**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙ РАБОТЕ**

**ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНЫХ СИСТЕМ»**

**НА ТЕМУ: *«ВСПОМОГАТЕЛЬНАЯ ПОДСИСТЕМА ИДЕНТИФИКАЦИИ АВТОМОБИЛЯ ПО НОМЕРНОМУ ЗНАКУ НА ПРОПУСКНОМ ПУНКТЕ»***

Студент ИУ6-53 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мамадаев И.М.**

(Подпись, дата)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Фетисов М.В.**

(Подпись, дата)

*2018 г.*

**РЕФЕРАТ**

Расчётно-пояснительная записка x страниц, x рисунков, x таблиц, x источников, x приложений.

Объектом разработки является прототип программы, предназначенной для автоматического распознания автомобильного номера по его отсканированному (сфотографированному) изображению и авторизации пользователей, имеющих доступ к ПИАНЗ.

Цель работы – создание описанного выше приложения, обеспечивающего:

* Регистрацию пользователей;
* Авторизацию в программе;
* Выбор изображения с автомобильным номером;
* Получение результатов в виде распознанного номера и времени распознания.

В процессе работы проводилось исследование предметной области и проектирование необходимого приложения, в результате чего было создано приложение.

Поставленная цель достигается средствами языка программирования C# и библиотекой компьютерного зрения для распознания объектов с изображений «emguCV». В качестве среды разработки используется «Visual Studio 2017». Для контроля версий используется интернет-ресурс «GitHub». Хранение данных пользователей было организовано с помощью средств стандартной СУБД для «Visual Studio» - Microsoft SQL Server.

Приложение позволит упростить работу сотрудника охраны, находящегося на контрольно-пропускном пункте со шлагбаумом. ПИАНЗ должна ускорить и повысить надежность процесса пропуска автомобилей на частную территорию, облегчить работу сотрудника охраны.

Ключевые слова: приложение, авторизация, распознание, автомобили, номера, охрана, C#, emguCV.

Содержание

[Введение 4](#_Toc532161418)

[1 Анализ технических требований и уточнение спецификаций 5](#_Toc532161419)

[1.1 Анализ задания, выбор технологии, языка и среды разработки 5](#_Toc532161420)

[1.2 Модель жизненного цикла программы 5](#_Toc532161421)

[1.3 Разработка терминологии проекта 6](#_Toc532161422)

[1.4 Разработка диаграммы вариантов использования 7](#_Toc532161423)

[1.5 Объектная декомпозиция предметной области 10](#_Toc532161424)

[1.6 Разработка базы данных 12](#_Toc532161425)

[1.7 Разработка структурной схемы 12](#_Toc532161426)

[2 Проектирование структуры и компонентов программного продукта 14](#_Toc532161427)

[2.1 Основания выбора вида интерфейса 14](#_Toc532161428)

[2.2 Разработка графа состояний интерфейса 14](#_Toc532161429)

[2.3 Разработка пользовательского сценария 16](#_Toc532161430)

[2.4 Разработка форм интерфейса 18](#_Toc532161431)

[3 Тестирование приложения 26](#_Toc532161432)

[3.1 Тестирование граничных условий 26](#_Toc532161433)

[3.2 Тестирование методом эквивалентных разбиений 27](#_Toc532161434)

[3.3 Тестирование причинно-следственных связей 28](#_Toc532161435)

[Заключение 30](#_Toc532161436)

# Введение

Цель данной работы – разработка «Подсистемы идентификации автомобиля по номерному знаку на пропускном пункте» с использованием основных методов анализа, проектирования, реализации и тестирования программных систем.

Для достижения поставленной цели выделяются следующие задачи:

–анализ предметной области и выбор технических средств реализации;

–анализ технологий, применяемых для распознавания текста с изображения;

–реализация пользовательского интерфейса;

–реализация авторизации и базы данных;

–реализация модуля распознавания автомобильного номера с изображения;

Разработка ПИАНЗ является актуальной, так как в настоящее время все профессии, требующие монотонной длительной деятельности человека, постепенно заменяются автоматизированными системами, позволяющими выполнять эту работу компьютерам. Человек в таких профессиях остается лишь для контроля работы компьютера и его замены в случае отказа.

Основное назначение «Подсистемы идентификации автомобиля по номерному знаку на пропускном пункте» заключается в упрощении работы сотрудника охраны, находящегося на контрольно-пропускном пункте со шлагбаумом. Использование ПИАНЗ уменьшит необходимость нахождения сотрудника охраны на пропускном пункте и создаст перспективу полного отказа от сотрудников охраны на КПП.

«Подсистема идентификации автомобиля по номерному знаку на пропускном пункте» может быть легко модифицирована и улучшена, так как реализация интерфейсной и логической части находятся в разных модулях.

# Анализ технических требований и уточнение спецификаций

# Анализ задания, выбор технологии, языка и среды разработки

Согласно требованиям задания требуется разработать программный продукт, имеющий авторизацию, ограничивающую доступ пользователей, не имеющих доступа к ПИАНЗ, а также имеющий графический интерфейс пользователя, объединяющий в себе остальные несколько самостоятельных компонентов программы.

В связи с этим был выбран объектно-ориентированный подход, позволяющий использовать такие свойства, как наследование и инкапсуляция, с использованием которых легко реализовать интерфейс любой сложности.

При проектировании и реализации программного продукта будет использоваться нисходящий подход, в данном случае он является предпочтительным, так как в предметной области используется большое количество классов и их методов, на которых основан функционал всего приложения. Их взаимодействие будет легче отладить и тестировать при наличии визуального отображения интерфейса.

# Модель жизненного цикла программы

Для данной разработки была выбрана спиральная модель жизненного цикла (рисунок 2.2.1). В соответствии с данной схемой программное обеспечение создается не сразу, а итерационно с использованием метода прототипирования, базирующегося на создании прототипов.

Основным достоинством данной схемы является то, что, начиная с некоторой итерации, на которой обеспечена определенная функциональная полнота, продукт можно предоставлять пользователю, что позволяет сократить время до появления первых версий программного продукта и создаёт благоприятные условия для долгосрочной поддержки программного продукта.



Рисунок 1.2.1 – Спиральная модель жизненного цикла

Для реализации программного продукта был выбран язык C#, являющийся полноценным современным языком программирования, поддерживающим необходимые в данной разработке возможности. Разработка проводилась в среде «Visual Studio 2017». Использование быстрой библиотеки «emguCV» (computer vision), которая может работать без доступа в Интернет, позволило использовать возможности «компьютерного зрения», сохраняя при этом кроссплатформенность разрабатываемого продукта.

Хранение данных пользователей было реализовано с помощью стандартной для «Visual Studio» СУБД «Microsoft SQL Server».

# Разработка терминологии проекта

Для грамотной разработки и реализации проекта необходимо разработать терминологию. С помощью выбранных терминов будет описываться вся деятельность приложения во время его разработки, основной деятельности и поддержки.

Для реализации деятельности сотрудника охраны, достаточно ввести модель «Пользователь», имеющей доступ к основному функционалу ПИАНЗ.

Для корректной организации регистрации новых пользователей необходима модель «Администратор», которая помимо доступа к функционалу ПИАНЗ будет также иметь возможность подтверждать или отклонять новые регистрационные заявки.

На основании данных выводов был составлен глоссарий ПИАНЗ (Таблица 2.3.1).

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| Администратор | Сторона, организующая регистрацию новых пользователей, обеспечивающая подтверждение или отклонение заявок новых пользователей. |
| Заявка | Пользовательская заявка на регистрацию. |
| Пользователь | Сторона, получающая книги и несущая ответственность за их своевременное возвращение.  Синонимы: сотрудник КПП. |

Таблица 1.3.1 – Глоссарий программного продукта

# Разработка диаграммы вариантов использования

Разрабатываемая ПИАНЗ предусматриваем однопользовательскую архитектуру, но разграничивающую доступ пользователей с различными правами.

Предметная область предполагает процесс распознания автомобильного номера с изображения, при этом данный процесс включает в себя взаимодействие с файловой системой платформы и получение графического отображения выбранного для распознания изображения.

Доступ к процессу распознания ограничивается авторизацией, которая осуществляется одним из пунктов в главном меню. Также в меню находится возможность регистрации для неавторизованных пользователей. Для пользователя типа «Администратор» определены привилегированные права, позволяющие подтвердить или отклонить регистрацию пользователей из соответствующей вкладки в главном меню. Таким образом, выделяются два варианта использования:

-главное меню, предполагающее:

* регистрацию;
* авторизацию;
* подтверждение учетной записи;
* выход.

-распознание автомобильного номера, предполагающее:

* выбор изображения для распознания;
* отображения выбранного изображения;
* отображение распознанного номера в виде текста;
* отображение времени, потраченного на процесс распознания.

Таким образом, отобразим эти варианты использования на соответствующей диаграмме на рисунке 1.3.1.

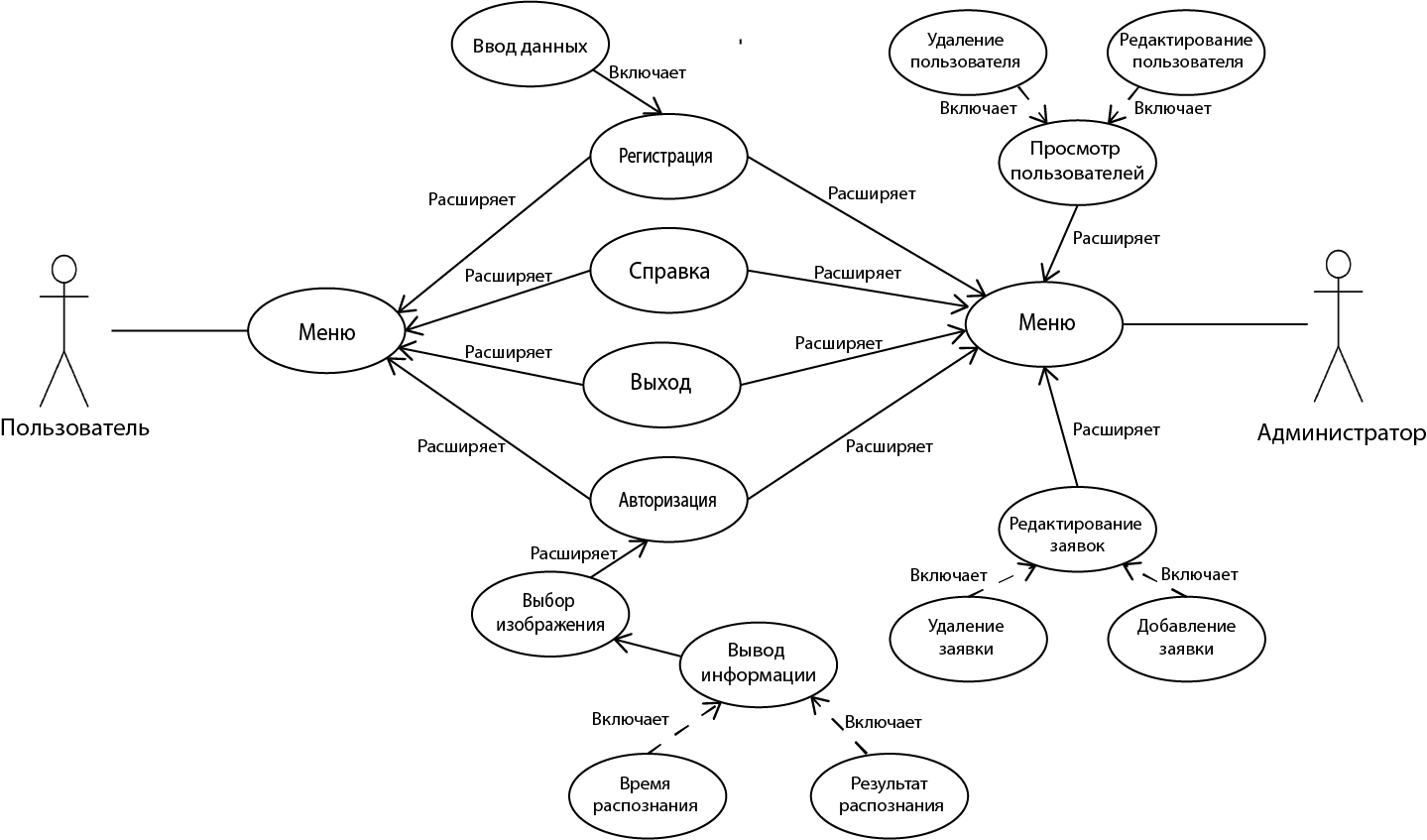


Рисунок 1.4.1 – Диаграмма вариантов использования

На диаграмме изображены семь основных вариантов использования – регистрация и авторизация пользователя, процесс распознания, просмотр пользователей, редактирование заявок, справка и выход.

Остальные варианты использования предполагают расширение основных вариантов. В их количество входят - ввод данных, выбор изображения, вывод времени распознания и результата распознания, удаление/добавление заявки и удаление пользователя.

Рассмотрим соответствующие варианты использования для уточнения принципов функционирования приложения:

# Объектная декомпозиция предметной области

Для реализации описанного приложения понадобится множество схожих между собой объектов, требуемых для реализации пользовательского интерфейса: кнопки, поля для ввода/вывода, поле для отображения изображения.

В связи с этим были составлены диаграммы классов, представленные на рисунках 2.6.1-2.6.2, которые также будут использованы в реализации ПИАНЗ.

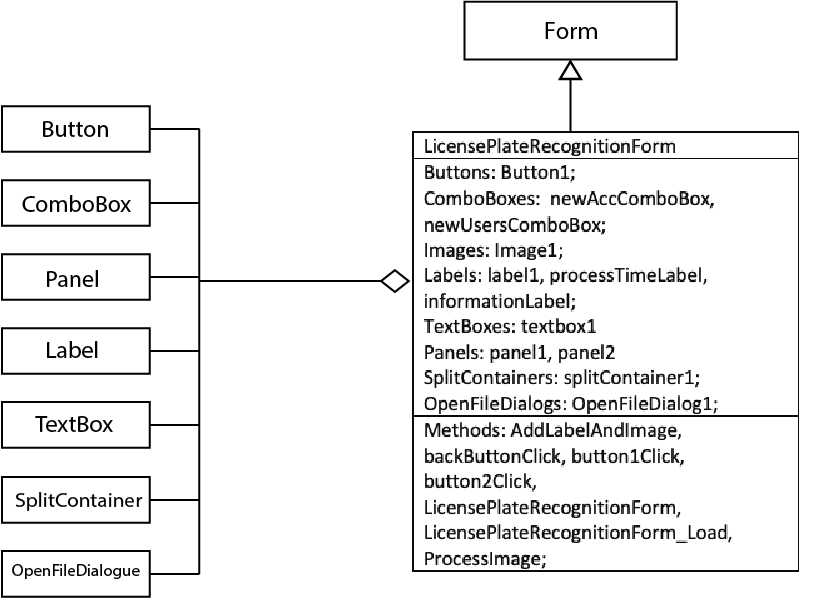


Рисунок 1.6.1 – Диаграмма классов формы «License Plate Recognition»

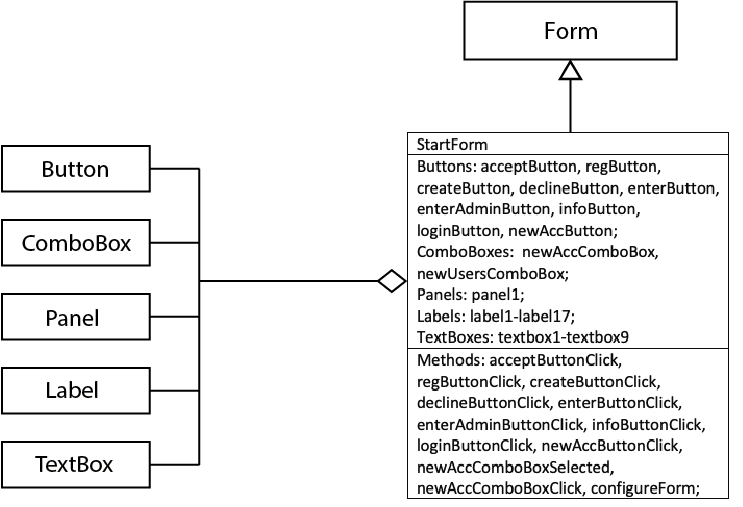


Рисунок 2.6.2 – Диаграмма классов формы «StartForm»

# Разработка базы данных

Для реализации авторизации необходимо где-то хранить данные пользователей, имеющий доступ к ПИАНЗ. В базе данных достаточно одной таблицы «Persons» (см рисунок 1.7.1), в которой будут следующие поля:

* id – уникальное идентификатор пользователя (первичный ключ);
* fullname – фамилия, имя, отчество пользователя;
* username – логин пользователя, использующийся для авторизации;
* password – пароль пользователя, использующийся для авторизации;
* email – адрес электронной почты.

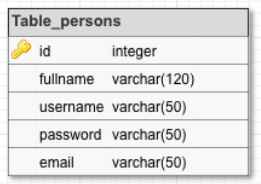


Рисунок 2.7.1 – Диаграмма базы данных

# Разработка структурной схемы

В проекте было выделено три области: область для хранения исходных изображений для распознания, область для хранения служебных файлов и изображений, а также область для хранения файлов-классов. Поскольку хранение всех файлов и изображений в одной директории ухудшает технологичность программного продукта, было решено разделить файлы по следующим папкам:

* Папка «\_pictures» - содержит изображения и файлы для реализации пользовательского интерфейса;
* Папка «\_numbers» - находится в папке «\_pictures» и содержит изображения автомобильных номеров, она открывается по умолчанию при выборе изображения в приложении;
* Папка «controllers» - содержит всей файлы, реализующие как логику программы (файлы с расширением «.cs»), так и её интерфейс (формы);
* Папка «tessdata» - является программно-сгенерированной с помощью библиотеки «emguCV» и содержит данные, используемые для распознания букв и цифр с изображения;

Структурная схема проекта приведена на рисунке 1.4.1

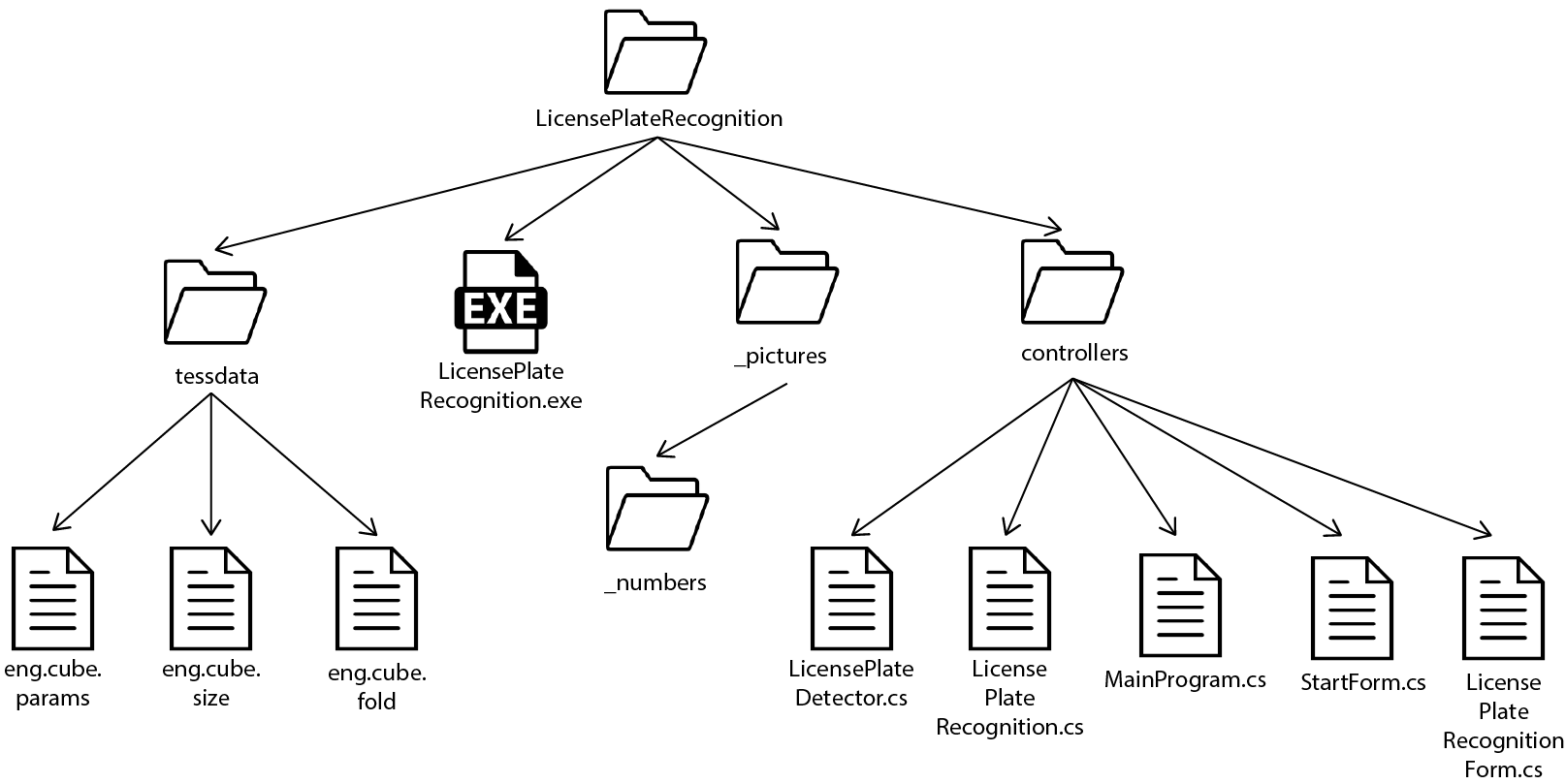


Рисунок 1.5.1 – Структурная схема проекта

# Проектирование структуры и компонентов программного продукта

# Основания выбора вида интерфейса

Интерфейс разрабатываемого программного продукта предназначен для узкого круга пользователей, имеющих доступ к ПИАНЗ, а так как в разрабатываемом приложении акцент делается на повторяющуюся операцию распознания номера с изображения, поэтому был выбран процедурно-ориентированный способ проектирования пользовательского интерфейса

В связи с тем, что при разработке программного продукта был выбран нисходящий способ проектирования, то первоочередной задачей ставится реализация пользовательского, а функционал компонентов временно замещается заглушками.

# Разработка графа состояний интерфейса

На основании предпроектных исследований и изучении работы ПИАНЗ было решено использовать интерфейс, соответствующий графу состояний интерфейса, представленному на рисунке 2.1.1.

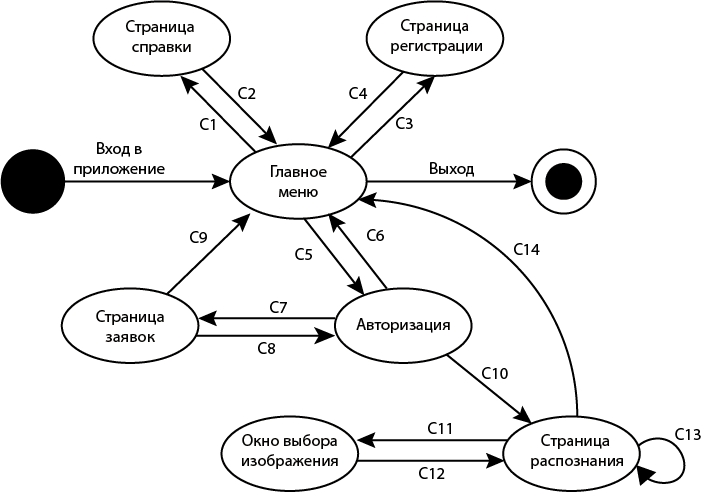


Рисунок 2.2.1 – Граф состояний интерфейса

Обозначения на графе состояний интерфейса:

С1 — нажатие кнопки «Справка»

С2 — возвращение в главное меню

С3 — нажатие кнопки «Регистрация»

С4 — Заполнение формы и возвращение в главное меню

С5 — нажатие кнопки «Авторизация»

С6 — Неуспешная авторизация и возвращение в главное меню

С7 — нажатие кнопки «Заявки» после авторизации

С8 — возвращение со страницы «Заявки» по окончанию редактирования

С9 — нажатие кнопки «Назад»

С10 — переход после успешной авторизации пользователя на страницу распознания

С11 — нажатие на «Загрузить»

С12 — нажатие кнопки «Выбрать» или «Отмена» или иконки «крестик»

С13 — повторный переход на страницу распознания после очередной итерации распознания изображения

С14 — нажатие иконки «крестик»

Детализуем диалог «Страница распознания» (рисунок 5), так как именно он является ключевым диалогом в данном. Данный диалог является управляемым системой, т.к. пользователю предоставляется всего 3 варианта действия: выбор изображения, вывод результата и выход.



Рисунок 3.1.2 – Граф диалога «Распознание»

# Разработка пользовательского сценария

Список методов, реализуемых на различных формах приведен на рисунке 2.6.2.

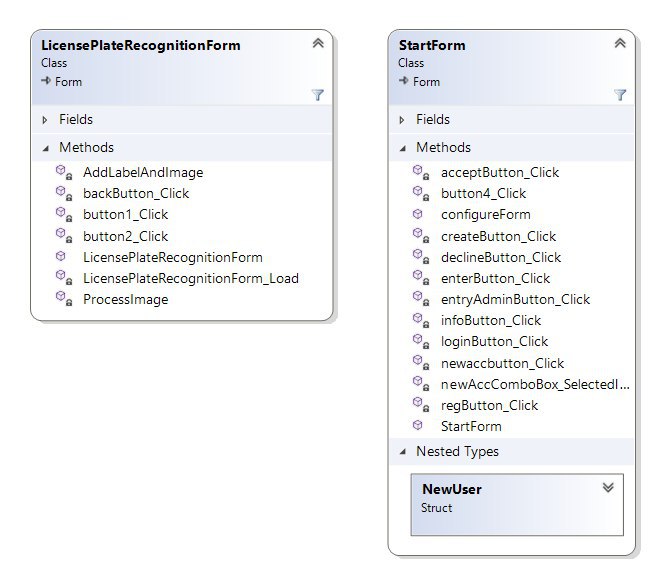


Рисунок 2.6.2 – Методы соответствующих классов

Таблица, отображающая поля структуры, необходимой для работы с базой данных приведена на рисунке 2.6.3

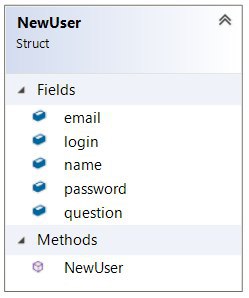
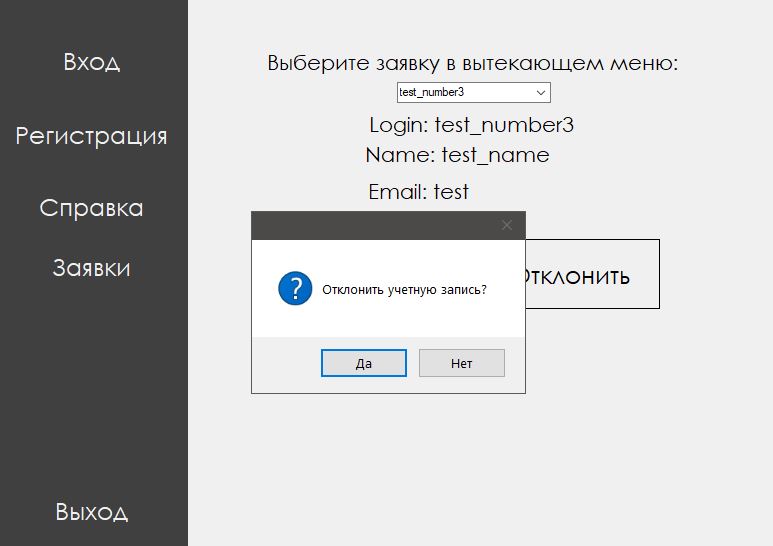
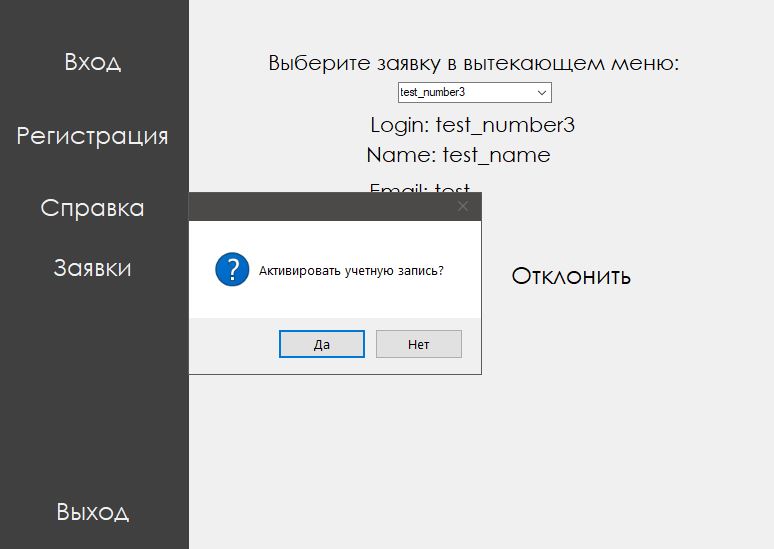
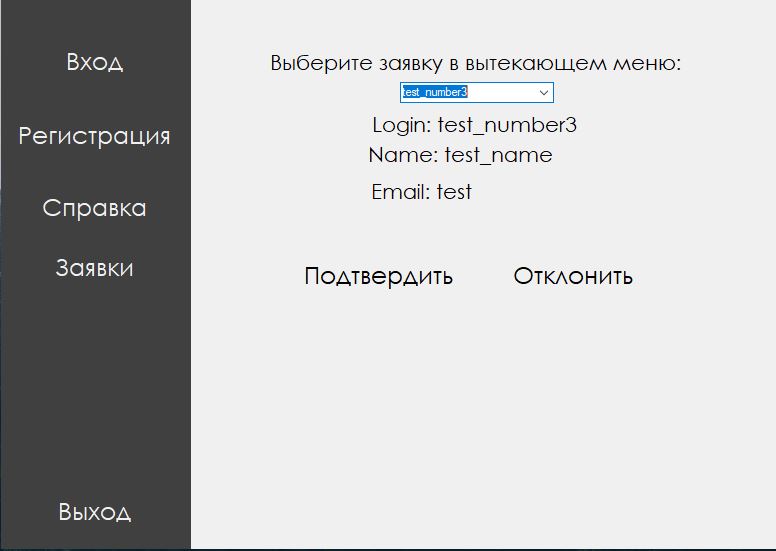
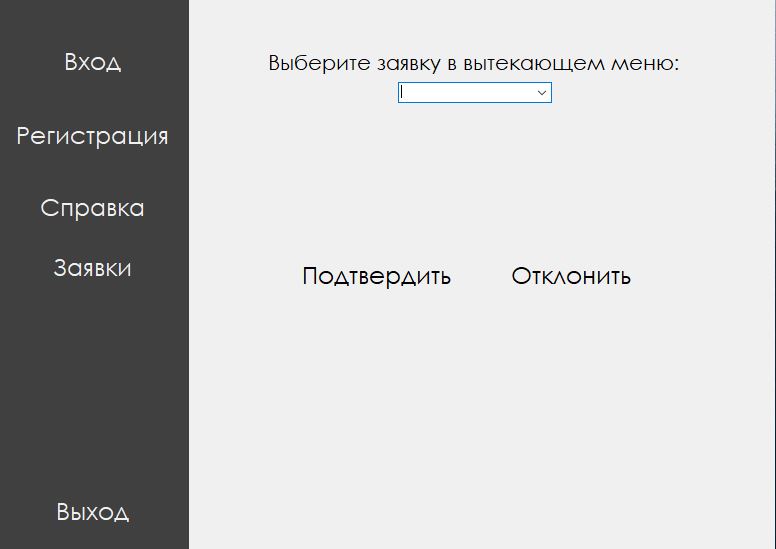
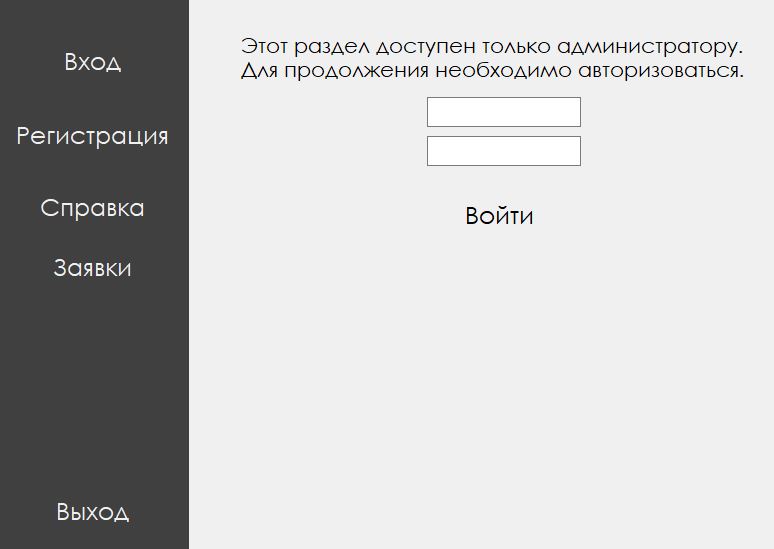
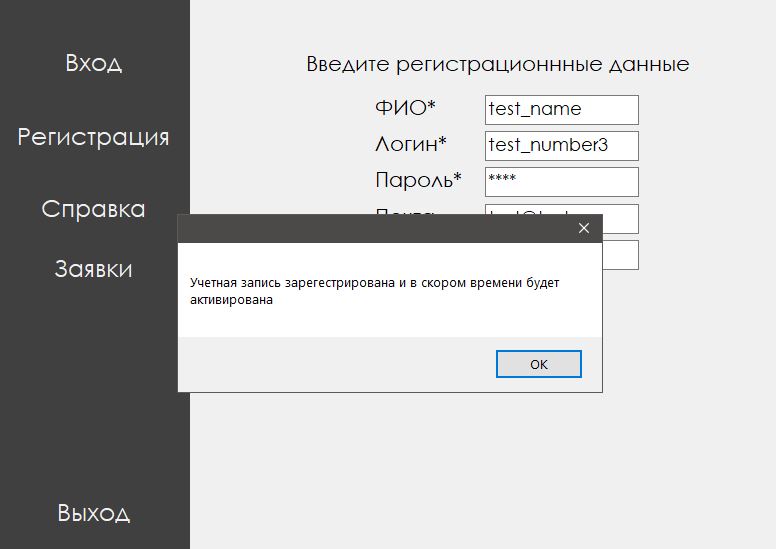
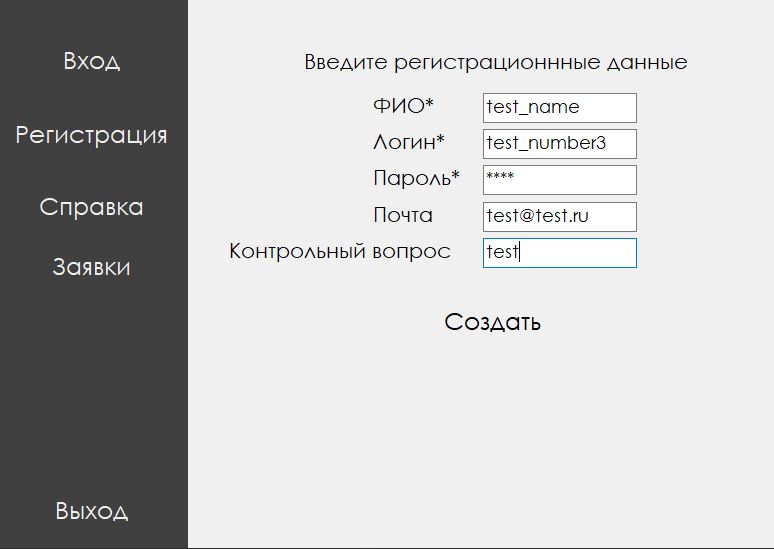
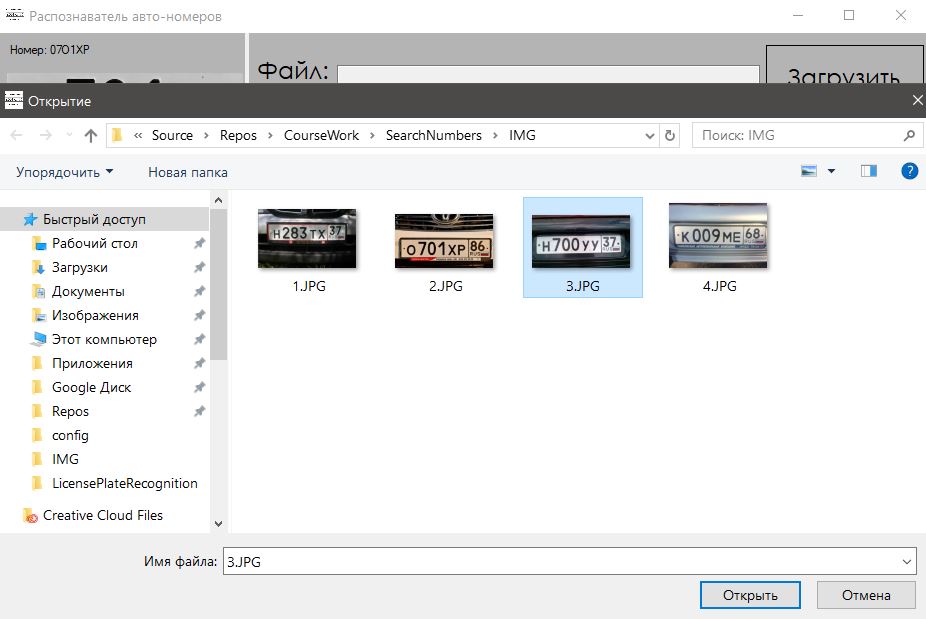
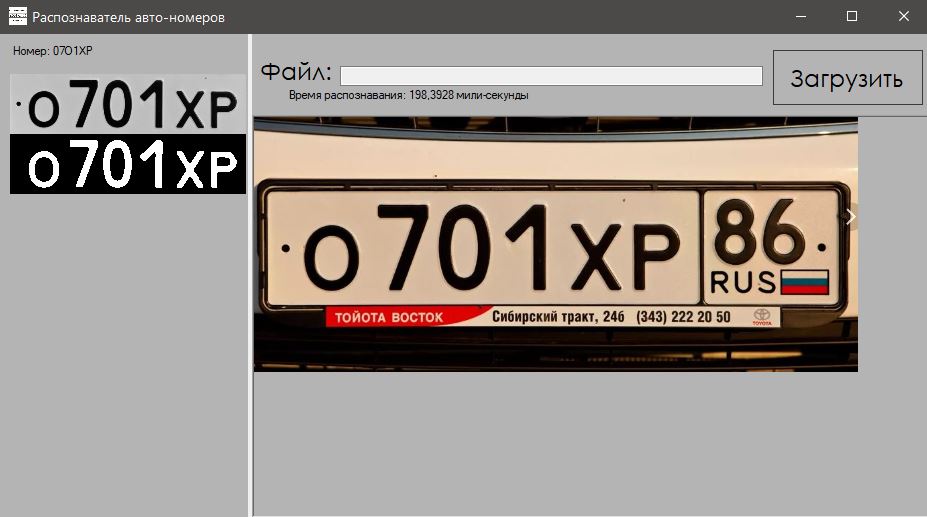
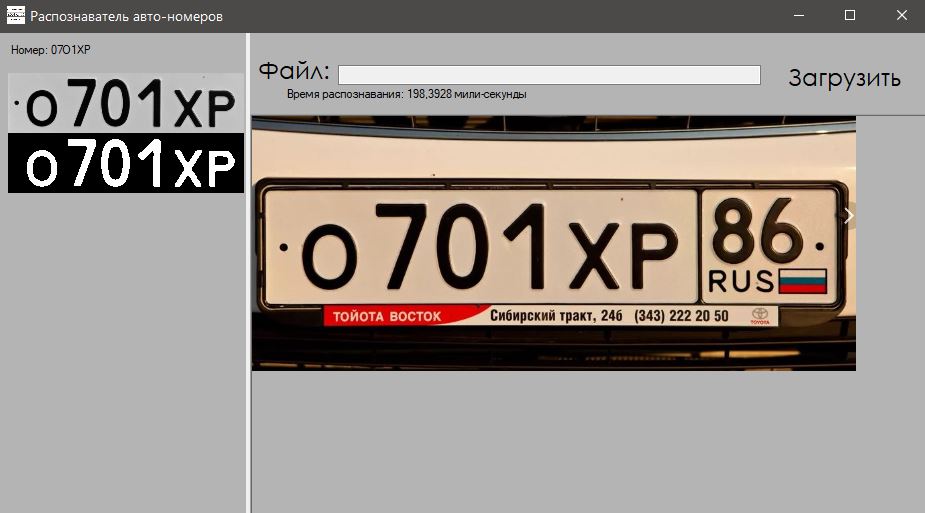
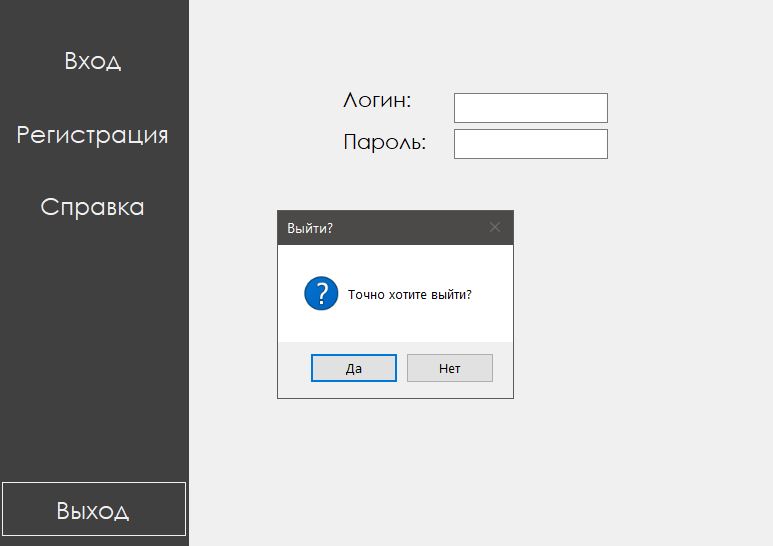
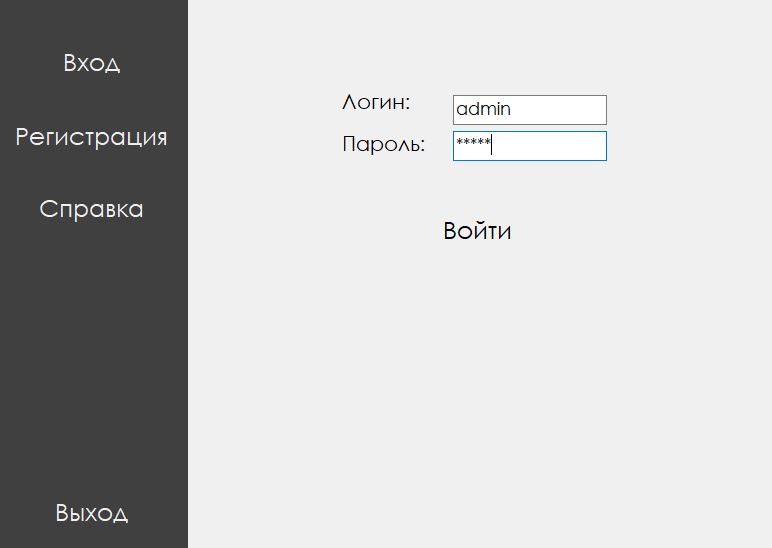
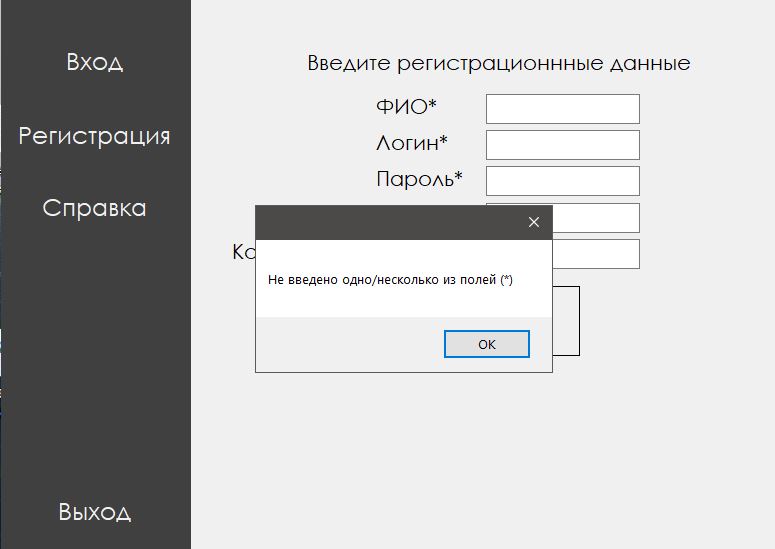
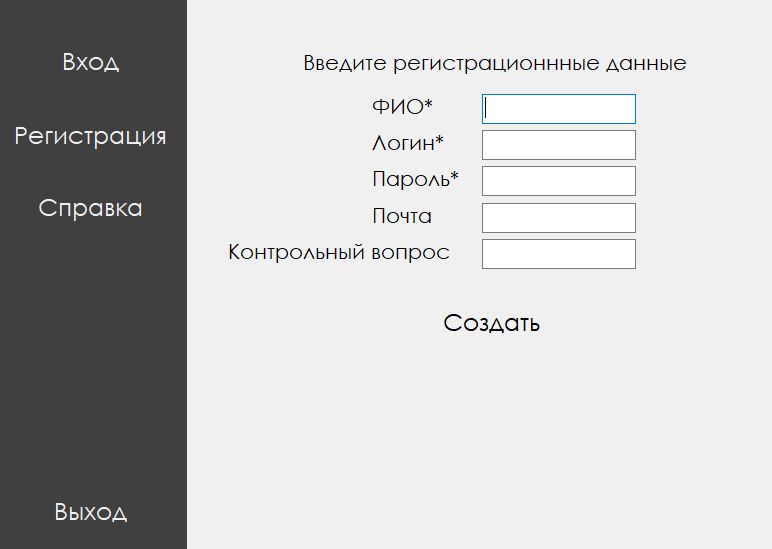
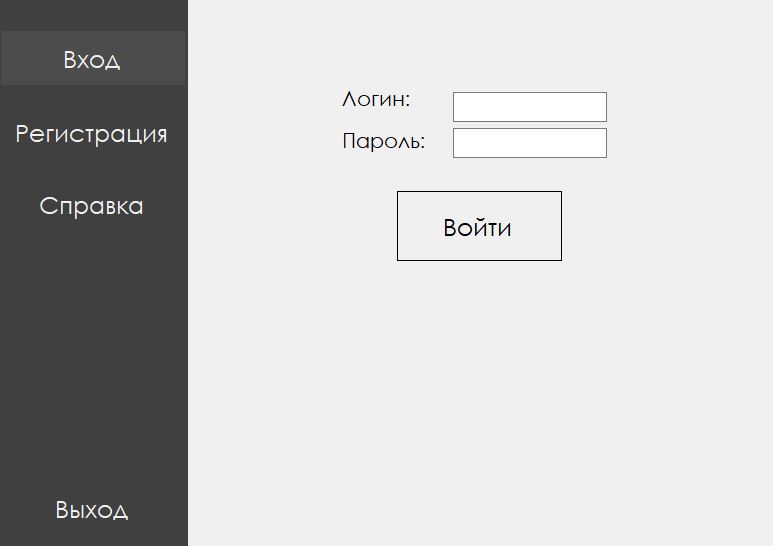


Рисунок 2.6.2 – Схема структуры

# Разработка форм интерфейса



# Тестирование приложения

Опишем проведенное тестирование по принципу “Черного ящика”, рассмотрев некоторые виды функционального тестирования.

# Тестирование граничных условий

Ввод данных в приложении осуществляется только для добавляемых или редактируемых книг и пользователей.

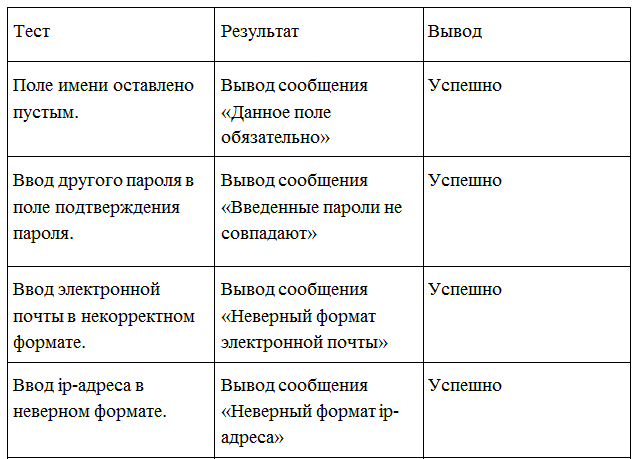
Проведем тестирование граничных условий для этих моделей

Таблица 2 - Тестирование граничных условий

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Ожидаемый результат** | **Результат** | **Вывод** |
| Ввод в поля имени, фамилии, номера документа пустых значений | Уведомление о некорректном вводе (пустые поля) | Уведомление о некорректном вводе (пустые поля) | Успешно |
| Ввод в поля имени, фамилии, отчества значений длинной больше 128 | Уведомление о некорректном вводе (превышен максимальный размер поля) | Уведомление о некорректном вводе (превышен максимальный размер поля) | Успешно |
| Ввод в поля названия, автора, издательства, количества страниц пустых значений | Уведомление о некорректном вводе (пустые поля) | Уведомление о некорректном вводе (пустые поля) | Успешно |
| Ввод в поля названия, имени, издательства значений длинной больше 128 | Уведомление о некорректном вводе (превышен максимальный размер поля) | Уведомление о некорректном вводе (превышен максимальный размер поля) | Успешно |
| Ввод в поле пароля при регистрации значение длинной 5 символов | Уведомление о некорректном вводе (меньше минимально возможного) | Уведомление о некорректном вводе (меньше минимально возможного) | Успешно |

# Тестирование методом эквивалентных разбиений

Определим классы эквивалентности. Выделим минимальную длину вводимых данных для всех текстовых полей. Получим 2 класса эквивалентности: верный – более 1 символа, и неверный - ни одного символа. В случае, если не введено ни одного символа, будет выведено сообщение об ошибке. Отдельное ограничение на вводимые символы для поля пароля – минимальная длина пароля – 6 символов. Помимо этого введем проверку формата электронной почты пользователя. Также производится проверка правильности формата введенного ip-адреса.



# Тестирование причинно-следственных связей

В данном приложении большую часть занимает работа с уже существующими данными без ввода дополнительных значений. Необходимо провести анализ причинно-следственных связей.

Таблица 3 - Анализ причинно-следственных связей

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест** | **Ожидаемый результат** | **Результат** | **Вывод** |
| Ввод при регистрации занятого email | Сообщение о занятости email | Сообщение о занятости email | Успешно |
| Попытка добавить книгу, находящуюся в заказе | Кнопка добавления в заказ заблокирована.  Отображение в списке книг красным | Кнопка добавления в заказ заблокирована. Отображение в списке книг красным | Успешно |
| Открытие списка текущих заказов при их отсутствии | Сообщение об отсутствии текущих заказов | Сообщение об отсутствии текущих заказов | Успешно |
| Открытие списка текущих карт при их отсутствии | Сообщение об отсутствии текущих карт | Сообщение об отсутствии текущих карт | Успешно |
| Открытие страницы книги без изображения | Страница информации о книге без изображения | Страница информации о книге без изображения | Успешно |
| Наличие заказа со сроком сдачи через 3 дня | Письмо с сообщением о дате возвращения книг | Письмо с сообщением о дате возвращения книг | Успешно |
| Наличие просроченного заказа для сдачи | Письмо каждые три дня о наличие долга | Письмо каждые три дня о наличие долга | Успешно |
| Заказ просрочен, но закрыт | Отображение в истории желтым | Отображение в истории желтым | Успешно |
| Заказ закрыт вовремя | Отображение в истории зеленым | Отображение в истории зеленым | Успешно |

# Заключение

В итоге данной разработки получен прототип программного решения поставленной задачи, соответствующий всем требованиям технического задания. В приложение и его программную реализацию включена возможность расширения функционала.

Можно сделать вывод, что приложение получилось технологичным, так как была достаточно хорошо проработана модель программы, подзадачи и структуры данных были четко определены. Программа написана в хорошем стиле, код хорошо читаем. Везде, где возможно, присутствует повторное использование кода.

В результате разработки получено Web-приложение управления работой библиотеки. Приложение предназначено для упрощения процесса получения читателями литературы и для централизации учета выданных библиотекой книг. Наиболее эффективной данная разработка может оказаться в школьных и районных городских библиотеках.

Разработанный программный продукт организует процессы выдачи и возвращения читателями книг, хранение информации по этим процессам, а также реализует уведомление читателей по их текущим заказам. Приложение обладает проработанным пользовательским интерфейсом, что расширяет круг пользователей программы. Программа написана в хорошем стиле, код хорошо читаем. Везде, где возможно, присутствует повторное использование кода. Также, разработанные модули обладают достаточной независимостью.

В результате выполнения работы был создан программный продукт «Клиентская часть приложения для контроля параметров удаленного сервера». Данное приложение используется для отображения статистических показателей удаленного сервера, таких как: память, нагрузка на сеть, состояние процессора.

Приложение реализовано на языке Ruby с использованием фрэймворка Rails и библиотеки Bootstrap.

Заявленная функциональность реализована в полной мере в соответствии с требованиями в задании.

В процессе реализации программы были выполнены:

— анализ предметной области;

— определение вариантов использования программы;

— разработка интерфейса пользователя;

— разработка компонентов программного продукта;

— тестирование программы.

Разработанную программу можно считать технологичной, так как она в достаточной мере соответствует всем четырем признакам технологичности: проработанности моделей, стилю программирования, уровню независимости модулей, и степени повторного использованию кода.

Модели программы можно считать проработанными, так как еще на этапе проектирования были выделены основные задачи, реализованные в приложении впоследствии.

Программа написана в соответствии со стилями руководства используемых языков. В коде программы присутствуют необходимые комментарии, имена переменных, функций и классов осмысленны, присутствуют отступы и пустые строки для облегчения чтения программы.

Степень независимости модулей довольно высока, так как все основные сущности программы вынесены в отдельные классы, также программа разбита на 3 крупные части (клиентскую и серверную части и базу данных)

Модули программы выделены таким образом, что они могут использоваться для других задач или являться общими для различных объектов. Также использованы шаблоны проектирования.