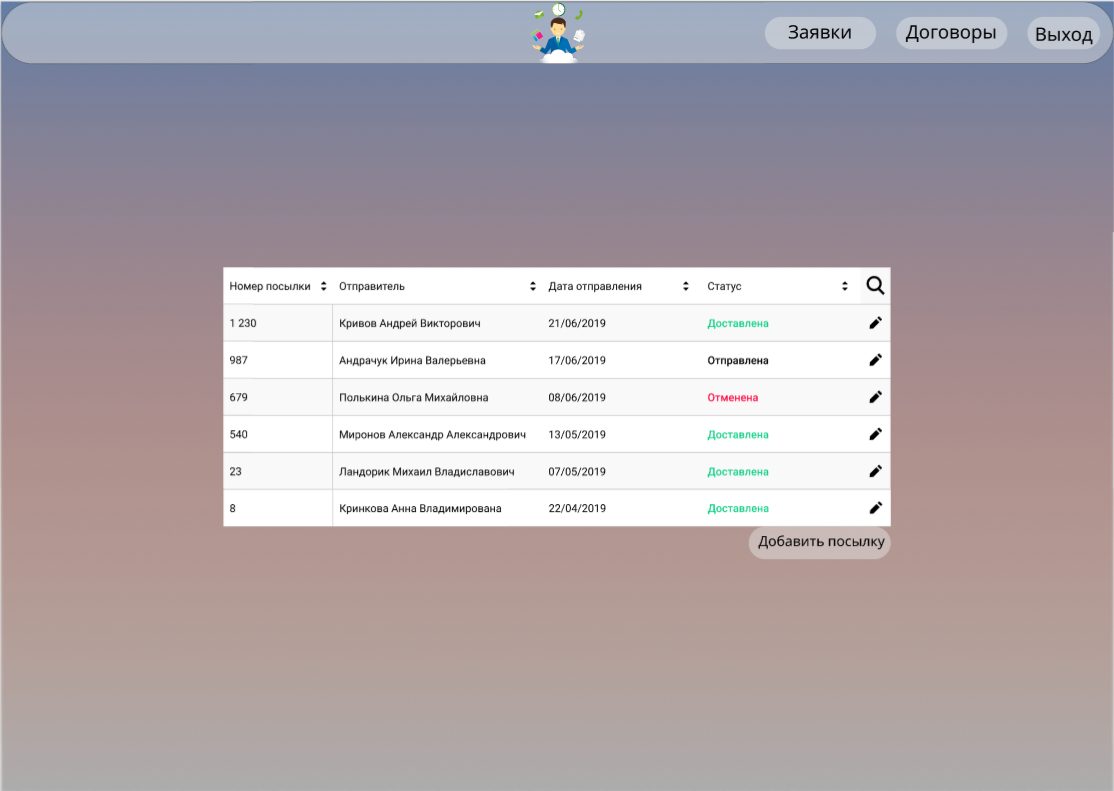


Рисунок 7.2.1 Главная страница менеджера

Действия менеджера могут быть различными:

1. Редактирование посылки, нажав кнопку карандаша на нужной посылке в таблице;
2. Сортировка таблицы, нажав на необходимый столбец таблицы;
3. Поиск записи в таблице, нажав на кнопку поиска;
4. Переход на страницу «Заявки на доставку», нажав на кнопку «Заявки»;
5. Переход на страницу «Договоры на доставку», нажав на кнопку «Договоры»;
6. Добавление новой посылки, нажав на кнопку «Добавить посылку»;
7. Выйти из системы, нажав кнопку «Выход».

На рисунке 7.2.2 изображена главная страница, для пользователя с ролью курьер. Из него видно, что пользователь при авторизации автоматически попадает на страницу просмотра маршрутных листов.

