**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Факультет прикладной математики и информатики

Кафедра технологий программирования

ОТЧЕТ ПО ППС

Генерация кода проектируемого программного обеспечения и разработка приложения студента 2 курса 13 группы

Козунов Алексей Леонидович

**Преподаватель**

Давидовская Мария Ивановна

Минск 2021

Оглавление

[Глава 1. Постановка задачи 4](#_Toc72911973)

[Система АИС «Оценки и задания» 4](#_Toc72911974)

[Глава 2. Определение требований 6](#_Toc72911975)

[Назначения и цели создания 6](#_Toc72911976)

[Требования к системе 6](#_Toc72911977)

[Глоссарий 6](#_Toc72911978)

[Функциональные возможности 6](#_Toc72911979)

[Требования по реализации 6](#_Toc72911980)

[Надёжность 7](#_Toc72911981)

[Производительность 7](#_Toc72911982)

[Безопасность 7](#_Toc72911983)

[Бизнес диаграмма 7](#_Toc72911984)

[Основная диаграмма 8](#_Toc72911985)

[Диаграмма деятельности 8](#_Toc72911986)

[Описание актеров: 9](#_Toc72911987)

[Варианты использования: 9](#_Toc72911988)

[Спецификации: 9](#_Toc72911989)

[Вариант использования «Войти в систему»: 9](#_Toc72911990)

[Сценарий событий для прецедента «Просмотр журнала учеником»: 10](#_Toc72911991)

[Сценарий событий для прецедента «Изменить данные». 10](#_Toc72911992)

[Глава 3. Анализ вариантов использования 11](#_Toc72911993)

[Общая диаграмма 11](#_Toc72911994)

[Диаграммы классов 11](#_Toc72911995)

[Диаграммы последовательности 13](#_Toc72911996)

[Для действия “Войти в систему” 15](#_Toc72911997)

[Для действия “Посмотреть журнал” 16](#_Toc72911998)

[Для действия “ Выдача заданий” 17](#_Toc72911999)

[Глава 4. Проектирование архитектуры системы 18](#_Toc72912000)

[Диаграмма пакетов 18](#_Toc72912001)

[Диаграмма развертывания 18](#_Toc72912002)

[Диаграмма компонентов 19](#_Toc72912003)

[Диаграммы объектов 19](#_Toc72912004)

[Макеты интерфейса 20](#_Toc72912005)

[Глава 5. Проектирование элементов системы 22](#_Toc72912006)

[Описание проектных классов системы 22](#_Toc72912007)

[Диаграммы состояний 23](#_Toc72912008)

[Глава 6. Разработка приложения 25](#_Toc72912009)

[Пример генерации кода Login: 25](#_Toc72912010)

[Пример кода класса Login: 25](#_Toc72912011)

[Пример создания таблицы Users: 27](#_Toc72912012)

[Скриншоты приложения: 28](#_Toc72912013)

[Глава 7. Результаты тестирования системы 33](#_Toc72912014)

[Unit testing 33](#_Toc72912015)

[Integration testing 33](#_Toc72912016)

[System testing 33](#_Toc72912017)

[Acceptance testing 33](#_Toc72912018)

[Scenario testing 33](#_Toc72912019)

[Список литературы 35](#_Toc72912020)

# Глава 1. Постановка задачи

## Система АИС «Оценки и задания»

Система контроля успеваемости используется в университете для отслеживания работы студентов в семестре.

Сотрудники учебного отдела вводят в систему данные о студентах, преподавателях и учебных дисциплинах. Данные могут быть изменены и/или удалены. Студенты университета делятся на курсы. Курс делится на группы. Каждая группа студентов ходит на занятия по дисциплинам, которые ведутся преподавателями. Преподаватель может вести любое количество дисциплин у одной или нескольких групп. Несколько преподавателей могут вести одну и ту же дисциплину в одной и той же группе или в разных группах.

Преподаватели пользуются системой, чтобы отмечать успехи и неуспехи студентов, посещающих их занятия. В течение семестра преподаватель последовательно выдаёт студентам "своей" группы несколько заданий по своей дисциплине. Для каждого задания устанавливается контрольный срок, когда оно должно быть сдано. Когда студент сдаёт задание, преподаватель отмечает этот факт в системе. Он ставит «±», если задание сдано не до конца, «+» -- если задание сдано полностью. Система контроля автоматически ставит «−», если задание пока не сдано. От времени сдачи зависит цвет, которым отметка о сдаче выделена в системе. Если сдача проходила до установленного срока (за неделю и более), отметка высвечивается зелёным. Если сдача проходила после установленного срока (через неделю и более) отметка высвечивается красным. В остальных случаях отметка чёрная. Поскольку сдача может быть не полной, студент может сдавать задание несколько раз. После получения отметки «±» за это же задание может быть поставлена отметка «±» или «+», относящаяся к старшей дате. если за задание уже стоит отметка «+» никакая отметка не может быть поставлена за это же задание. По этим правилам преподаватели могут ставить одному и тому же студенту несколько отметок о сдаче одного и того же задания. Если преподаватель ошибся, он может удалить отметку или изменить её.

За сдачу заданий студентам начисляются баллы. Баллы зависят от текущей даты и от сведений об отметке по этому заданию, относящихся к самой старшей дате. Если на текущую дату студент имеет отметку «−» или «±», то ему начисляются штрафные баллы в количестве равном, количеству полных недель с момента наступления контрольного срока до текущей даты. Количество штрафных баллов неотрицательно. Если на текущую дату студент имеет отметку «+», полученную раньше наступления срока сдачи, то ему начисляются бонусные баллы в количестве, равном количеству полных недель с момента получения отметки «+» до наступления контрольного срока. Количество бонусных баллов неотрицательно. Если на текущую дату студент имеет отметку «+», полученную позже наступления срока сдачи, то ему начисляются штрафные баллы в количестве равном количеству полных недель с момента наступления контрольного срока до момента получения отметки. Количество штрафных баллов неотрицательно.

Преподаватель для любой группы, в которой он ведёт занятия, может видеть полный список группы с полным набором отметок по заданиям, раскрашенных по указанным выше правилам, с указанием штрафных и бонусных баллов и общей суммы баллов по всем заданиям. Преподаватель может начислять студентам, у которых он ведёт задания, дополнительные бонусные (и штрафные) баллы, поощряя или наказывая их по своему усмотрению.

Студенты также могут пользоваться системой, чтобы узнать сроки сдачи заданий и результаты собственных сдач. Каждому студенту высвечивается сводная таблица, с отметками о сдаче заданий по всем, изучаемым им дисциплинам, с указанием штрафных и бонусных баллов, и 9 номера в рейтинге по данной дисциплине. Рейтинг составляется на основе вычитания из бонусных баллов штрафных баллов. Сортировка в рейтинге идёт по не возрастанию. Чужие отметки студенты видеть не должны, это конфиденциальная информация.

# Глава 2. Определение требований

## Назначения и цели создания

Система создаётся с целью упорядочить работу государственного учреждения образования. Система должна предоставлять весь необходимый функционал для отслеживания работы студентов в семестре.

## Требования к системе

Необходимо хранить информацию о отметках, данных о студентах, дисциплинах и преподавателях.

## Глоссарий

|  |  |
| --- | --- |
| Студент (Student) | Пользователь, выполняющий задание в система |
| Преподаватель (Professor) | Пользователь, создающий и назначающий задания |
| Задание (Task) | Вид поручения учителя учащимся, в котором содержится требование выполнить учебные (теоретические или практические) действия |
| Сотрудник учебного отдела (Administrator) | Управляет учетными записями учащихся, преподавателей и дисциплин |
| Журнал (Journal) | Содержит оценку за каждое задание, сроки для выполнения заданий |
| Система контроля (Control system) | Система, начисляющая баллы за задания |

## Функциональные возможности

Система должна автоматически начислять баллы за каждое задание.

## Требования по реализации

Система должна быть совместима с Windows.

Преподаватель для любой группы, в которой он ведёт занятия, может видеть полный список группы с полным набором отметок по заданиям, раскрашенных по указанным выше правилам, с указанием штрафных и бонусных баллов и общей суммы баллов по всем заданиям.

Студенты также могут пользоваться системой, чтобы узнать сроки сдачи заданий и результаты собственных сдач. Каждому студенту высвечивается сводная таблица, с отметками о сдаче заданий по всем, изучаемым им дисциплинам, с указанием штрафных и бонусных баллов, и 9 номера в рейтинге по данной дисциплине. Рейтинг составляется на основе вычитания из бонусных баллов штрафных баллов. Сортировка в рейтинге идёт по не возрастанию. Чужие отметки студенты видеть не должны, это конфиденциальная информация.

## Надёжность

Система должна быть в работоспособном состоянии 24 часа в день 7 дней в неделю.

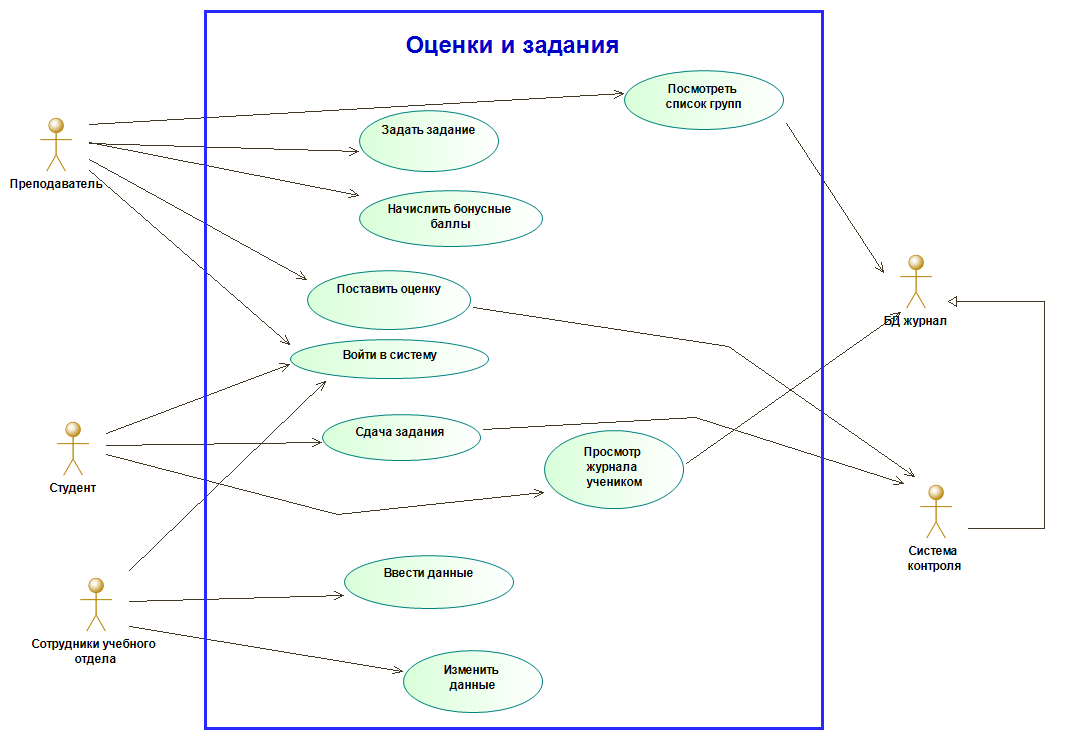
## Производительность

Система должна поддерживать несколько одновременно работающих пользователей.

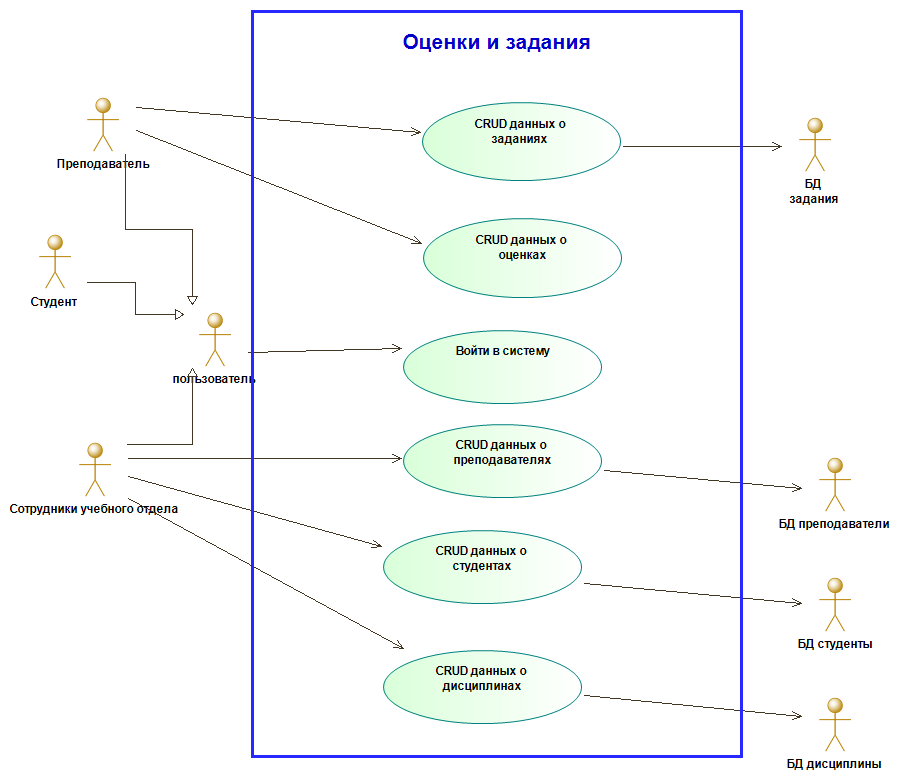
## Безопасность

В системе должна быть предусмотрена аутентификация для всех пользователей системы.

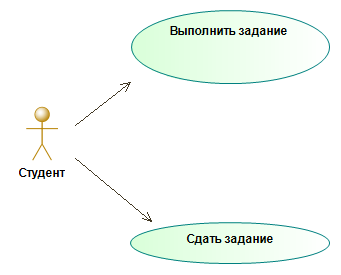
## Бизнес диаграмма



## Основная диаграмма



## Диаграмма деятельности



## Описание актеров:

|  |  |
| --- | --- |
| Сотрудник учебного отдела | Имеет доступ к редактированию учетных записей пользователей, данных о дисциплинах |
| Студент | Пользователь, выполняющий задание |
| Преподаватель | Пользователь, создающий и назначающий задания в системе |

## Варианты использования:

|  |  |
| --- | --- |
| Задать задание | Используется преподавателем для назначения заданий студентам |
| Посмотреть список групп/ Просмотр журнала учеником | Используется учащимися и преподавателями для просмотра результатов |
| Начислить бонусные баллы | Используется преподавателем для выдачи бонусных баллов |
| Поставить оценку | Используется преподавателем для фиксации выполненных заданий |
| Сдача задания | Используется студентом для сдачи заданий |
| Ввести данные/ Изменить данные | Используется сотрудником учебного отдела для добавления/удаления/изменения данных |

## Спецификации:

### Вариант использования «Войти в систему»:

Краткое описание

Данный вариант использования описывает вход пользователя в систему.

Основной поток событий

1. Система запрашивает имя пользователя и пароль.

2. Пользователь вводит имя и пароль.

3. Система подтверждает правильность имени и пароля, определяет тип пользователя (студент, профессор или регистратор) и выводит главное меню, дающее доступ к функциям системы в соответствии с типом пользователя.

Альтернативные потоки

Неправильное имя/пароль

1. Система обнаруживает, что комбинация имени и пароля не верна.

2. Система сообщает об ошибке и предлагает пользователю либо заново ввести имя и пароль, либо отказаться от входа в систему.

3. Пользователь сообщает системе свой выбор.

4. В соответствии с выбором пользователя либо выполнение переходит на начало основного потока, либо вариант использования завершается.

Предусловия

Для входа в систему пользователь должен быть зарегистрирован в системе.

Постусловие

Если вариант использования выполнен успешно, система предоставляет доступ к главному меню пользователю, сообщившему верную комбинацию имени и пароля. В противном случае система гарантирует, что пользователю, сообщившему неверную комбинацию имени и пароля, доступ к меню не будет предоставлен.

### Сценарий событий для прецедента «Просмотр журнала учеником»:

Краткое описание

Позволяет узнать сроки сдачи заданий и результаты собственных сдач.

Основной поток событий

1. