**Содержание**

[Введение 4](#_Toc126406206)

[1. Общая часть 5](#_Toc126406207)

[1.1.Классификация программных средств 5](#_Toc126406208)

[1.2Жизненный цикл прикладной программы 7](#_Toc126406209)

[1.3Методология и технология разработки ПП 9](#_Toc126406210)

[1.4Тестирование программных средств 11](#_Toc126406211)

[2. Специальная часть 13](#_Toc126406212)

[2.1.Описание программы 13](#_Toc126406213)

[2.2.UML-диаграмма 14](#_Toc126406214)

[2.3.Регистрация 15](#_Toc126406215)

# Введение

В настоящее время компьютеры широко применяются в самых разных областях: науке, промышленности, медицине, образовании и т.д. Разработка данного приложения позволит автоматизировать деятельность и работу «Автошколы».

Целью курсового проекта является закрепление теоретических знаний и совершенствование практических навыков программирования.

В ходе выполнения курсового проекта необходимо решить следующие задачи:

1. Разработать алгоритм поставленной задачи.

2. Реализовывать алгоритм средствами автоматизированного проектирования.

3. Разработать код программного продукта на основе готовой спецификации на уровне модуля с использованием современных языков программирования.

4. Провести отладку программного продукта с использованием инструментальных средств.

5. Провести тестирование программного модуля по определенному сценарию.

6. Оформить документацию на программные средства с использованием средств для автоматизации оформления документации.

Объектом исследования данного курсового проекта является автошкола. С помощью приложения планируется упростить и ускорить работу автошколы, в частности: выдавать нужную информацию пользователю.

# Общая часть

# Классификация программных средств

Программное обеспечение – это набор команд, управляющих работой компьютера. Без программного обеспечения компьютер не сможет выполнять задачи, которые мы обычно связываем с компьютерами. Функции программного обеспечения следующие:

управлять компьютерными ресурсами организации;

обеспечивать пользователя всеми инструментами, необходимыми для извлечения пользы из этих ресурсов;

выполнять роль посредника между организациями и хранимой информацией.

Классификация ПО

По степени переносимости программы делят на:

* платформозависимые;
* кроссплатформенные.

По способу распространения и использования программы делят на:

* несвободные (закрытые);
* открытые;
* свободные.

По назначению программы делят на:

* системные;
* прикладные;
* инструментальные.

По видам программы делят на:

компонент: программа, рассматриваемая как единое целое, выполняющая законченную функцию и применяемая самостоятельно или в составе комплекса;

программа, состоящая из двух или более компонентов и (или) комплексов, выполняющих взаимосвязанные функции, и применяемая самостоятельно или в составе другого комплекса.

     Уровни ПО:

         1. Базовое ПО – базовый уровень

         2. Системное ПО – системный уровень

         3. Служебное (сервисное) ПО

         4. Прикладное ПО

Каждый вышележащий уровень повышает функциональность всей системы.

  Базовое ПО – это минимальный набор программных средств, обеспечивающих работу компьютера. В базовое ПО входят операционная система, операционные оболочки (текстовые и графические), сетевая операционная система.

 Системное ПО — комплекс программ, которые обеспечивают управление компонентами компьютерной системы, такими как процессор, оперативная память, устройства ввода-вывода, сетевое оборудование, выступая как «межслойный интерфейс», с одной стороны которого аппаратура, а с другой — приложения пользователя.

Служебное (сервисное) ПО- это программы и программные комплексы, которые расширяют возможности базового ПО и организуют более удобную среду работы пользователя.

Прикладное ПО– [программа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), предназначенная для выполнения определённых задач и рассчитанная на непосредственное взаимодействие с пользователем.

Инструментальное ПО–программное обеспечение, предназначенное для использования в ходе [проектирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F), [разработки](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) и сопровождения программ, в отличие от [прикладного](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) и [системного](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) программного обеспечения

# Жизненный цикл прикладной программы

Жизненный цикл программного обеспечения (ПО) — период времени, который начинается с момента принятия решения о необходимости создания программного продукта и заканчивается в момент его полного изъятия из эксплуатации. Этот цикл — процесс построения и развития ПО.

Этапы жизненного цикла:

1. Анализ.
2. Проектирование.
3. Реализация.
4. Сборка, тестирование, испытание.
5. Внедрение.
6. Сопровождение.

Различают 2 случая производства ПО: 1) ПО делается для конкретного заказчика. В этом случае нужно прикладную задачу превращать в программистскую. Нужно понять, как функционирует та среда, которую нужно автоматизировать (анализ бизнес-процессов). В результате появляется документация-спецификация требования, где указаны, какие именно задачи решены и при каких условиях. Эту работу выполняет системный аналитик (аналитик бизнес-процессов). 2) ПО разрабатывается для рынка. Нужно проводить маркетинговые исследования и найти какого продукта на рынке нет. Это связано с большим риском. Цель – разработка спецификации требований.

Цель проектирования - определение общей структуры (архитектуры) ПО. Результат – спецификация ПО. Эту работу выполняет системный программист.

Реализация включает и разработку, и тестирование, и документацию, а также написание программного кода.

На четвертом этапе происходит сборка всего, что сделано разными программистами. Тестирование всего программного комплекса. Отладка – поиск и устранение причин ошибок. Испытание – уточнение технических характеристик. В результате – гарантия работоспособности программы.

Внедрение – когда работают на одного заказчика. Включает постановку программы у заказчика, обучение заказчика, консультации, устранение ошибок и явных недостатков. Должно произойти отчуждение ПО – пользователь может работать с ПО без участия автора.

Проектирование

Разработка требований

Реализация

Тестирование

Эксплуатация и поддержка

Рисунок 2 – Схема жизненного цикла ПО

Выпуск – когда ПО разрабатывается на рынок. Начинается с этапа бета-тестирования. Альфа-тестирование – тестирование людьми из той же организации, не участвовавших в разработке программ. Бета-тестирование – изготовление нескольких экземпляров ПО и отправка потенциальным заказчикам. Цель – еще раз проверить разработку ПО.

На последнем этапе происходит Устранение замеченных в ходе эксплуатации ошибок. Внесение не принципиальных усовершенствований. Накопление предложений для разработки следующей версии.

# Методология и технология разработки ПП

Под технологией разработки программного обеспечения (ТРПО) понимается совокупность обобщенных и систематизированных знаний, или наука об оптимальных способах (приемах) проведения процесса разработки программного обеспечения, обеспечивающего в заданных условиях получение программной продукции с заданными свойствами.

Технология разработки программного обеспечения представляет собой инженерный подход к разработке программных средств ЭВМ, охватывающий методологию программирования, проблемы обеспечения надежности программ, оценки рабочих характеристик и качества проектов.

Технология разработки программного обеспечения рассматривает вопросы управления проектированием систем программного обеспечения, а также средства и стандарты разработки программ.

Технология разработки программного обеспечения определяет некоторую профессиональную культуру работы специалистов (не только программистов), обеспечивающую заданный уровень производительности труда и качества, получаемой в результате программной продукции.

Технология разработки программного обеспечения охватывает процесс разработки программного обеспечения от появления потребности в нем до его изготовления, передачи пользователю, модификации в процессе эксплуатации и прекращения его использования вследствие морального старения.

В идеале технология разработки программного обеспечения должна удовлетворять основным нижеперечисленным требованиям:

1. Необходима стандартизация языков проектирования программ, оформления и испытания программных модулей, а также гарантии их качества. Это позволит значительно сократить дублирующие разработки, внедрить сборочное программирование и вести накопление на предприятиях и в стране высококачественного программного продукта для его многократного использования в качестве типовых комплектующих изделий.
2. Вести постоянный контроль и обеспечение качества программ.
3. Программы не должны содержать непроверенных путей и ситуаций функционирования, которые приводят к неожиданным результатам.
4. Пользователю или покупателю программ необходимо дать четкое представление о возможностях данной программы и технологических условиях эксплуатации, при которых гарантируются определенные функции и качества.
5. Технология разработки программного обеспечения должна обеспечивать отторжимость программного изделия от его разработчика, т.е. человеческий фактор в программировании должен быть сведен к минимуму.
6. Технология разработки программного обеспечения и средства ее поддержки (автоматизации) должны обеспечивать целенаправленную работу, прежде всего коллектива программистов, а не отдельных личностей. Она должна побуждать коллектив работать только правильно и слаженно; должна автоматически блокировать любые не санкционированные технологией действия.
7. Необходимо вести аккуратное документирование всех этапов разработки. Документация должна также заноситься и храниться на магнитных носителях. Доступ к этой информации должен быть открытым, простым и автоматизированным.
8. Работа пользователя должна обеспечиваться развитой информационно-справочной системой.
9. Средства автоматизации технологии должны охватывать все этапы работы коллектива программистов.
10. Технология разработки программного обеспечения должна быть простой в освоении, с автоматически включаемыми средствами подсказки.
11. Технология разработки программного обеспечения должна иметь средства автоматической фиксации в хронологическом порядке всех действий, выполняемых в процессе коллективного изготовления программного изделия - должны вестись и храниться в системе журналы (протоколы, дневники) разработки. Эти средства должны позволять восстанавливать любое состояние процесса разработки на любом интервале изготовления программной эксплуатации.

# Тестирование программных средств

Отладка ПС — это деятельность, направленная на обнаружение и исправление ошибок в ПС с использованием процессов выполнения его программ.

Тестирование ПС — это процесс выполнения его программ на некотором наборе данных, для которого заранее известен результат применения или известны правила поведения этих программ. Указанный набор данных называется тестовым или просто тестом. Таким образом, отладку можно представить в виде многократного повторения трех процессов: тестирования, в результате которого может быть констатировано наличие в ПС ошибки, поиска места ошибки в программах и документации ПС и редактирования программ и документации с целью устранения обнаруженной ошибки. Другими словами,

Отладка = Тестирование + Поиск ошибок + Редактирование.

Успех отладки ПС в значительной степени предопределяет рациональная организация тестирования. При отладке ПС отыскиваются и устраняются, в основном, т.е. ошибки, наличие которых в ПС устанавливается при тестировании. Как было уже отмечено, тестирование не может доказать правильность ПС, в лучшем случае оно может продемонстрировать наличие в нем ошибки. Другими словами, нельзя гарантировать, что тестированием ПС практически выполнимым набором тестов можно установить наличие каждой имеющейся в ПС ошибки. Поэтому возникает две задачи.

Первая задача: подготовить такой набор тестов и применить к ним ПС, чтобы обнаружить в нем по возможности большее число ошибок. Однако, чем дольше продолжается процесс тестирования (и отладки в целом), тем большей становится стоимость ПС.

Вторая задача: определить момент окончания отладки ПС (или отдельной его компоненты). Признаком возможности окончания отладки является полнота охвата пропущенными через ПС тестами (т.е. тестами, к которым применено ПС) множества различных ситуаций, возникающих при выполнении программ ПС, и относительно редкое проявление ошибок в ПС на последнем отрезке процесса тестирования. Последнее определяется в соответствии с требуемой степенью надежности ПС, указанной в спецификации его качества.

# Специальная часть

# Описание программы

При запуске приложения «Ресторан» перед пользователем открывается окно-приветствие с названием проекта и автором работы. Переход на следующую форму осуществлен с помощью таймера с ожиданием в пять секунд.

На второй форме пользователь видит номер горячей линии поликлиники, а также стоит перед выбором – регистрация или же авторизация.

При нажатии кнопки «Регистрация» перед пользователем появляется окно с пустыми полями. Для регистрации пользователь должен ввести следующие данные:

* логин;
* пароль;
* повтор пароля.

Поля с паролями скрыты маской ввода, при этом реализована функция их просмотра. Обязательным условием регистрации является совпадение двух полей с паролем.

При вводе пароля пользователь может наблюдать за тем, насколько сложный пароль он придумал – об этом будет выводиться сообщение в этом же окне.

После регистрации система сообщает об этом пользователю, а данные регистрации заносятся в базу данных. Затем пользователя переносят на форму регистрации-авторизации. Теперь пользователь способен зайти в свой личный кабинет. Для этого он нажимает кнопку «Вход» и попадает на страницу авторизации.

Для входа в систему пользователю требуется ввести логин и пароль. В случае, при вводе не верного пароля в первый раз выходит сообщении об ошибке в пароле, когда пользователь 2 раза ввел неверный пароль или логин, то появляется «captcha» - набор символов, который пользователь должен «разгадать» и ввести в текстовое поле, если же в третий раз пароль введен не верно, система блокирует вход на 10 секунд.

После входа в систему, появляется окно «Меню пользователя» - именно тут реализован весь функционал программы. На данной форме находятся несколько кнопок, при переходе по которым пользователь сможет узнать нужную ему информацию или же осуществить какие-либо действия.

При входе под администратором, администратор может удалять функционал пользователя, после удаления может вернуть все в первоначальном виде.

# UML-диаграмма

UML, или Unified Modeling Language, — унифицированный язык моделирования. Это графический язык, который с помощью диаграмм и схем описывает разнообразные процессы и структуры. Это не язык программирования, но чаще всего UML применяют именно в IT — с его помощью можно автоматически генерировать код.

Язык UML предназначен прежде всего для разработки программных систем. Его использование особенно эффективно в следующих областях:

* информационные системы масштаба предприятия;
* банковские и финансовые услуги;
* телекоммуникации;
* транспорт;
* оборонная промышленность, авиация и космонавтика;
* розничная торговля;
* медицинская электроника;
* наука;
* распределенные Web-системы.

Для ресторана, в котором есть администратор, гость и официант UML-диаграмма будет выглядеть следующим образом:

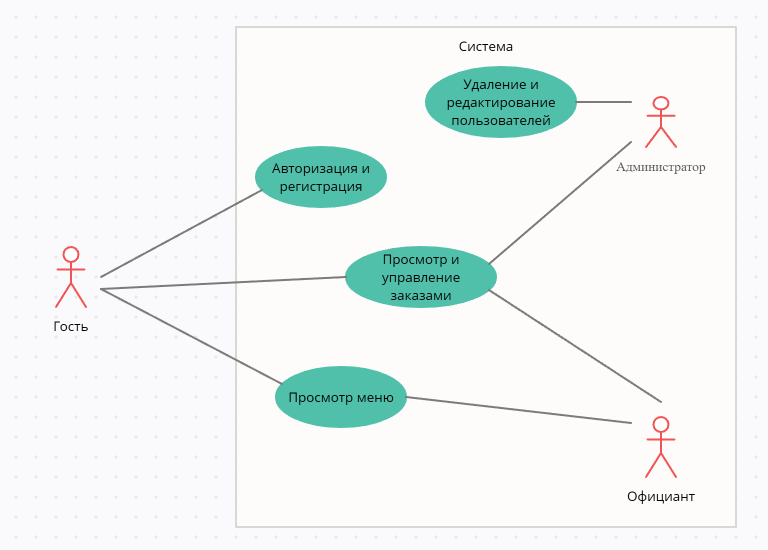


Рисунок 1- UML-диаграмма ресторана

Пользовательские прецеденты для каждого типа пользователя следующие:

Администратор

* управление правами доступа;
* удаление пользователей.

Гость

* авторизация (если ранее уже был зарегистрирован) и регистрация;
* просмотр меню;
* оформление заказа.

Официант

* авторизация;
* просмотр заказов.
  1. **Регистрация**

Регистрация пользователя - процесс создания нового аккаунта для пользователя в системе, который позволяет получить доступ к функциональности системы и сохранить ранее введенные данные. Обычно при регистрации пользователь указывает свои данные (имя, электронная почта, пароль и т.д.), которые должны быть уникальными, чтобы система могла идентифицировать пользователя. После успешной регистрации пользователь может войти в систему, используя свой логин и пароль, и получить доступ к функциональности системы в соответствии с его уровнем доступа. Регистрация пользователя часто является необходимой процедурой для использования различных онлайн-сервисов и приложений, а также для сохранения персональных настроек и данных пользователя.

В данном приложении зарегистрировать пользователя можно перейдя с окна авторизации по кнопке «Зарегистрироваться» на окно регистрации.

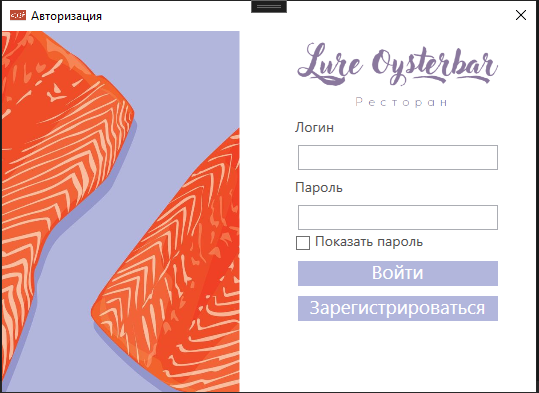


Рисунок 2-Окно авторизации

Пользователю необходимо заполнить форму регистрации, указав необходимые данные, такие как логин и пароль. Подтвердить регистрацию, введя повторив пароль. Если все данные указаны верно и регистрация прошла успешно, то пользователь будет перенаправлен на страницу пользователя.

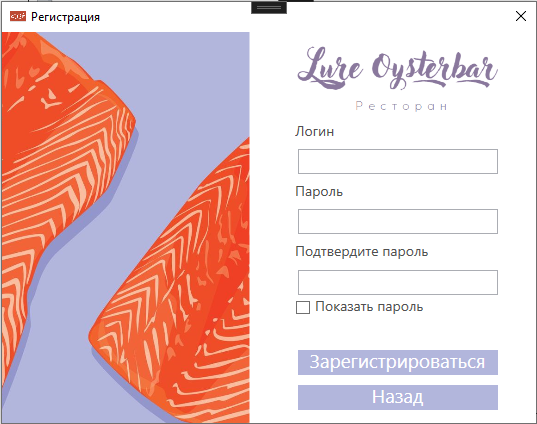


Рисунок 3-Окно регистрации

Если какое-то поле не будет заполнено, то система сообщит пользователю что необходимо заполнить определенные поля.

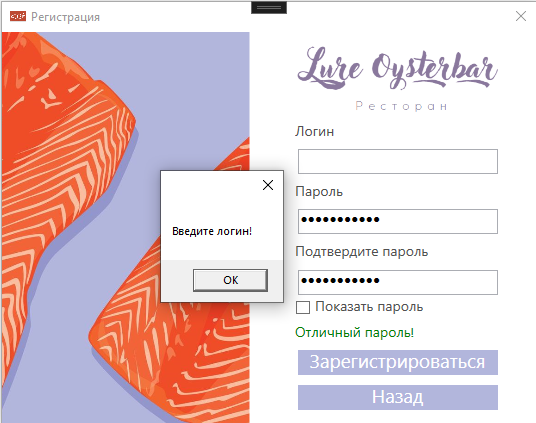


Рисунок 4-Сообщении об отсутствии логина

После этого пользователь сможет войти в приложение, используя свой логин и пароль, и начать пользоваться его функциональностью.

Так же в приложении представлена система проверки пароля на соответствие требованиям безопасности системы. Проверка пароля включает в себя анализ длины, сложности и уникальности пароля. Если введенный пароль соответствует требованиям безопасности системы, то пользователь получает доступ к функциональности системы. Если пароль не проходит проверку, то пользователю может быть предложено изменить свой пароль на более сложный и безопасный. Проверка пароля является важным шагом в защите учетных записей пользователей и предотвращении несанкционированного доступа к конфиденциальным данным. В данном приложении проверяется пароль на следующие требования:

* имеет ли пароль верхний и нижний регистр букв;
* длинна пароля должна быть не менее 8 и не более 16 символов;
* пароль должен иметь специальные символы, такие как: @#$%.

Если пароль не соответствует этим требованиям, то пользователю выводится оповещение о присутствии ошибок в веденном пароле.

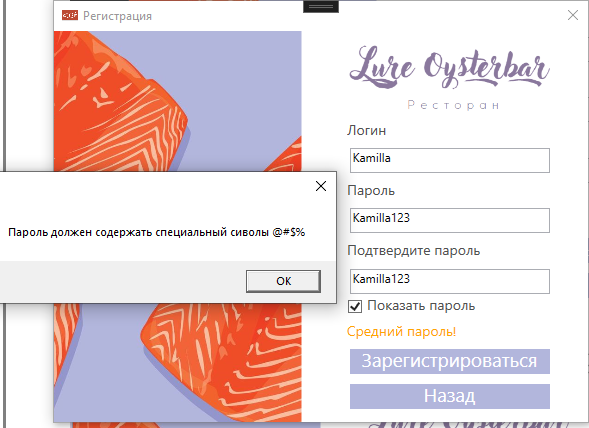


Рисунок 5-Сообщение о несоответствии требованиям пароля

Так же введенный пароль проверяется на совпадение паролей.

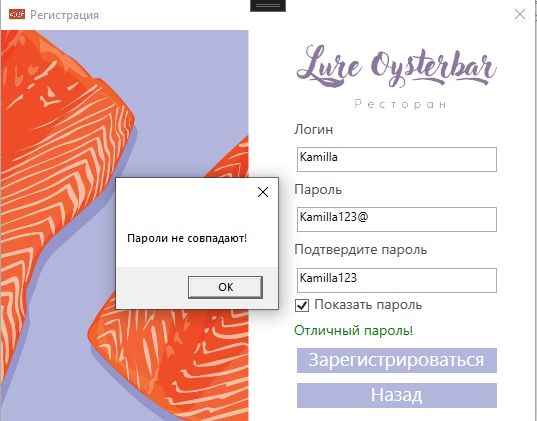


Рисунок 6-Сообщении о несовпадении паролей

В приложении имеется индикатор, показывающий пользователю качество надежности пароля, если пароль соответствует хотя бы одному требованию, то он будет считаться «плохим паролем» и индикатор будет подсвечиваться красным цветом, если есть два соответствия требованиям, то индикатор сообщит пользователю что это «средний пароль» и окрасится в оранжевы цвет, если пароль удовлетворяет всем требованиям, то это «отличный пароль» и окрасится в зеленый цвет.

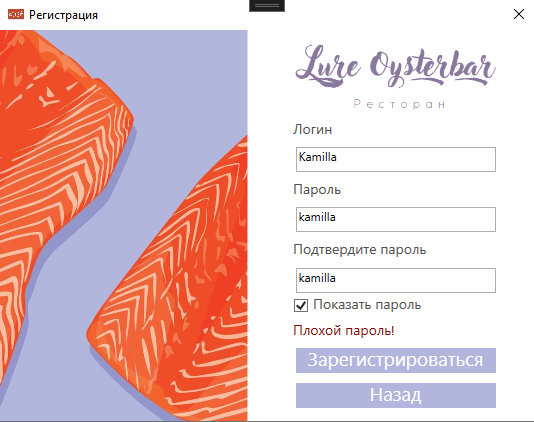


Рисунок 7-Индикатор плохого пароля

**2.4 Руководство пользователя**

При запуске приложения пользователя встречает приветственное окно, в нем содержится информация о названии программы и авторе.

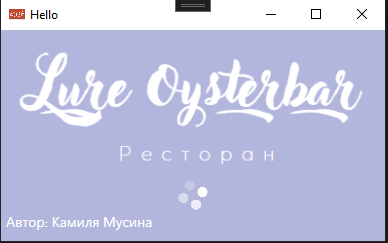


Рисунок 8-Приветвенное окно

После окончания загрузки программы пользователь перенаправляется на окно авторизации, где пользователь, раннее уже зарегистрированный, может авторизоваться введя логин и пароль.

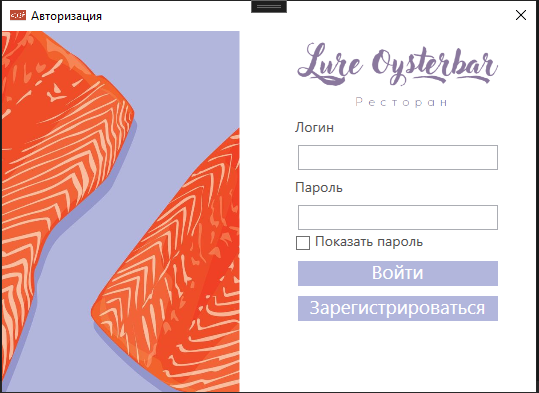
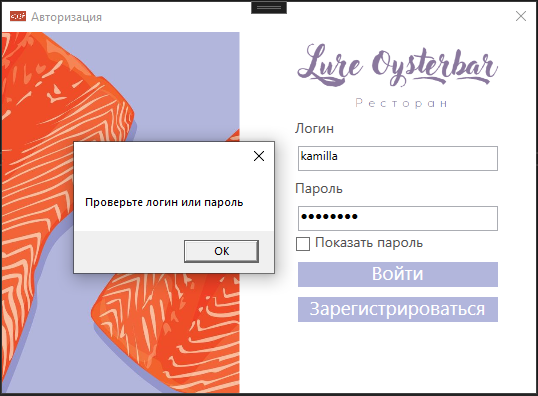


Рисунок 9-Окно авторизации

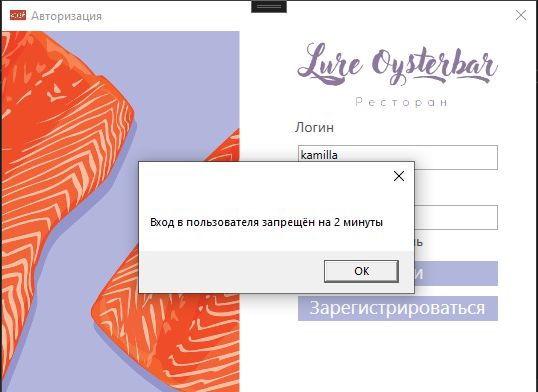
Если пользователь введет логин или пароль не верно, то система оповестит его сообщением об ошибки авторизации



Если повторный ввод пароля снова был безуспешным, то появится окно «Captcha», где пользователя система попросит ввести символы с изображения, если же пользователю они не понятны, то он может обновить картинку. Пользователю предоставлено 3 попытки ввести символы, если он ошибается, то приложение блокирует какой-либо ввод на 10 секунд.



Если после верного ввода «Captcha» пользователь не смог верно ввести пароль, то система заблокируется на 2 минуты



Если логин и пароль введены верно, при нажатии кнопки «Войти» пользователь перенаправляется в свою учетную запись и может использовать весь функционал программы.

Пользователь может иметь разные роли, которые будут соответствовать уровню доступа пользования приложением. В данном приложении имеется три роли, такие как:

* администратор;
* официант;
* гость.

Личный кабинет администратора будет встречать пользователя интерфейсом, на котором расположены кнопки для перехода на окна через которые можно управлять всеми остальными пользователями, например, изменять роль, логин или пароль

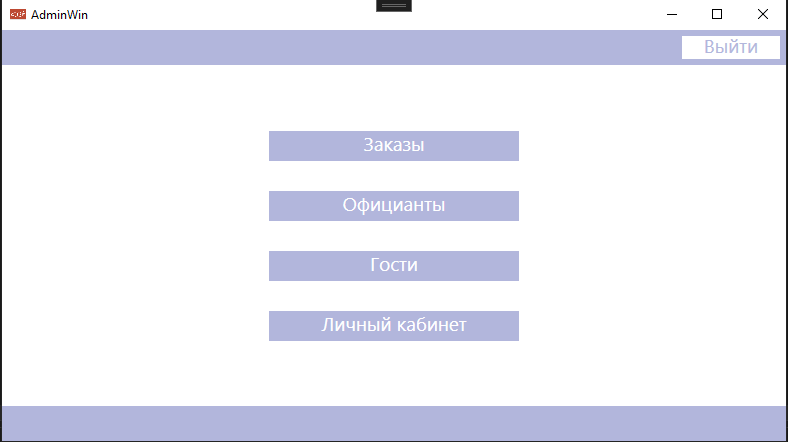


Рисунок 13-Личный кабинет администратора

Если пользователь не зарегистрирован, то может перейти на окно регистрации с помощью кнопки «Зарегистрироваться» и создать свою учетную запись.

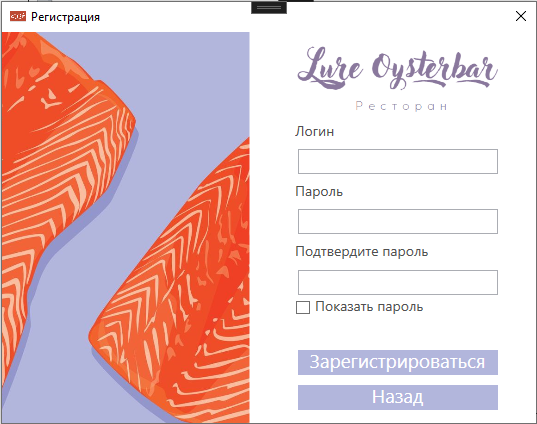


Рисунок 3-Окно регистрации

Создать свою учетную запись пользователь, может введя в советующие поля следующие данные:

* логин;
* пароль;
* повторный ввод пароля.

Так же при регистрации проверяется существует ли такой пользователь в системе, так же проверяется сложность пароля и соответствие на схожесть паролей, если все введено верно, то пользователю создается учетная запись «Гостя» по умолчанию. Так же на форме присутствует индикатор сложности пароля.