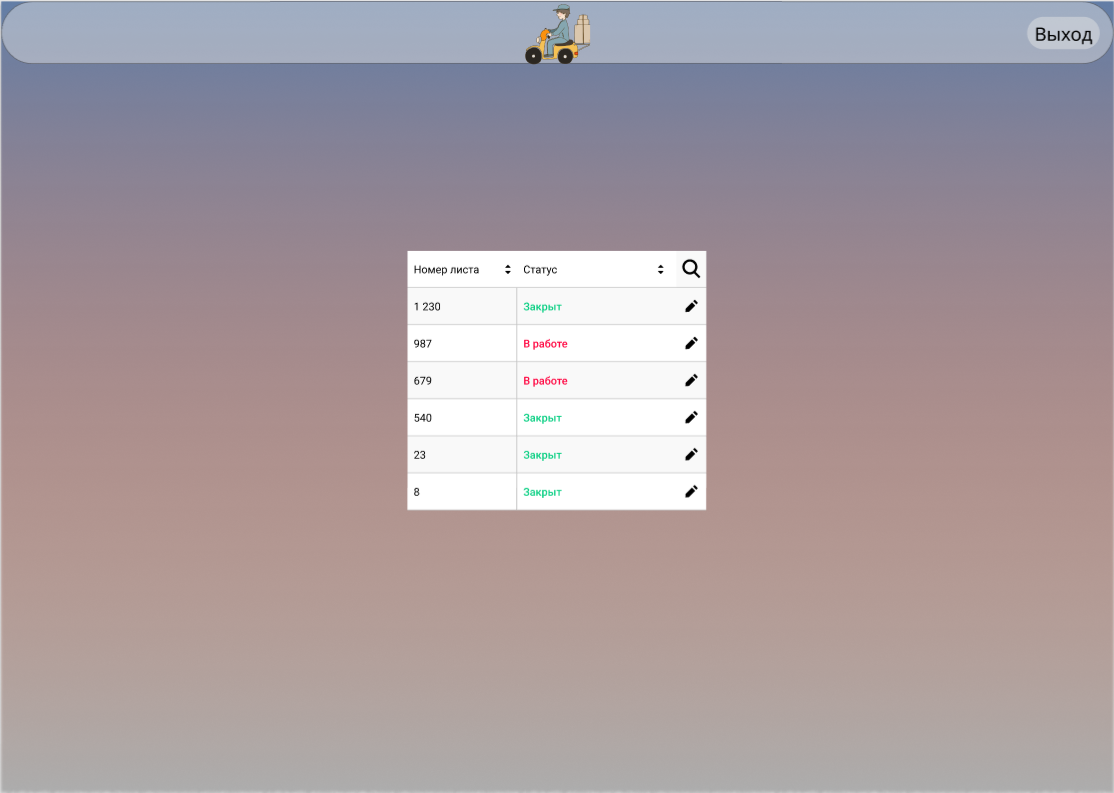


Рисунок 7.2.2. Главная страница курьера

Действия курьера могут быть следующими:

1. Просмотр конкретного маршрутного листа, нажав кнопку карандаша на нужном маршрутном листе в таблице;
2. Сортировка таблицы, нажав на необходимый столбец таблицы;
3. Поиск записи в таблице, нажав на кнопку поиска;
4. Выйти из системы, нажав кнопку «Выход».

На рисунке 7.2.3 изображена главная страница, для пользователя с ролью логист. Из него видно, что пользователь при авторизации автоматически попадает на страницу просмотра заявок на доставку.

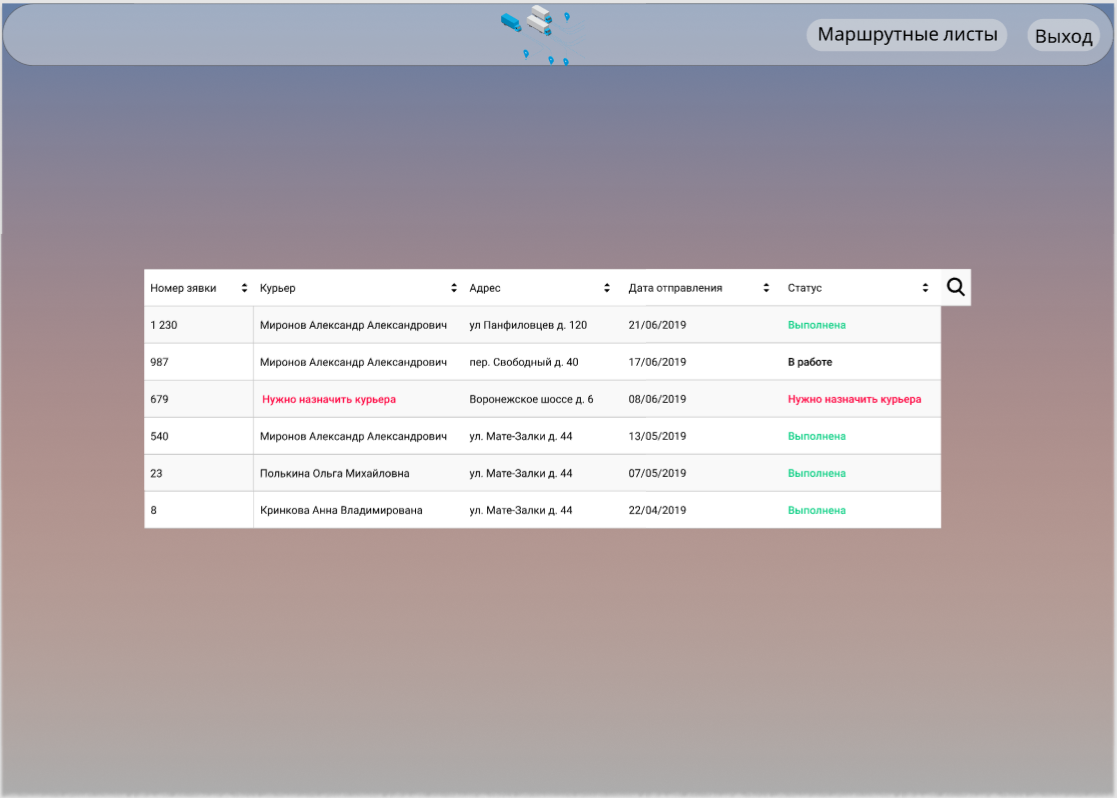


Рисунок 7.2.3. Главная страница логиста

Действия курьера могут быть следующими:

1. Просмотр конкретной заявки на доставку, нажав кнопку карандаша на нужной заявке в таблице;
2. Сортировка таблицы, нажав на необходимый столбец таблицы;
3. Поиск записи в таблице, нажав на кнопку поиска;
4. Переход на страницу «Маршрутные листы», нажав на кнопку «Маршрутные листы»;
5. Выйти из системы, нажав кнопку «Выход».

На рисунке 7.2.4 изображена главная страница, для пользователя с ролью администратор. Из него видно, что пользователь при авторизации автоматически попадает на страницу просмотра списка пользователей.

