СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc151842406)

[1 ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ 5](#_Toc151842407)

[1.1 Основные требования по технике безопасности во время прохождения учебной практики 6](#_Toc151842408)

[1.2 Нормативно-правовые документы 7](#_Toc151842409)

[2 ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ 9](#_Toc151842410)

[2.1 Описание предметной области 9](#_Toc151842411)

[2.2 Группы пользователей информационной системы 9](#_Toc151842412)

[2.3 Основные требования, предъявляемые к информационной системе 10](#_Toc151842413)

[3 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 12](#_Toc151842414)

[3.1 Моделирование некоторых динамических аспектов системы 12](#_Toc151842415)

[3.1.1 Создание диаграммы вариантов использования 13](#_Toc151842416)

[3.1.2 Создание диаграммы деятельности 14](#_Toc151842417)

[3.2 Разработка функциональной структуры информационной системы. Карта навигации по информационной системе 18](#_Toc151842418)

[3.3 Разработка макета информационной системы 19](#_Toc151842419)

[4 РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ 22](#_Toc151842420)

[4.1 Моделирование и разработка базы данных 22](#_Toc151842421)

[4.2 Разработка программных модулей информационной системы 26](#_Toc151842422)

[5 РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ 46](#_Toc151842423)

[5.1 Назначение эксперимента. Выбор и обоснование методики проведения тестирования 46](#_Toc151842424)

[5.2 Технология тестирования 47](#_Toc151842425)

[5.3 Результаты проведения тестирования 47](#_Toc151842426)

[ЗАКЛЮЧЕНИЕ 48](#_Toc151842427)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 49](#_Toc151842428)

[ПРИЛОЖЕНИЕ A 50](#_Toc151842429)

# ВВЕДЕНИЕ

Развитие сельского хозяйства и садоводства неразрывно связано с качеством посевного материала. Инновационный институт, сфокусированный на сборе, разведении и распространении разнообразных семян, играет ключевую роль в этом процессе. Его обширный ассортимент охватывает не только базовые сорта, но и редкие и экзотические растения, создавая возможность для разносторонних аграрных и садоводческих проектов.

Важным этапом в работе института является тщательное тестирование только что выведенных сортов. Это обеспечивает не только высокий уровень качества посевного материала, но и формирует базу для последующего анализа и улучшения характеристик растений. Каждый семенной сорт имеет свои уникальные особенности, включая урожайность, степень морозоустойчивости, адаптацию к окружающим условиям и время созревания, что позволяет удовлетворить разнообразные потребности клиентов.

Компания активно работает как с оптовыми, так и розничными покупателями, стремясь обеспечить максимальную доступность своего ассортимента. Оптовые партнеры, занесенные в специализированную базу данных, получают информацию о новых поступлениях или изменениях в наличии сортов, что обеспечивает оперативность их реагирования на рыночные изменения и потребности клиентов.

Этот контекст создает необходимость в оптимизации процессов управления ассортиментом, углубленном изучении характеристик семян и повышении уровня обслуживания клиентов. В данном задании будет рассмотрена детальная стратегия по улучшению информационной базы, оптимизации системы поиска и выбора семян, а также укреплению взаимодействия с оптовыми партнерами с целью укрепления позиций института на рынке качественных семян.

В рамках модуля ПМ 01 Разработка модулей программного обеспечения для компьютерных систем, будет разработана и спроектирована информационная система “Продажа сортов семян”, которая будет включать в себя несколько видов профессиональных компетенций:

1. выполнять разработку спецификаций отдельных компонент;
2. осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля;
3. выполнять отладку программных модулей с использованием специализированных программных средств;
4. выполнять тестирование программных модулей;
5. осуществлять оптимизацию программного модуля;
6. разрабатывать компоненты проектной и технической документации с использованием графических языков спецификаций.

# ЦЕЛЬ И ЗАДАЧИ УЧЕБНОЙ ПРАКТИКИ

Разработка прикладного программного обеспечения института селекции растений. Целью является создание информационной системы, способствующей оптимизации процессов управления семенным ассортиментом института, а также в улучшении информационной базы для оптовых покупателей.

Задачи:

1. Закрепление и углубление знаний и умений.
2. Формирование необходимых навыков профессиональной компетенции.
3. Развить готовность выполнять поставленные профессиональные задачи.
4. Разработка программных модулей¸ оптимизированных для работы.
5. Тестирование разработанных программных модулей на соответствие функциональным требованиям.

Цель и задачи в совокупности направлены на повышение эффективности обслуживания и обеспечение безопасности и надежности пассажирского транспорта через современные информационные технологии.

Для достижения цели и задач будет разработана и спроектирована информационная система “Продажа сортов растений” нацеленная на оптимизацию процесса внесения и хранения данных в этой предметной области, в рамках учебной практики по модулю ПМ 01 (Разработка модулей программного обеспечения для компьютерных систем), овладевая профессиональными компетенциями и задачами:

* систематизация, обобщение, закрепление и углубление знаний и умений;
* формирование у студента знаний, умений и навыков, профессиональных компетенций, профессионально значимых личностных качеств;
* развитие профессионального интереса, формирование мотивационно-целостного отношения к профессиональной деятельности, готовности к выполнению профессиональных задач в соответствии с нормами морали, профессиональной этики и служебного этикета.

## Основные требования по технике безопасности во время прохождения учебной практики

К работе в компьютерном классе допускаются лица, ознакомленные с данной инструкцией по технике безопасности и правилам поведения. Необходимо неукоснительно соблюдать правила по технике безопасности, т.к. нарушение этих правил может привести к поражению электрическим током, вызвать возгорание и навредить вашему здоровью.

- Работа, обучающегося в компьютерном классе разрешается только в присутствии преподавателя (лаборанта).

- Во время занятий посторонние лица могут находиться в классе только с разрешения преподавателя.

- Во время перемен между уроками проводится обязательное проветривание компьютерного кабинета с обязательным выходом обучающихся из класса.

Каждый обучающийся в ответе за состояние своего рабочего места и сохранность размещенного на нем оборудования.

Обучающийся допустивший нарушение по технике безопасности и правилам поведения в компьютерном классе, может быть привлечен к дисциплинарной ответственности. Если нарушение техники безопасности связано с причинением имущественного ущерба, обучающийся несет и материальную ответственность в установленном законом порядке.

## Нормативно-правовые документы

Оформление и содержание технического задания соответствует требованиям стандарта «ГОСТ 19.201-78. ЕСПД. Общие положения.

Техническое задание оформлено в соответствии с ГОСТ 19.106-78 на листах формата 11 и 12 по ГОСТ 2.301-68, как правило, без заполнения полей листа. Номера листов (страниц) проставлены в верхней части листа над текстом.

Лист утверждения и титульный лист оформлен в соответствии с ГОСТ 19.104-78.

Информационную часть (аннотацию и содержание), лист регистрации изменений допускается в документ не включать.

Для внесения изменений или дополнений в техническое задание на последующих стадиях разработки программы или программного изделия выпускают дополнение к нему.

Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

Техническое задание должно содержать следующие разделы:

* введение;
* основания для разработки;
* назначение разработки;
* требования к программе или программному изделию;
* требования к программной документации;
* технико-экономические показатели;
* стадии и этапы разработки;
* порядок контроля и приемки;
* в техническое задание допускается включать приложения.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

Вывод по главе: были описаны, техника безопасности во время прохождения учебной практики; нормативно-правовые документы, которые отвечают за стандарты разработки; цели и задачи учебной практики, включая вводную часть, где описана тематика отрасли и её тенденции развития.

# ПРЕДМЕТНАЯ ОБЛАСТЬ. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ

Предметная область информационной системы — это материальная система или система, характеризующая элементы материального мира, информация о которой хранится и обрабатывается. Предметная область рассматривается как некоторая совокупность реальных объектов и связей между ними.

## Описание предметной области

Необходимо разработать и спроектировать прикладное программное обеспечение для эффективного управления продажами сортов семян.

Данный институт занимается сбором, выведением и продажей различных сортов семян. В его ассортименте можно найти семена практически всех возможных видов растений: от помидоров до редких цветов. Только что выведенные сорта заносятся в отдельный список для дальнейшего тестирования. Каждый сорт семян имеет свои характеристики, такие как морозоустойчивость, адаптация к местным условиям, сроки созревания (раннеспелый, среднеспелый, поздний) и т.п. Покупатель может выбрать сорт, отвечающий тем или иным характеристикам. Компания занимается как оптовыми, так и розничными продажами. Оптовые покупатели заносятся в базу главным образом для того, чтобы информировать их о поступлении новых или отсутствовавших в определенный момент в продаже сортов.

## Группы пользователей информационной системы

Администратор - имеет полный доступ ко всей информационной системе. Может авторизоваться, просмотреть или изменить данные во всех таблицах. Также администратор может добавлять, удалять, изменять и просматривать учётные записи пользователей.

Продавец - имеет ограниченные возможности в использовании информационной системы. Может авторизоваться, а также просматривать и изменять записи в таблицах Продажи, Покупатели и Оптовые\_покупатели, а также только просматривать таблицы Продавцы, Сорта\_растений, Новые\_сорта\_растений, Упаковки\_семян.

Покупатель - ограничен в использованиях информационной системы, может только авторизоваться и просматривать данные из таблиц Сорта\_растений, Упаковки\_семян.

Гость - ограничен в использованиях информационной системы, может только просматривать данные из таблиц Сорта\_растений, Упаковки\_семян.

## Основные требования, предъявляемые к информационной системе

Гибкость:

* Система должна обладать высокой степенью конфигурируемости, чтобы пользователи могли адаптировать её под различные потребности селекционного института;
* Возможность легкого масштабирования системы при увеличении объема данных или числа пользователей.

Надёжность:

* Обеспечение бесперебойной работы системы с минимальным количеством сбоев и перерывов;
* Регулярное создание резервных копий данных для предотвращения потери информации в случае сбоев.

Эффективность:

* Обеспечение высокой производительности при обработке данных, особенно в условиях одновременного использования множества пользователей
* Обеспечение высокой производительности при обработке данных, особенно в условиях одновременного использования множества пользователей

Безопасность:

* Реализация надежной системы аутентификации для проверки личности пользователей и установление строгих прав доступа.
* Использование шифрования для защиты передаваемой и хранимой чувствительной информации.
* Ведение журнала аудита для отслеживания действий пользователей и обнаружения возможных угроз безопасности.

Вывод по главе: было описано, что такое предметная область и её описание; группы пользователей, которые будут пользоваться информационной системой «Продажа сортов семян» и основные требования к ней.

# ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Проектированием информационных систем называется многоступенчатый процесс их создания и/или модернизации путём применения упорядоченной совокупности методологий и инструментария. Проектирование (в отличие от моделирования) предполагает работу с пока несуществующим объектом и направлено на создание информационной системы в области:

* обработки объектов будущей базы данных;
* написания программ (в том числе – отчётных и экранных форм), обеспечивающих выполнение запросов к данным,
* выполнения учёта функционирования конкретной среды (технологии).

Если выделять стадию проектирования информационных систем в качестве отдельного этапа, то его можно разместить между этапами анализа и разработки. Однако на практике чёткое разделение на этапы, как правило, затруднено или невозможно, поскольку проектирование, формально начинаясь с определения цели проекта, часто продолжается на стадиях тестирования и реализации.

## Моделирование некоторых динамических аспектов системы

Диаграммы деятельности применяются для моделирования динамических аспектов поведения системы. Эти динамические аспекты могут включать деятельность на любом уровне абстракции в любом представлении системной архитектуры, включая классы (в том числе активные), интерфейсы, компоненты и узлы.

Использовать диаграммы деятельности для моделирования некоторых динамических аспектов системы вы можете в контексте почти любого моделируемого элемента. Однако чаще вы будете прибегать к таким диаграммам в контексте всей системы, подсистемы, операции или класса. Диаграмму деятельности можно присоединить к варианту использования, чтобы моделировать сценарий, и к кооперации, чтобы моделировать динамические аспекты поведения совокупности объектов.

При разработке программного обеспечения заблаговременное планирование и моделирование значительно упрощают программирование. На этапе создания концептуальной модели для описания функционального назначения используется диаграмма вариантов использования, а для описания логики процедур и бизнес-процессов используется диаграмма деятельности.

### Создание диаграммы вариантов использования

В приложении 1, на рисунке 1 представлен пример диаграммы вариантов использования информационной системы для продажи сортов растений.



Рисунок 1 – Диаграмма вариантов использования

### Создание диаграммы деятельности

На рисунках 2 – 5 представлена диаграмма деятельности (по ролям) информационной системы «Продажа сортов растений».



Рисунок 2 – Диаграмма деятельности (Администратор)

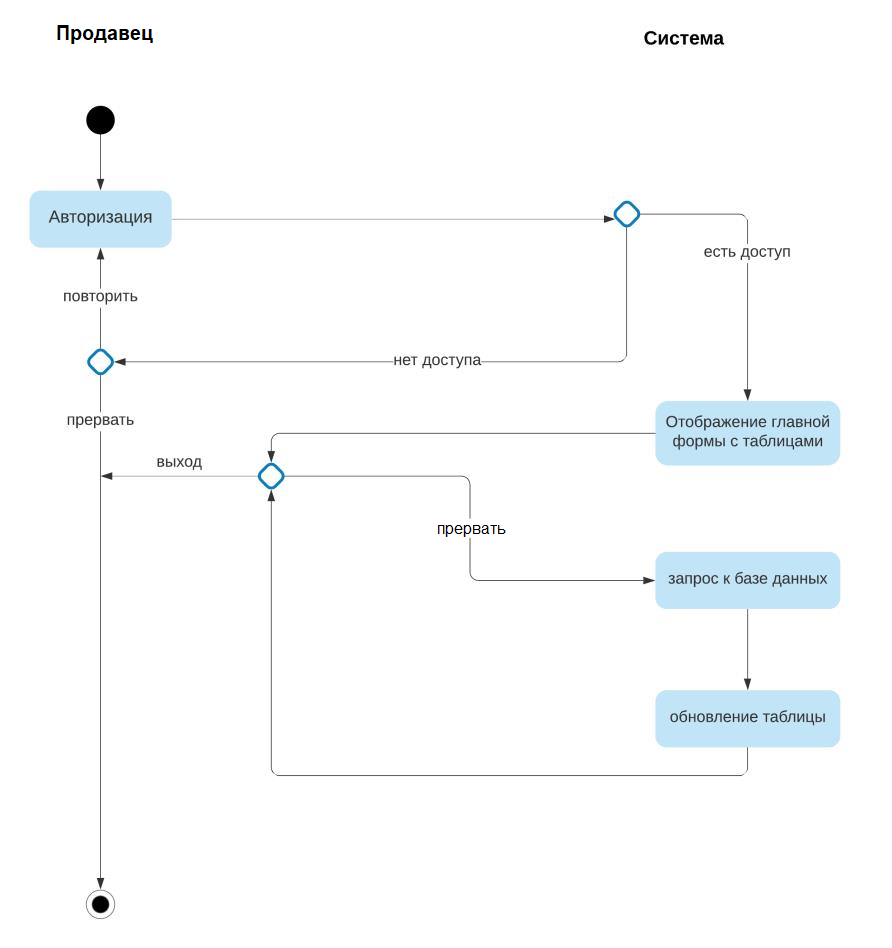


Рисунок 3 – Диаграмма деятельности (Продавец)

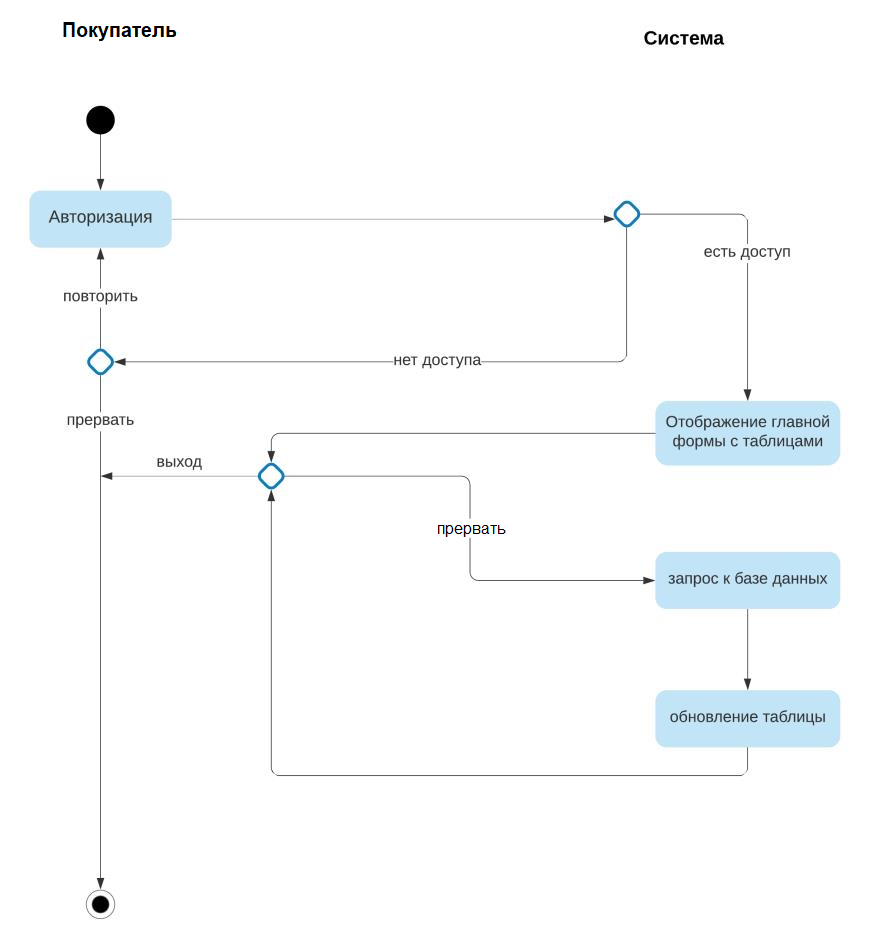


Рисунок 4 – Диаграмма деятельности (Покупатель)



Рисунок 5 – Диаграмма деятельности (Гость)

## Разработка функциональной структуры информационной системы. Карта навигации по информационной системе

В приложении 2 на рисунке 6 представлена карта переходов по информационной системе «Продажа сортов растений»

Рисунок 6 – Карта переходов

## Разработка макета информационной системы

В приложении 3 на рисунке 7 представлено окно авторизации.

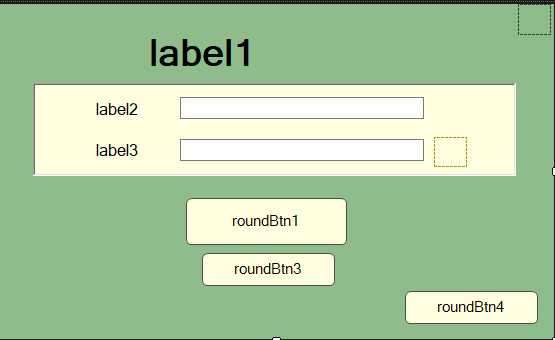


Рисунок 7 – Окно авторизации

На этапе авторизации пользователь может ввести логин и пароль, чтобы продолжить работу в информационной системе.

В случае если пользователь еще не зарегистрирован в системе, он может сделать это, нажав на кнопку «Регистрация». В этом случае откроется окно регистрации, представленное в приложении 8 на рисунке 8.

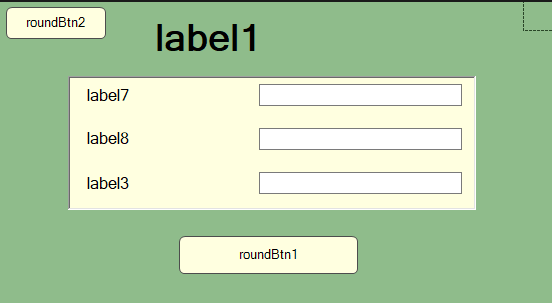


Рисунок 8 – Окно регистрации

На этапе регистрации пользователю необходимо придумать свой логин и пароль, а также ввести персональные данные, такие как почта.

Если сотрудник с введенными данными не зарегистрирован в базе

данных, в качестве сотрудника, то пользователь не сможет зарегистрироваться, и, соответственно, продолжить работу, так как данная информационная система предназначена только для сотрудников образовательной организации.

После того, как пользователь успешно прошел авторизацию, он может продолжить работу с таблицами на главной форме. На рисунке 8 - 11 представлена главная форма, которая будет отображена, в зависимости от прав пользователя в системе.

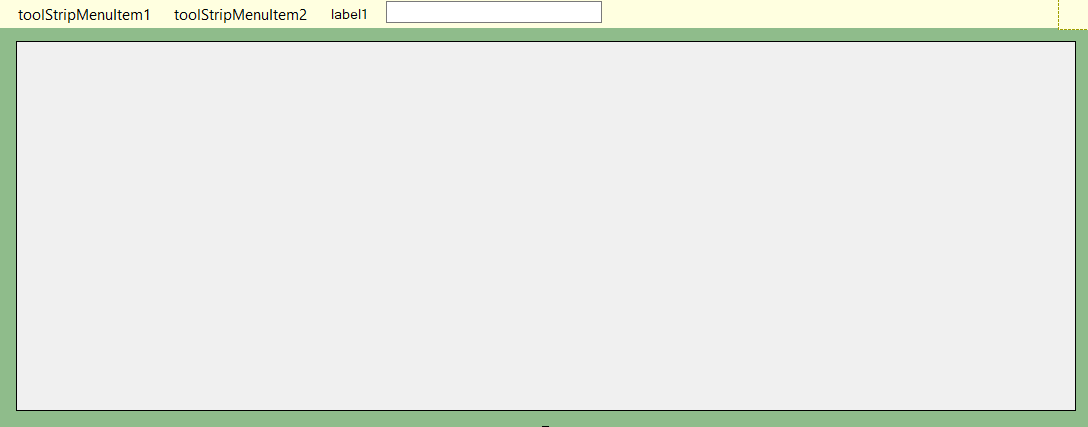


Рисунок 8 – Форма «Администратора»

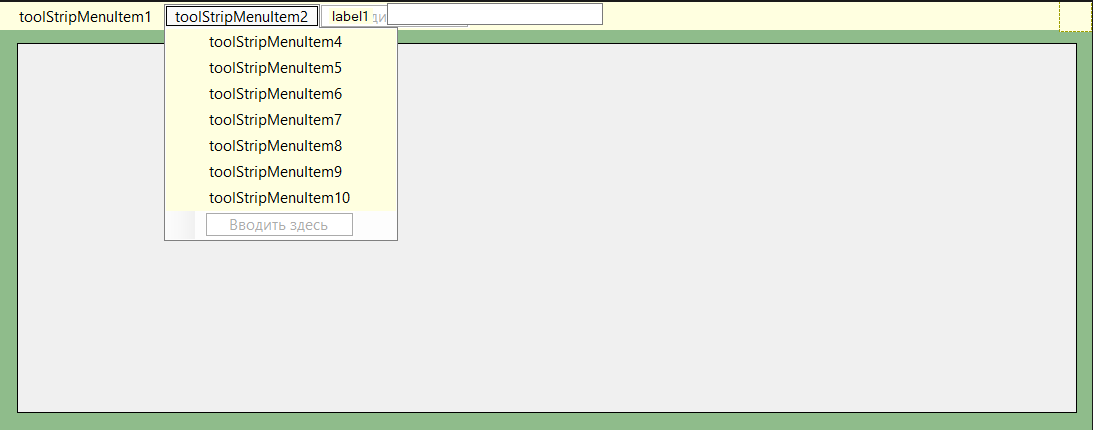


Рисунок 9 – Форма «Продавца»

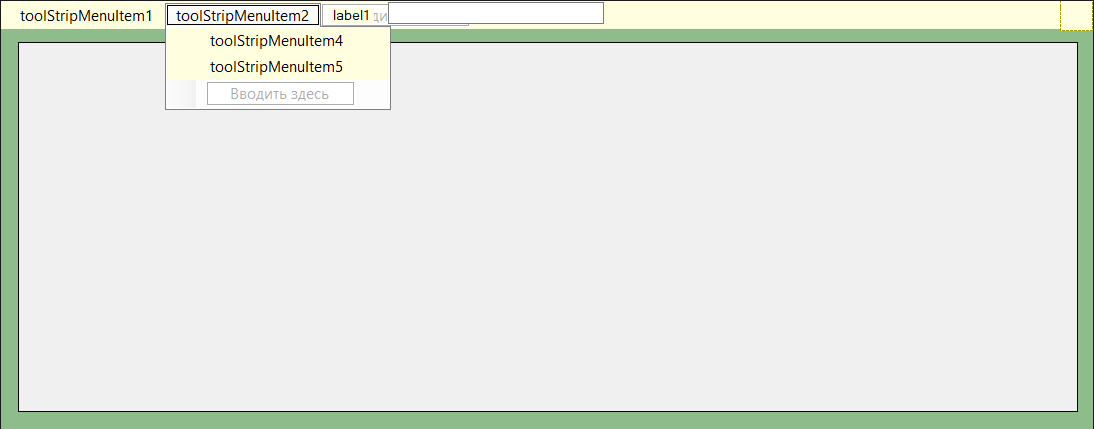


Рисунок 10 – Форма «Покупателя»

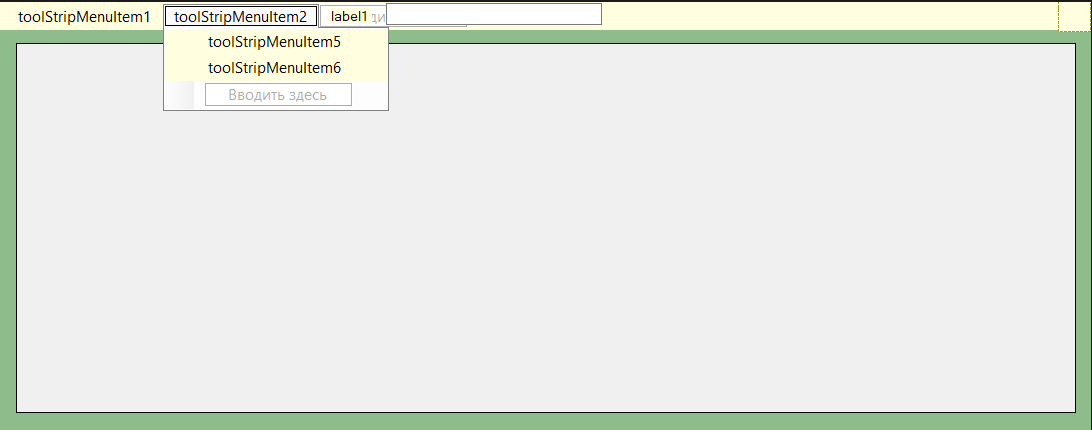


Рисунок 11 – Форма «Гостя»

Аналогичным образом в зависимости от роли пользователя выглядит главная форма, на которой расположены все необходимые таблицы. Например, продавец сможет изменять только определённые таблицы, а для гостя будут скрыты кнопки редактирования, добавления и удаления записи, так как данной группе пользователей доступен только просмотр данных.

Вывод по главе: была спроектирована диаграмма вариантов использования, диаграмма деятельности и макеты будущей информационной системы.

# РАЗРАБОТКА ПРОТОТИПА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Прототипирование — это один из этапов разработки, который заключается в продумывании содержания и расположения важных элементов интерфейса. Прототип — это моделирование конечного продукта. Это интерактивный макет, который может иметь любую степень точности. Основная цель создания прототипов – проверить, насколько последователен путь пользователя, и выявить препятствия, которые могут возникнуть в процессе его взаимодействия с продуктом.

Прототипы не только позволяют проверить удобство разрабатываемого продукта до начала написания кода, они также приводят к неожиданным открытиям и новым идеям, которые могут вывести проектируемый продукт на новый уровень.

## Моделирование и разработка базы данных

Исходя из анализа предметной области, можно выделить восемь сущностей: «Сорта растений», «Новые сорта растений», «Упаковки семян», «Покупатели», «Оптовые покупатели», «Продавцы», «Продажи», «Пользователи».

В приложении 4 на рисунке 12 представлена логическая модель данных информационной системы «Продажа сортов растений».

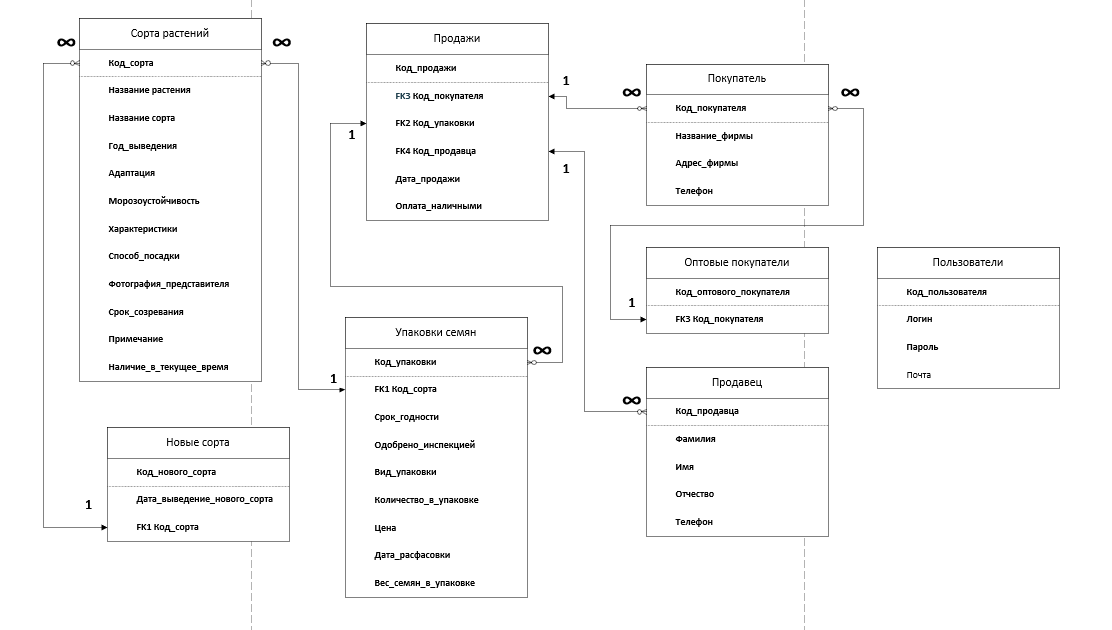


Рисунок 12 – Логическая модель данных

Создание базы данных происходит в MS SQL.

С помощью раздела «Создание» и конструктора таблиц MS Access созданы все таблицы и поля, представленные на логической модели данных.

Названия полей и их типы данных представлены в таблицах 1-8.

Таблица 1 – Поля таблицы «Сорта растений»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_сорта\_растения | Числовой |
| Название\_растения | Текстовый |
| Название\_сорта | Текстовый |
| Год\_выведения | Дата |
| Адаптация | Логический |
| Морозоустойчивость | Логический |
| Характеристики | Текстовый |
| Способ\_посадки | Текстовый |
| Фотография\_представителя | OLE |
| Срок\_созревания | Текстовый |
| Примечание | Текстовый |
| Наличие\_в\_текущее\_время | Логический |

Таблица 2 – Поля таблицы «Новые сорта растений»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_нового\_сорта\_растения | Числовой |
| Дата\_выведения\_нового\_сорта | Дата |
| Код\_сорта\_растения | Числовой |

Таблица 3 – Поля таблицы «Упаковки семян»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_упаковки\_семян | Числовой |
| Срок\_годности | Дата |
| Одобрено\_инспекцией | Логический |
| Вид\_упаковки | Текстовый |
| Количество\_в\_упаковке | Числовой |
| Цена | Денежный |
| Дата\_расфасовки | Дата |
| Вес\_семян\_в\_упаковке | Числовой |
| Код\_сорта\_растения | Числовой |

Таблица 4 – Поля таблицы «Покупатели»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_покупателя | Числовой |
| Название\_фирмы | Дата |
| Адрес\_фирмы | Текстовый |
| Телефон | Текстовый |

Таблица 5 – Поля таблицы «Оптовые покупатели»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_оптового\_покупателя | Числовой |
| Код\_покупателя | Числовой |

Таблица 6 – Поля таблицы «Продавцы»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_продавца | Числовой |
| Фамилия | Текстовый |
| Имя | Текстовый |
| Отчество | Текстовый |
| Телефон | Текстовый |

Таблица 7 – Поля таблицы «Продажи»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Код\_продажи | Числовой |
| Код\_покупателя | Числовой |
| Код\_упаковки\_семян | Числовой |
| Код\_продавца | Числовой |
| Дата\_продажи | Дата |
| Оплата\_наличными | Логический |

Таблица 8 – Поля таблицы «Пользователи»

|  |  |
| --- | --- |
| Название поля | Тип данных |
| Логин | Текстовый |
| Пароль | Числовой |
| Почта | Текстовый |

На рисунке 13 представлены все созданные таблицы в обозревателе объектов базы данных.

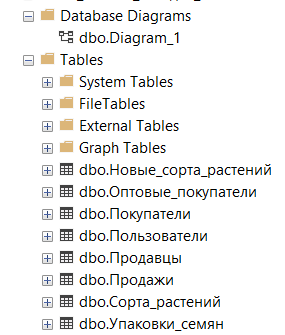
****

Рисунок 13 – Созданные таблицы

После того, как таблицы созданы, необходимо создать между ними связи.

В приложении 5 на рисунке 14 представлена физическая модель данных, отражающая все таблицы и их поля, а также связи между таблицами.

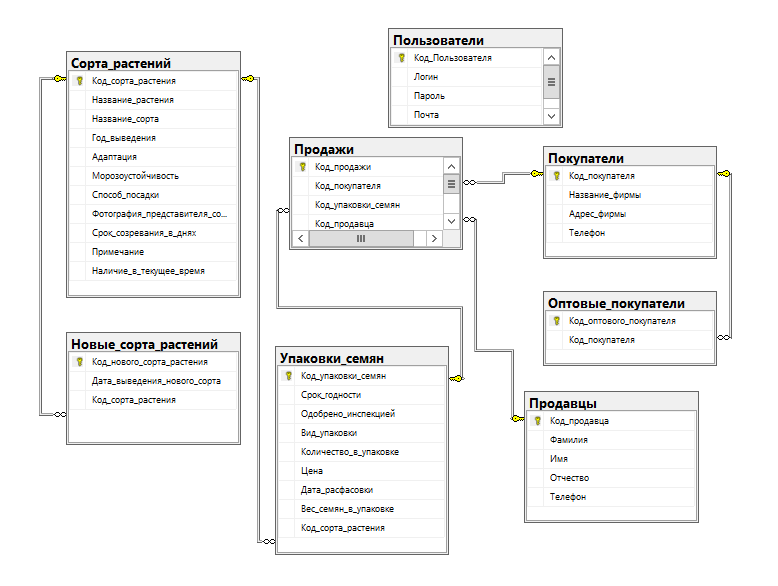


Рисунок 14 – Физическая модель базы данных

## Разработка программных модулей информационной системы

Разработанные в MS Visual Studio формы продемонстрированы в приложении 6 на рисунках 15-20.

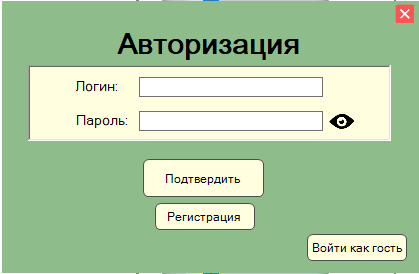


Рисунок 15 – Форма “Авторизация”

Свойства элементов формы Авторизация представлены в таблице 9.

Таблица 9 – Свойства элементов формы Авторизация

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Авторизация | FormBorderStyle | None |
| Text | Авторизация |
| StartPosition | CenterScreen |
| Size | 417; 272 |

Событийно-управляемые процедуры в форме авторизации представлены в таблице 10

Таблица 10 – Событийно-управляемые процедуры в форме «Авторизация»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Авторизация | public partial class Авторизация : Form  {  public int ogr = 5;  public Авторизация()  {  InitializeComponent();  }  } |
| Кнопка\_подтверждения\_Click | private void Кнопка\_подтверждения\_Click(object sender, EventArgs e)  {  if (Логин\_textBox1.Text == "" || Пароль\_textBox2.Text == "")  {  MessageBox.Show("Поля не должны быть пустыми", "Ошибка");  }  else  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True");  sqlConnect.Open();  SqlDataAdapter da = new SqlDataAdapter("select \* from Пользователи", sqlConnect);  DataTable dt = new DataTable();  da.Fill(dt);  int flag = -1;  for (int i = 0; i < dt.Rows.Count; i++)  {  if ((dt.Rows[i]["Права\_доступа"].ToString() == "Администратор") && (dt.Rows[i]["Логин"].ToString() == Логин\_textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() == Пароль\_textBox2.Text))  flag = 0;  if ((dt.Rows[i]["Права\_доступа"].ToString() == "Продавец") && (dt.Rows[i]["Логин"].ToString() == Логин\_textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() == Пароль\_textBox2.Text))  {  flag = 1;  }  if ((dt.Rows[i]["Права\_доступа"].ToString() == "Покупатель") && (dt.Rows[i]["Логин"].ToString() == Логин\_textBox1.Text) && (dt.Rows[i]["Пароль"].ToString() == Пароль\_textBox2.Text))  {  flag = 2;  }  }  if ((flag == 2) || (flag == 1) || (flag == 0))  {  if (flag == 0)  {  this.Hide();  Администратор UsrFrm = new Администратор();  UsrFrm.Show();  }  if (flag == 1)  {  this.Hide();  Продавец UsrFrm5 = new Продавец();  UsrFrm5.Show();  }  if (flag == 2)  {  this.Hide();  Покупатель UsrFrm6 = new Покупатель();  UsrFrm6.Show();  }  }  else  MessageBox.Show("Пересоздайте пользователя! Такого пользователя не существует.");  }  } |
| Кнопка\_Рагистрация\_Click | private void Кнопка\_Рагистрация\_Click(object sender, EventArgs e)  {  this.Hide();  Регистрация UsrFrm4 = new Регистрация();  UsrFrm4.Show();  } |
| Кнопка\_Войти\_как\_гость\_Click | private void Кнопка\_Войти\_как\_гость\_Click(object sender, EventArgs e)  {  this.Hide();  Гость UsrFrm3 = new Гость();  UsrFrm3.Show();  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Видимость\_пароля\_Click | private void Видимость\_пароля\_Click(object sender, EventArgs e)  {  a++;  if (a==1)  {  Пароль\_textBox2.PasswordChar = '\0';  }  if (a==2)  {  Пароль\_textBox2.PasswordChar = '\*';  a = 0;  }  } |
| Пароль\_KeyPress | private void Пароль\_KeyPress(object sender, KeyPressEventArgs e)  {  if (!char.IsDigit(e.KeyChar) && (e.KeyChar != (char)Keys.Back))  {  e.Handled = true;  MessageBox.Show("Поле не может содержать буквы");  }  } |
| Авторизация\_MouseMove | private void Авторизация\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Авторизация\_MouseDown | private void Авторизация\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |

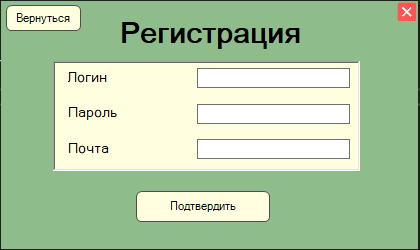


Рисунок 16 – Форма “Регистрация”

Свойства элементов формы Регистрация представлены в таблице 11.

Таблица 11 – Свойства элементов формы Регистрация

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Регистрация | FormBorderStyle | None |
| Text | Регистрация |
| StartPosition | CenterScreen |
| Size | 417; 248 |

Событийно-управляемые процедуры в форме Регистрация представлены в таблице 12

Таблица 12 – Событийно-управляемые процедуры в форме «Регистрация»

| Объект | Событийно-управляемые процедуры | |
| --- | --- | --- |
| class Регистрация | public partial class Регистрация : Form  {  public Регистрация()  {  InitializeComponent();  }  } | |
| Кнопка\_подтверждения\_Click | private void Кнопка\_подтверждения\_Click(object sender, EventArgs e)  {  if ((Логин\_textBox5.Text == "") || (Пароль\_textBox4.Text == "") || (Почта\_textBox2.Text == ""))  {  MessageBox.Show("Не все данные заполнены!", "Ошибка");  }  else  {  using (SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True"))  {  SqlDataAdapter info = new SqlDataAdapter($"INSERT INTO Пользователи (Логин,Пароль,Почта) VALUES('{Логин\_textBox5.Text}','{Пароль\_textBox4.Text}','{Почта\_textBox2.Text}');", sqlConnect);  DataTable tt = new DataTable();  info.Fill(tt);  }  MessageBox.Show("Вы зарегестрированы как 'Покупатель'", "Уведомление");  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  }  } | |
| Пароль\_KeyPress | private void Пароль\_KeyPress(object sender, KeyPressEventArgs e)  {  if (!char.IsDigit(e.KeyChar) && (e.KeyChar != (char)Keys.Back))  {  e.Handled = true;  MessageBox.Show("Поле не может содержать буквы");  }  } | |
| Вернуться\_Click | | private void Вернуться\_Click(object sender, EventArgs e)  {  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  } |
| выйти\_Click | | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Регистрация\_MouseMove | | private void Регистрация\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Регистрация\_MouseDown | | private void Регистрация\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |

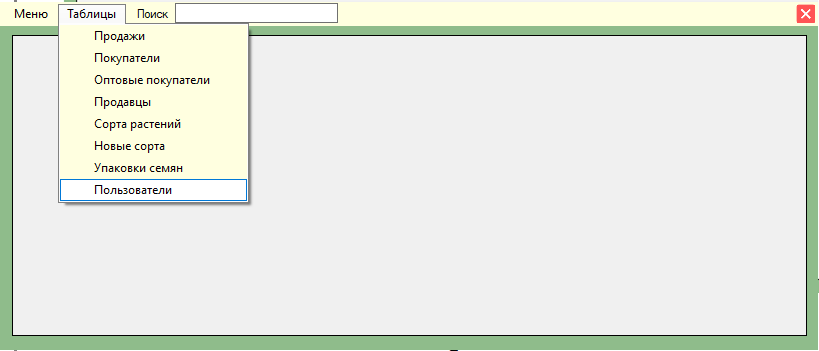


Рисунок 17 – Форма “ Администратора”

Свойства элементов формы Администратор представлены в таблице 13.

Таблица 13 – Свойства элементов формы Администратор

| Объект | Свойство | Значение |
| --- | --- | --- |
| Администратор | FormBorderStyle | None |
| Text | Администратор |
| StartPosition | CenterScreen |
| Size | 819; 348 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Администратор представлены в таблице 14.

Таблица 14 – Событийно-управляемые процедуры в форме «Администратор»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Администратор | public partial class Администратор : Form  {  public Администратор()  {  InitializeComponent();  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  }  public void tab(int i)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True");  sqlConnect.Open();  switch (i)  {  case 0: Поле\_для\_таблиц\_DataGridView1.DataSource = продажиBindingSource; break;  case 1: Поле\_для\_таблиц\_DataGridView1.DataSource = покупателиBindingSource; break;  case 2: Поле\_для\_таблиц\_DataGridView1.DataSource = оптовыепокупателиBindingSource; break;  case 3: Поле\_для\_таблиц\_DataGridView1.DataSource = продавцыBindingSource; break;  case 4: Поле\_для\_таблиц\_DataGridView1.DataSource = сортарастенийBindingSource; break;  case 5: Поле\_для\_таблиц\_DataGridView1.DataSource = новыесортарастенийBindingSource; break;  case 6: Поле\_для\_таблиц\_DataGridView1.DataSource = упаковкисемянBindingSource; break;  case 7: Поле\_для\_таблиц\_DataGridView1.DataSource = пользователиBindingSource; break;  }  } |
| вернуться\_Click | private void вернуться\_Click(object sender, EventArgs e)  {  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  } |
| обновить\_Click | private void обновить\_Click(object sender, EventArgs e)  {  switch (c)  {  case 0: продажиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи); break;  case 1: покупателиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели); break;  case 2: оптовые\_покупателиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели); break;  case 3: продавцыTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы); break;  case 4: сорта\_растенийTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений); break;  case 5: новые\_сорта\_растенийTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений); break;  case 6: упаковки\_семянTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян); break;  case 7: пользователиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Пользователи); break;  }  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Таблица\_Продажи\_Click | private void Таблица\_Продажи\_Click(object sender, EventArgs e)  {  tab(0);  c = 0;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Покупатели\_Click | private void Таблица\_Покупатели\_Click(object sender, EventArgs e)  {  tab(1);  c = 1;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Оптовые\_покупатели\_Click | private void Таблица\_Оптовые\_покупатели\_Click(object sender, EventArgs e)  {  tab(2);  c = 2;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Продавцы\_Click | private void Таблица\_Продавцы\_Click(object sender, EventArgs e)  {  tab(3);  c = 3;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Сорта\_растений\_Click | private void Таблица\_Сорта\_растений\_Click(object sender, EventArgs e)  {  tab(4);  c = 4;  a = true;  Фотография\_представителя\_PictureBox2.Visible = true;  Кнопка\_найти\_фотографию\_Button1.Visible = true;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(598, 301);  } |
| Таблица\_Новые\_сорта\_растений\_Click | private void Таблица\_Новые\_сорта\_растений\_Click(object sender, EventArgs e)  {  tab(5);  c = 5;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Упаковки\_семян\_Click | private void Таблица\_Упаковки\_семян\_Click(object sender, EventArgs e)  {  tab(6);  c = 6;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Таблица\_Пользователи\_Click | private void Таблица\_Пользователи\_Click(object sender, EventArgs e)  {  tab(7);  c = 7;  Фотография\_представителя\_PictureBox2.Visible = false;  Кнопка\_найти\_фотографию\_Button1.Visible = false;  Поле\_для\_таблиц\_DataGridView1.Size = new Size(794, 301);  } |
| Поисковая\_строка\_TextChanged | private void Поисковая\_строка\_TextChanged(object sender, EventArgs e)  {  for (int i = 0; i < Поле\_для\_таблиц\_DataGridView1.RowCount; i++)  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = false;  for (int j = 0; j < Поле\_для\_таблиц\_DataGridView1.ColumnCount; j++)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value != null)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value.ToString().Contains(Строка\_поиска\_TextBox1.Text))  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = true;  break;  }  }  } |
| Поле\_для\_таблиц\_CellClick | private void Поле\_для\_таблиц\_CellClick(object sender, DataGridViewCellEventArgs e)  {  id = int.Parse(Поле\_для\_таблиц\_DataGridView1.SelectedCells[0].RowIndex.ToString());  if (c == 4)  {  try  {  pic = Поле\_для\_таблиц\_DataGridView1[7, id].Value.ToString();  выйти\_PictureBox1.Image = Image.FromFile(pic);  if (выйти\_PictureBox1.Image != null)  {  Фотография\_представителя\_PictureBox2.Visible = true;  }  }  catch  {  MessageBox.Show("Отсутствует изображение");  }  }  } |
| Администратор\_MouseMove | private void Администратор\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Администратор\_MouseDown | private void Администратор\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |
| Администратор\_Load | private void Администратор\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян". При необходимости она может быть перемещена или удалена.  this.упаковки\_семянTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений". При необходимости она может быть перемещена или удалена.  this.сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи". При необходимости она может быть перемещена или удалена.  this.продажиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы". При необходимости она может быть перемещена или удалена.  this.продавцыTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Пользователи". При необходимости она может быть перемещена или удалена.  this.пользователиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Пользователи);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели". При необходимости она может быть перемещена или удалена.  this.покупателиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели". При необходимости она может быть перемещена или удалена.  this.оптовые\_покупателиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений". При необходимости она может быть перемещена или удалена.  this.новые\_сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet1.Пользователи". При необходимости она может быть перемещена или удалена.  } |

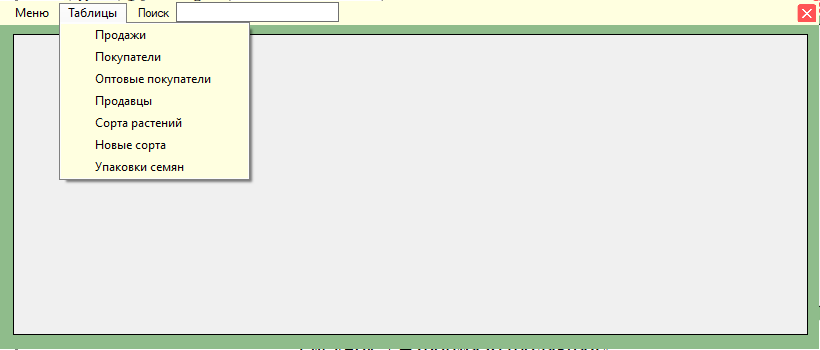


Рисунок 18 – Форма “Продавца”

Свойства элементов формы Продавец представлены в таблице 15.

Таблица 15 – Свойства элементов формы Продавец

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Продавец | FormBorderStyle | None |
| Text | Продавец |
| StartPosition | CenterScreen |
| Size | 819; 348 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Продавец представлены в таблице 16.

Таблица 16 – Событийно-управляемые процедуры в форме «Продавец»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Продавец | public partial class Продавец : Form  {  public Продавец()  {  InitializeComponent();  }  public void Tub(int i)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True");  sqlConnect.Open();  SqlDataAdapter[] so = new SqlDataAdapter[7];  so[0] = new SqlDataAdapter("select \* from Продажи", sqlConnect);  so[1] = new SqlDataAdapter("select \* from Покупатели", sqlConnect);  so[2] = new SqlDataAdapter("select \* from Оптовые\_покупатели", sqlConnect);  so[3] = new SqlDataAdapter("select \* from Продавцы", sqlConnect);  so[4] = new SqlDataAdapter("select \* from Сорта\_растений", sqlConnect);  so[5] = new SqlDataAdapter("select \* from Новые\_сорта\_растений", sqlConnect);  so[6] = new SqlDataAdapter("select \* from Упаковки\_семян", sqlConnect);  DataSet ps = new DataSet();  so[i].Fill(ps);  Поле\_для\_таблиц\_DataGridView1.DataSource = ps.Tables[0];  } |
| Продавец\_Load | private void Продавец\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян". При необходимости она может быть перемещена или удалена.  this.упаковки\_семянTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений". При необходимости она может быть перемещена или удалена.  this.сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи". При необходимости она может быть перемещена или удалена.  this.продажиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы". При необходимости она может быть перемещена или удалена.  this.продавцыTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели". При необходимости она может быть перемещена или удалена.  this.покупателиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели". При необходимости она может быть перемещена или удалена.  this.оптовые\_покупателиTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений". При необходимости она может быть перемещена или удалена.  this.новые\_сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений);  } |
| обновить\_Click | private void обновить\_Click(object sender, EventArgs e)  {  switch (t)  {  case 0: продажиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Продажи); break;  case 1: покупателиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Покупатели); break;  case 2: оптовые\_покупателиTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Оптовые\_покупатели); break;  case 3: продавцыTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Продавцы); break;  case 4: сорта\_растенийTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений); break;  case 5: новые\_сорта\_растенийTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Новые\_сорта\_растений); break;  case 6: упаковки\_семянTableAdapter.Update(уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян); break;  }  } |
| вернуться\_Click | private void вернуться\_Click(object sender, EventArgs e)  {  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  } |
| Таблица\_Продажи\_Click | private void Таблица\_Продажи\_Click(object sender, EventArgs e)  {  Tub(0);  t = 0;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = false;  } |
| Таблица\_Покупатели\_Click | private void Таблица\_Покупатели\_Click(object sender, EventArgs e)  {  Tub(1);  t = 1;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = false;  } |
| Таблица\_Оптовые\_покупатели\_Click | private void Таблица\_Оптовые\_покупатели\_Click(object sender, EventArgs e)  {  Tub(2);  t = 2;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = false;  } |
| Таблица\_Продавцы\_Click | private void Таблица\_Продавцы\_Click(object sender, EventArgs e)  {  Tub(3);  t = 3;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = true;  } |
| Таблица\_Сорта\_растений\_Click | private void Таблица\_Сорта\_растений\_Click(object sender, EventArgs e)  {  Tub(4);  t = 4;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = true;  } |
| Таблица\_Новые\_сорта\_растений\_Click | private void Таблица\_Новые\_сорта\_растений\_Click(object sender, EventArgs e)  {  Tub(5);  t = 5;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = true;  } |
| Таблица\_Упаковки\_семян\_Click | private void Таблица\_Упаковки\_семян\_Click(object sender, EventArgs e)  {  Tub(6);  t = 6;  a = true;  Поле\_для\_таблиц\_DataGridView1.ReadOnly = true;  } |
| Поисковая\_строка\_TextChanged | private void Поисковая\_строка\_TextChanged(object sender, EventArgs e)  {  for (int i = 0; i < Поле\_для\_таблиц\_DataGridView1.RowCount; i++)  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = false;  for (int j = 0; j < Поле\_для\_таблиц\_DataGridView1.ColumnCount; j++)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value != null)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value.ToString().Contains(Строка\_поиска\_TextBox1.Text))  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = true;  break;  }  }  } |
| Продавец\_MouseMove | private void Продавец\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Продавец\_MouseDown | private void Продавец\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |

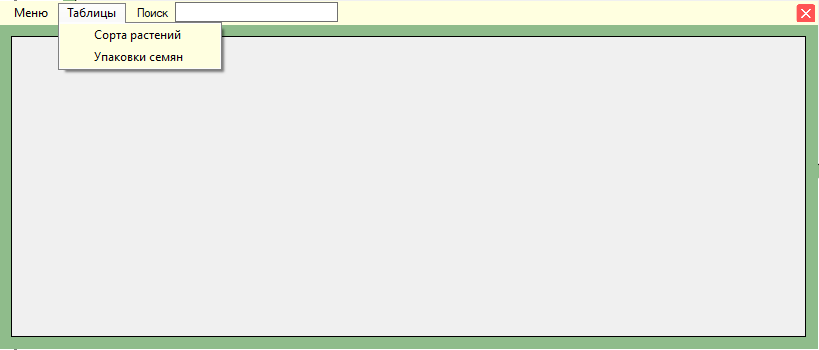


Рисунок 19 – Форма “Покупателя”

Свойства элементов формы Покупатель представлены в таблице 17.

Таблица 17 – Свойства элементов формы Покупатель

| Объект | Свойство | Значение |
| --- | --- | --- |
| Покупатель | FormBorderStyle | None |
| Text | Покупатель |
| StartPosition | CenterScreen |
| Size | 819; 348 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Покупатель представлены в таблице 18.

Таблица 18 – Событийно-управляемые процедуры в форме «Покупатель»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Покупатель | public partial class Покупатель : Form  {  public Покупатель()  {  InitializeComponent();  }  public void Tob(int i)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True");  sqlConnect.Open();  SqlDataAdapter[] so = new SqlDataAdapter[2];  so[0] = new SqlDataAdapter("select Название\_растения, Название\_сорта, Год\_выведения, Адаптация, Морозоустойчивость, Способ\_посадки, Фотография\_представителя\_сорта, Срок\_созревания\_в\_днях, Примечание, Наличие\_в\_текущее\_время from Сорта\_растений", sqlConnect);  so[1] = new SqlDataAdapter("select Срок\_годности, Одобрено\_инспекцией, Вид\_упаковки, Количество\_в\_упаковке, Цена, Дата\_расфасовки, Вес\_семян\_в\_упаковке, Код\_сорта\_растения from Упаковки\_семян", sqlConnect);  DataSet ps = new DataSet();  so[i].Fill(ps);  Поле\_для\_таблиц\_DataGridView1.DataSource = ps.Tables[0];  } |
| вернуться\_Click | private void вернуться\_Click(object sender, EventArgs e)  {  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  } |
| Таблица\_Сорта\_растений\_Click | private void Таблица\_Сорта\_растений\_Click(object sender, EventArgs e)  {  Tob(0);  a = true;  } |
| Таблица\_Упаковки\_семян\_Click | private void Таблица\_Упаковки\_семян\_Click(object sender, EventArgs e)  {  Tob(1);  a = true;  } |
| Покупатель\_Load | private void Покупатель\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян". При необходимости она может быть перемещена или удалена.  this.упаковки\_семянTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений". При необходимости она может быть перемещена или удалена.  this.сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений);  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Поисковая\_строка\_TextChanged | private void Поисковая\_строка\_TextChanged(object sender, EventArgs e)  {  for (int i = 0; i < Поле\_для\_таблиц\_DataGridView1.RowCount; i++)  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = false;  for (int j = 0; j < Поле\_для\_таблиц\_DataGridView1.ColumnCount; j++)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value != null)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value.ToString().Contains(Строка\_поиска\_TextBox1.Text))  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = true;  break;  }  }  } |
| Покупатель\_MouseMove | private void Покупатель\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Покупатель\_MouseDown | private void Покупатель\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |

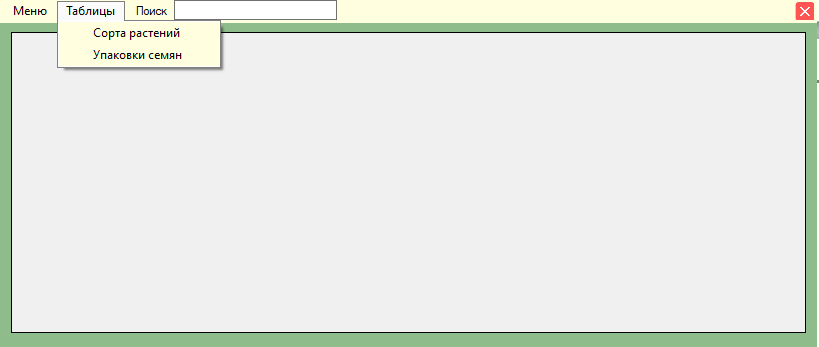


Рисунок 20 – Форма “Гостя”

Свойства элементов формы Гость представлены в таблице 19.

Таблица 19 – Свойства элементов формы Гость

|  |  |  |
| --- | --- | --- |
| Объект | Свойство | Значение |
| Гость | FormBorderStyle | None |
| Text | Гость |
| StartPosition | CenterScreen |
| Size | 819; 348 |
| ShowIcon | False |

Событийно-управляемые процедуры в форме Гость представлены в таблице 20.

Таблица 20 – Событийно-управляемые процедуры в форме «Гость»

| Объект | Событийно-управляемые процедуры |
| --- | --- |
| class Гость | public partial class Гость : Form  {  public Гость()  {  InitializeComponent();  }  public void Tob(int i)  {  SqlConnection sqlConnect = new SqlConnection("Data Source=sql;Initial Catalog = уП01\_ИСПП5\_Швидко\_ИА; Integrated Security = True");  sqlConnect.Open();  SqlDataAdapter[] so = new SqlDataAdapter[2];  so[0] = new SqlDataAdapter("select Название\_растения, Название\_сорта, Год\_выведения, Адаптация, Морозоустойчивость, Способ\_посадки, Фотография\_представителя\_сорта, Срок\_созревания\_в\_днях, Примечание, Наличие\_в\_текущее\_время from Сорта\_растений", sqlConnect);  so[1] = new SqlDataAdapter("select Срок\_годности, Одобрено\_инспекцией, Вид\_упаковки, Количество\_в\_упаковке, Цена, Дата\_расфасовки, Вес\_семян\_в\_упаковке, Код\_сорта\_растения from Упаковки\_семян", sqlConnect);  DataSet ps = new DataSet();  so[i].Fill(ps);  Поле\_для\_таблиц\_DataGridView1.DataSource = ps.Tables[0];  } |
| Гость\_Load | private void Гость\_Load(object sender, EventArgs e)  {  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян". При необходимости она может быть перемещена или удалена.  this.упаковки\_семянTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений". При необходимости она может быть перемещена или удалена.  this.сорта\_растенийTableAdapter.Fill(this.уП01\_ИСПП5\_Швидко\_ИАDataSet.Сорта\_растений);  // TODO: данная строка кода позволяет загрузить данные в таблицу "уП01\_ИСПП5\_Швидко\_ИАDataSet.Упаковки\_семян". При необходимости она может быть перемещена или удалена.  } |
| вернуться\_Click | private void вернуться\_Click(object sender, EventArgs e)  {  Авторизация AvtFrm = new Авторизация();  AvtFrm.Show();  this.Hide();  } |
| выйти\_Click | private void выйти\_Click(object sender, EventArgs e)  {  Application.Exit();  } |
| Таблица\_Сорта\_растений\_Click | private void Таблица\_Сорта\_растений\_Click(object sender, EventArgs e)  {  Tob(0);  a = true;  } |
| Таблица\_Упаковки\_семян\_Click | private void Таблица\_Упаковки\_семян\_Click(object sender, EventArgs e)  {  Tob(1);  a = true;  } |
| Поисковая\_строка\_TextChanged | private void Поисковая\_строка\_TextChanged(object sender, EventArgs e)  {  for (int i = 0; i < Поле\_для\_таблиц\_DataGridView1.RowCount; i++)  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = false;  for (int j = 0; j < Поле\_для\_таблиц\_DataGridView1.ColumnCount; j++)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value != null)  if (Поле\_для\_таблиц\_DataGridView1.Rows[i].Cells[j].Value.ToString().Contains(Строка\_поиска\_TextBox1.Text))  {  Поле\_для\_таблиц\_DataGridView1.Rows[i].Selected = true;  break;  }  }  } |
| Гость\_MouseMove | private void Гость\_MouseMove(object sender, MouseEventArgs e)  {  if (e.Button == MouseButtons.Left)  {  Left += e.X - Mouse.X;  Top += e.Y - Mouse.Y;  }  } |
| Гость\_MouseDown | private void Гость\_MouseDown(object sender, MouseEventArgs e)  {  Mouse = new Point(e.X, e.Y);  } |

Вывод по главе: была полностью спроектирована и разработана система базы данных, были полностью спроектированы и разработаны все программные модули информационной системы “Продажа сортов растений”.

# РАЗРАБОТКА ТЕСТОВЫХ НАБОРОВ И ТЕСТОВЫХ СЦЕНАРИЕВ

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом.

Тестовый сценарий – это неавтоматизированный или автоматизированный сценарий, содержащий инструкции по реализации тестового набора. Тестовый сценарий может быть написан вручную (для выполнения человеком) либо полностью или частично автоматизирован.

## Назначение эксперимента. Выбор и обоснование методики проведения тестирования

В данном разделе проведены два вида тестирования. Провести анализ тестирования и выбрать, как будет проходить тестирование ИС.

Модульное тестирование Цель: проверить, что код работает именно так, как должен (при заданных входных параметрах выдает предполагаемый результат).

Анализ покрытия кода тестами. Цель: проверить, что весь наш код отрабатывает при модульном тестировании, что нет не участвующих в тестировании участков кода.

Анализ стилистики кода. Цель: проверка кода на соблюдение стандартов разработки (отраслевых, корпоративных, принятых на проекте).

Анализ производительности. Цель: проанализировать производительность ключевых операций, потребление памяти, утечку памяти.

## Технология тестирования

Рассмотрим, какие технологии можно применить для модульного тестирования и покрытия кода тестами.

Как в ходе выполнения проекта, так и при развитии системы, изменения в ее алгоритмы вносятся в соответствии с техническими заданиями, хотя данный документ может иметь различные названия. В технических заданиях, как правило, описываются критерии приемки: как мы и заказчик убедимся в том, что заказанный функционал работает верно.

В качестве примера можно привести такой упрощенный критерий:

* Исходные данные: в систему внесен сорт растения, который в дальнейшем приобретёт покупатель.
* Действия пользователя: пользователь ищет в таблице упаковку семян с определённым сортом растений и сколько он будет стоить и т п.
* В результате в результате отображена цена упаковки семян 500 руб.
* Вручную такое тестирование, как правило, выполняется единожды, поскольку выполнять каждый раз полный комплект тестов очень трудоемко.
* Нет гарантии того, что тесты, описанные в техническом задании, покрывают функциональность полностью. Т.е. даже если тесты описаны и проводятся регулярно, вероятно наличие не протестированного функционала.

Автоматизированное тестирование лишено данных недостатков. Конечно, возникает отдельная задача кодирования данного теста на основании критериев приемки, но это скорее дисциплинирующий фактор…

Прежде чем начинать юзабилити-тестирование необходимо собрать группу пользователей, которые будут тестировать данную информационную систему. Количество привлеченных пользователей должно быть не менее пяти человек. Пользователям будет представлен список некоторых вопросов, относящихся к графическому интерфейсу.

В таблице 21 представлены сведения о пользователях.

Таблица 21 – Информация о пользователях

|  |  |  |  |
| --- | --- | --- | --- |
| ФИО | Пол | Возраст | Образование |
| Кузнецова Анна Павловна | Женский | 38 | Полное общее |
| Сидоров Александр Васильевич | Мужской | 45 | Среднее специальное |
| Козлова Екатерина Александровна | Женский | 35 | Среднее специальное |
| Зайцева Мария Анатольевна | Женский | 23 | Среднее специальное |
| Васильев Владимир Николаевич | Мужской | 43 | Основное общее |

## Результаты проведения тестирования

В ходе тестирования был применен метод нагрузочного тестирования и юзабилити-тестирования, которые проверяют работу при нагрузке системы и сделать простым и удобным в пользовании информационной системы.

Суть тестирования заключается в проверке работы всех обязательных функций.

Результаты проведения нагрузочного тестирования

Для проведения нагрузочного тестирования составлена таблица 22, в которой указано количество запущенных копий программного продукта, нагрузка на процесс и оперативную память.

Таблица 22 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| Число копий | Нагрузка на процессор (%) | Нагрузка на память (%) |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |

Исходя из результатов тестирования, можно прийти к выводу, что нагрузка, оказываемая на центральный процессор и память, не имеет критичных значений. Это значит, что показатель производительности информационной системы на приемлемом уровне, то есть сбои и длительная загрузка не должны помешать пользователю при работе с данным программным продуктом.

Вывод по главе: были проведены тесты путём проведения модульного тестирования, анализа покрытия кода тестами, анализа стилистики кода и анализом производительности. Также было проведено нагрузочное тестирование, в результате чего нагрузка на процессор и память не имела критичных значений.

# ЗАКЛЮЧЕНИЕ

В результате учебной практики был проведен анализ предметной области, спроектирована и разработана информационная система «Продажа сортов растений», серверная часть которой была реализована в MS SQL, а клиентская часть в MS Visual Studio.

В MS SQL была создана база данных для хранения информации о вагонах, персонале, должностях, видов ремонта и о самом ремонте вагонов.

Клиентская часть, реализованная в MS Visual Studio, отражает все хранящиеся в базе данных таблицы. Также разработаны некоторые запросы, позволяющие редактировать и удалять существующие данные, добавлять новые и осуществлять поиск по таблицам.

Цели учебной практики были достигнуты путем проведения анализа предметной области, проектирования информационной системы и разработки программных модулей.

Это позволяет сделать вывод, что поставленные цели и задачи выполнены в полном объеме.

В дальнейшем данная информационная система может быть модернизирована путем добавления формы личного кабинета, создания версии для слабовидящих или редизайна.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

# ПРИЛОЖЕНИЕ 1



Рисунок 1 – Диаграмма вариантов использования

# ПРИЛОЖЕНИЕ 2

# ПРИЛОЖЕНИЕ 3

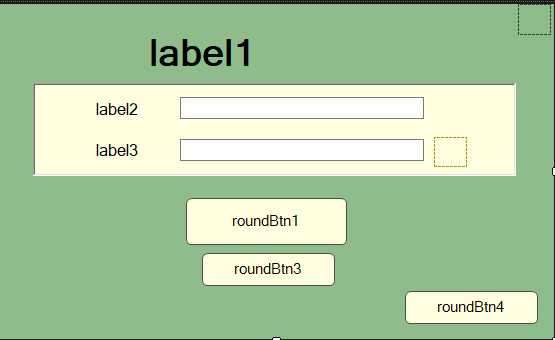


Рисунок 7 – Окно авторизации

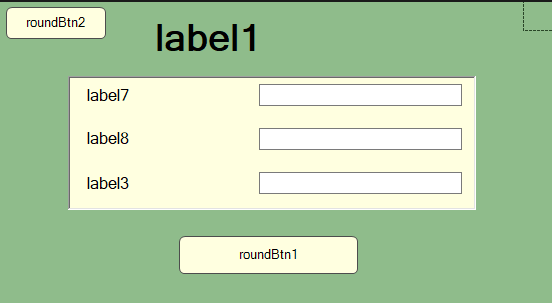


Рисунок 8 – Окно регистрации

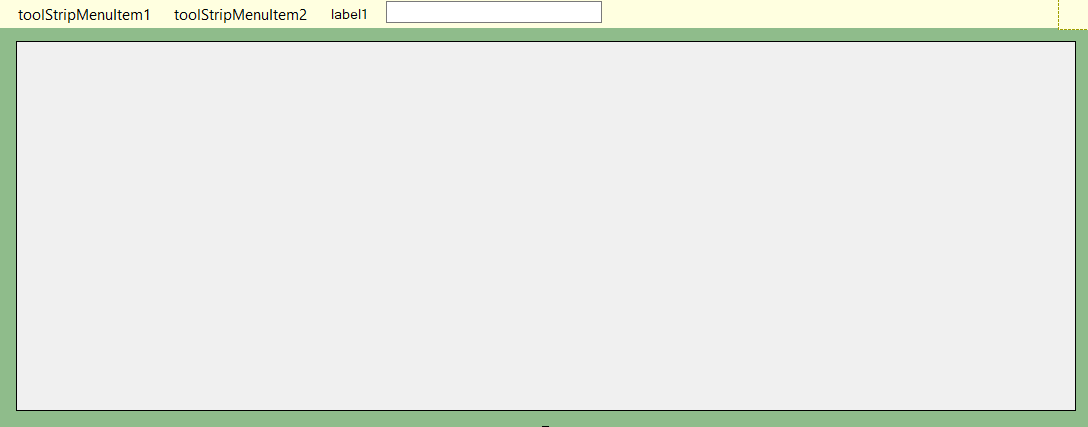


Рисунок 8 – Форма «Администратора»

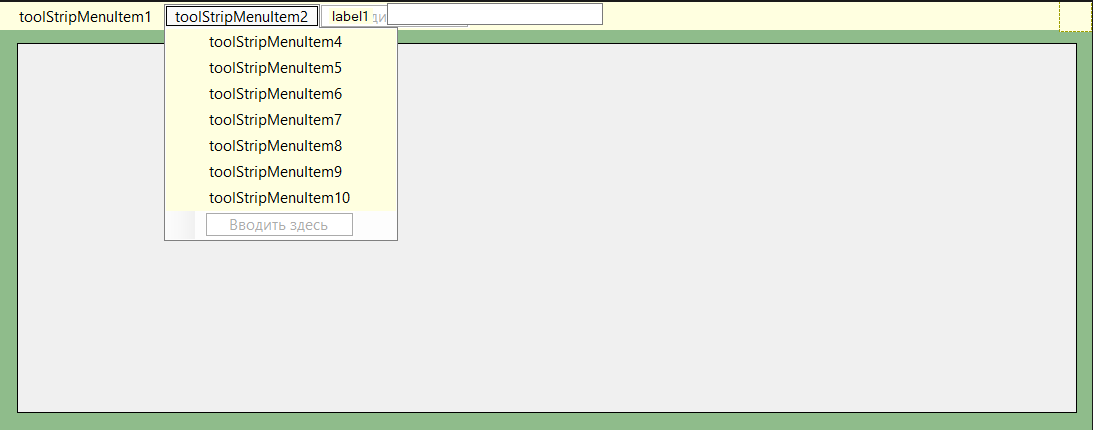


Рисунок 9 – Форма «Продавца»

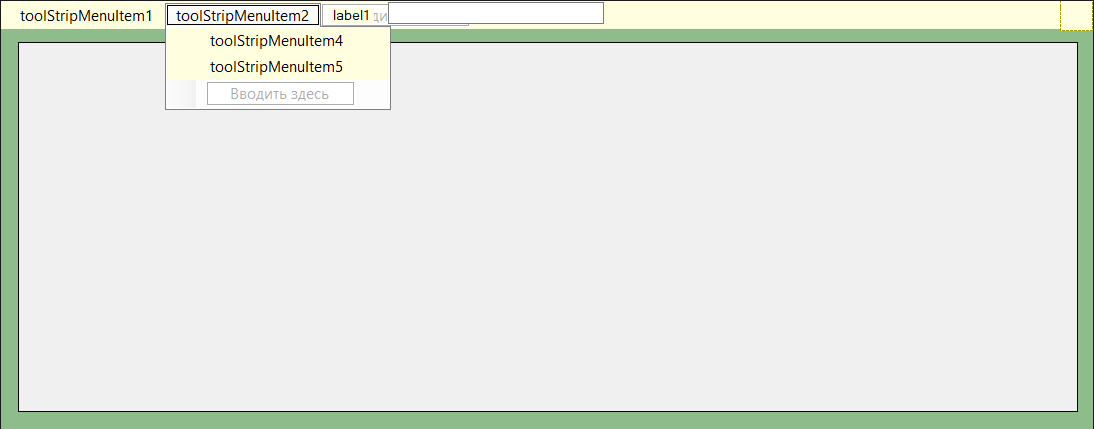


Рисунок 10 – Форма «Покупателя»

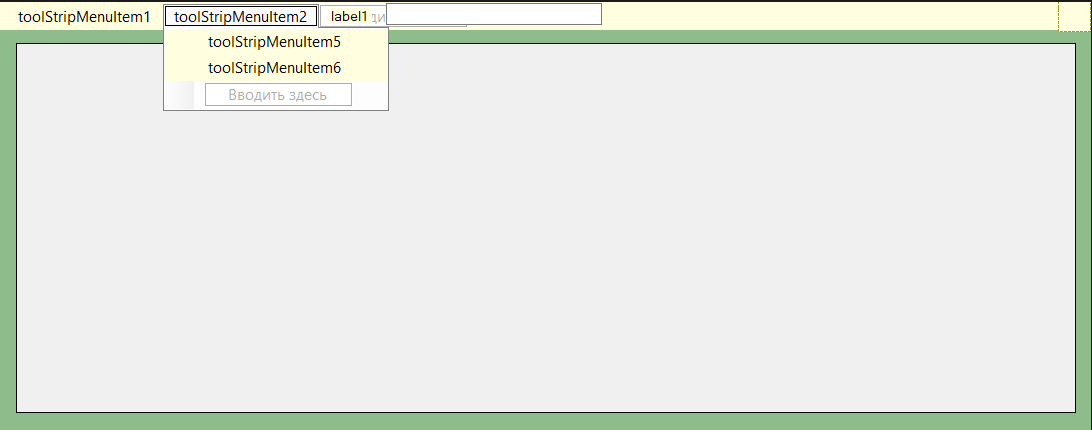


Рисунок 11 – Форма «Гостя»

# ПРИЛОЖЕНИЕ 4

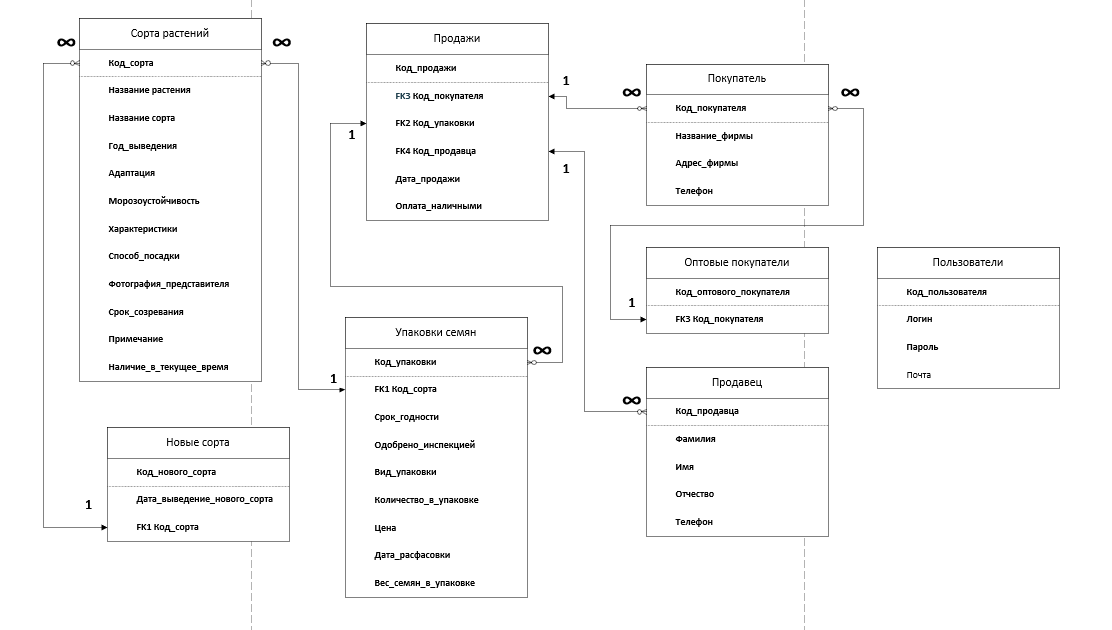


Рисунок 12 – Логическая модель данных

# ПРИЛОЖЕНИЕ 5

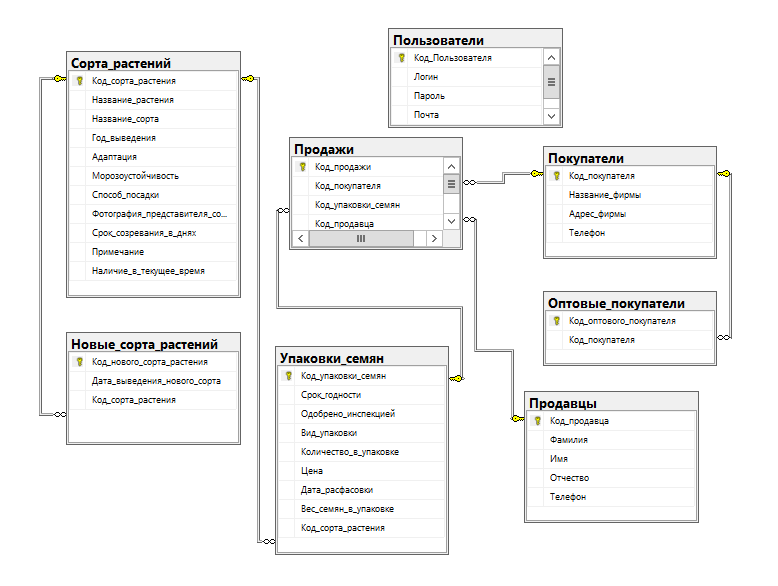


Рисунок 14 – Физическая модель базы данных

# ПРИЛОЖЕНИЕ 6

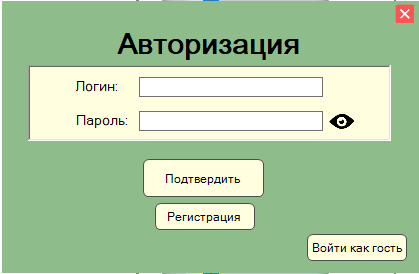


Рисунок 15 – Форма “Авторизация”

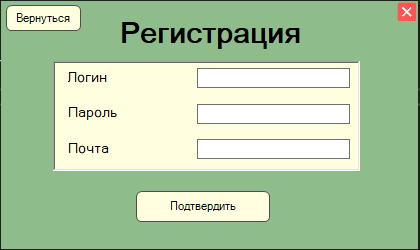


Рисунок 16 – Форма “Регистрация”

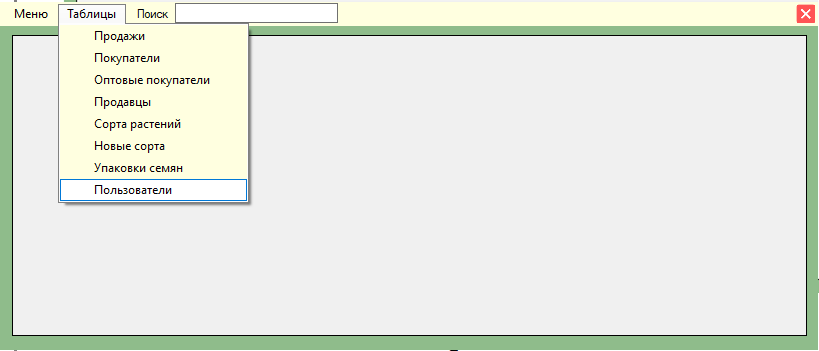


Рисунок 17 – Форма “ Администратора”

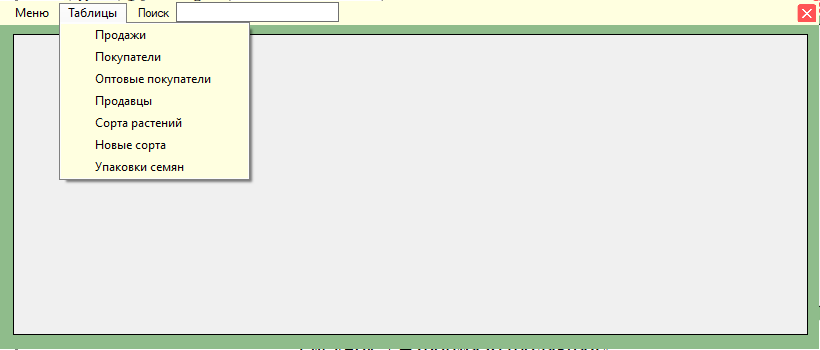


Рисунок 18 – Форма “Продавца”

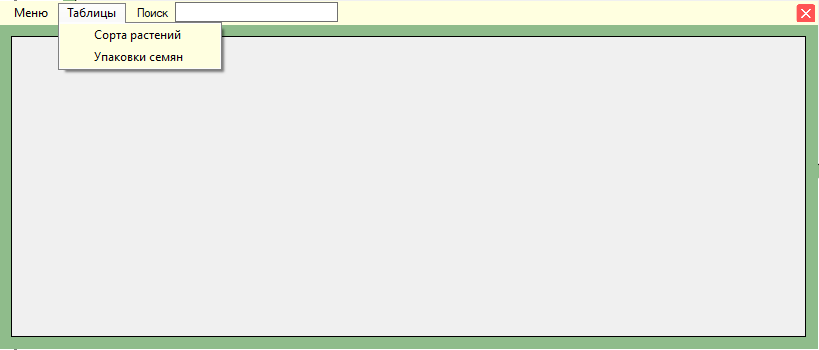


Рисунок 19 – Форма «Покупателя»

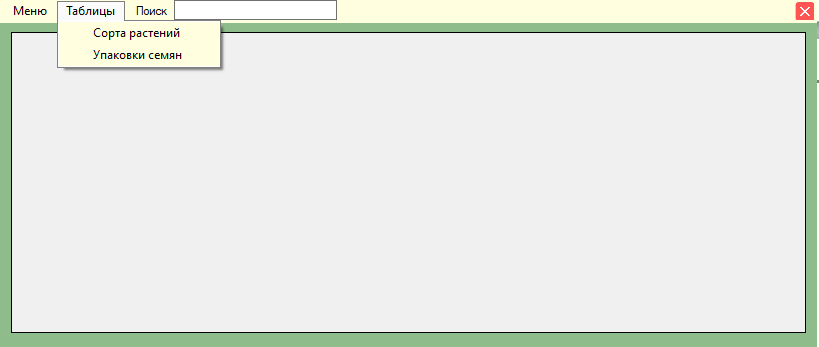


Рисунок 20 – Форма «Гостя»