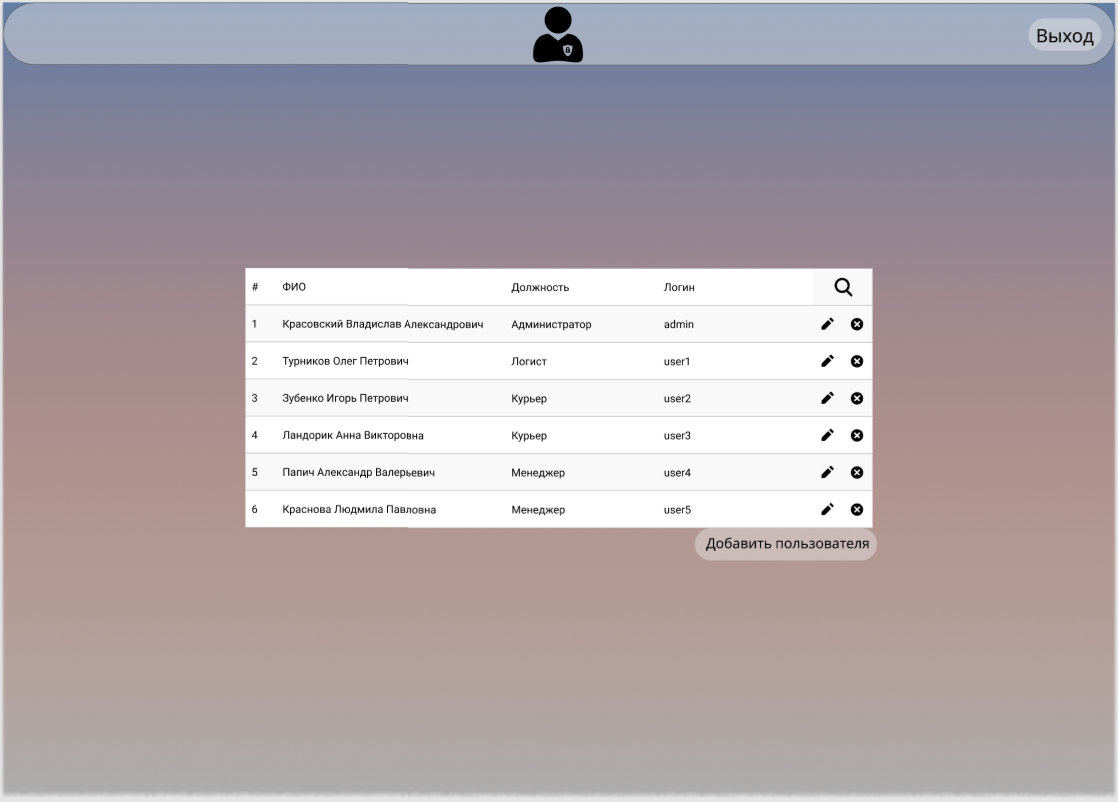


Рисунок 7.2.4. Главная страница администратора

Действия администратора могут быть следующими:

1. Просмотр конкретной учётной записи, нажав кнопку карандаша на нужной учётной записи в таблице;
2. Сортировка таблицы, нажав на необходимый столбец таблицы;
3. Поиск записи в таблице, нажав на кнопку поиска;
4. Добавление новой учётной записи, нажав на кнопку «Добавить пользователя»;
5. Выйти из системы, нажав кнопку «Выход».

## 7.3 Процесс работы менеджера

Процесс работы менеджера сводится к заполнению форм на страницах: «Создание посылки» (Рисунок 7.3.1), «Создание договора» (Рисунок 7.3.2), «Создание заявки на доставку» (Рисунок 7.3.3)

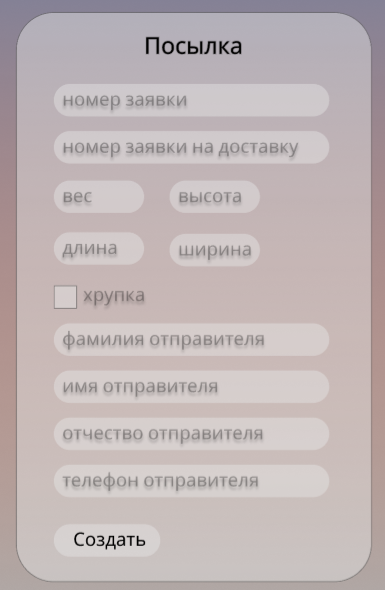


Рисунок 7.3.1. Форма создания посылки



Рисунок 7.3.2. Форма создания договора



Рисунок 7.3.3. Форма создания заявки на доставку

Во всех случаях менеджеру необходимо заполнить все поля на форме и нажать кнопку «Создать». При успешном создании, появиться соответствующее уведомление.

Если менеджеру необходимо изменить существующую запись, необходимо выбрать её в общей таблице, как описано в пункте 7.2 руководства пользователя (Рисунок 7.2.1). Далее откроется одна из выше перечисленных форм с уже заполненными значениями.

## 7.4 Процесс работы логиста

Процесс работы логиста сводится к заполнению формы на странице «Создание маршрутного листа» (Рисунок 7.4.1) и просмотру списка заявок на главной странице (эта страница описана в разделе 7.2 руководства пользователя (Рисунок 7.2.3))

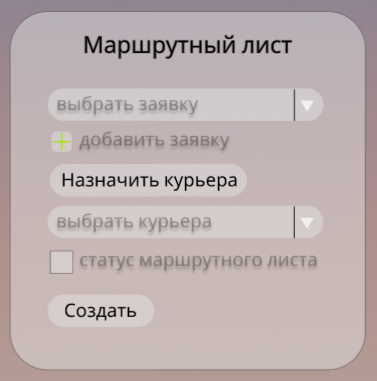


Рисунок 7.4.1. Форма создания маршрутного листа

Логисту необходимо заполнить все поля на форме и нажать кнопку «Создать». При успешном создании, появиться соответствующее уведомление.

Если необходимо изменить существующую запись, необходимо выбрать её в общей таблице, как описано в пункте 7.2 руководства пользователя (Рисунок 7.2.3). Далее откроется форма, упомянутая выше, с уже заполненными значениями.

## 7.5 Процесс работы курьера

Процесс работы курьера сводится к заполнению формы на странице «Создание отчёта» (Рисунок 7.5.1) и просмотру списка маршрутных листов на главной странице (эта страница описана в разделе 7.2 руководства пользователя (Рисунок 7.2.2))

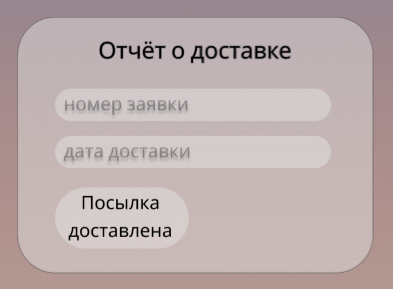


Рисунок 7.5.1. Форма создания отчёта о доставке

Курьеру необходимо заполнить все поля на форме и нажать кнопку «Посылка доставлена». При успешном создании, появиться соответствующее уведомление.

## 7.6 Процесс работы администратора

Процесс работы администратора сводится к заполнению формы на странице «Создание пользователя» (Рисунок 7.6.1) и просмотру списка пользователей на главной странице (эта страница описана в разделе 7.2 руководства пользователя (Рисунок 7.2.4))

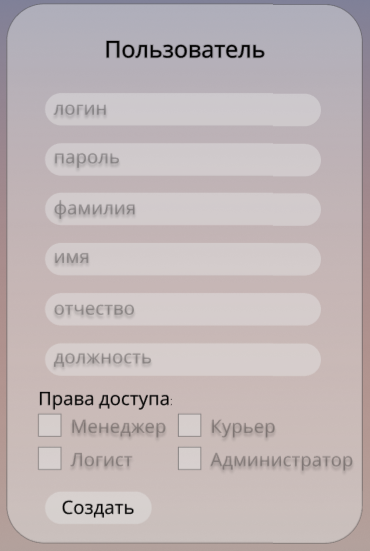


Рисунок 7.6.1. Форма создания пользователя

Администратору необходимо заполнить все поля на форме и нажать кнопку «Создать». При успешном создании, появиться соответствующее уведомление.

Если необходимо изменить существующую запись, необходимо выбрать её в общей таблице, как описано в пункте 7.2 руководства пользователя (Рисунок 7.2.4). Далее откроется форма, упомянутая выше, с уже заполненными значениями.

# Заключение

Целью данного курсового проект являлась разработка проекта информационной системы «Курьерская служба».

В ходе курсового проекта было проведено исследование предметной области, беседы с заказчиком и в полной мере раскрыта тема поставленной цели.

Разработаны и построены различные модели для проекта информационной системы. С учетом построенных моделей была разработана информационная система. Для разработанной информационной системы было написано руководство пользователя.

Разработанное приложение построено с использованием актуальных технологий веб-разработки на сегодняшний день.

# Список используемых источников

1. Анисимов, В. В. Проектирование информационных систем: курс лекций [Текст] : в 2 ч. / В.В. Анисимов. – Хабаровск: Изд-во ДВГУПС, 2006. – Ч. 1; Структурный подход. – 2006. – 112 с.

2. Анисимов, В. В. Проектирование информационных систем: курс лекций [Текст] : в 2 ч. / В. В. Анисимов. – Хабаровск: Изд-во ДВГУПС, 2007. – Ч. 2; Обектно-ориентированный подход. – 2007. – 100 с.

3. Гурвиц, Г. А. Microsoft Access 2010. Разработка приложений на реаль-ном примере [Текст] / Г. А. Гурвиц. Санкт-Петербург : БХВ-Петербург, 2010. - 493 с.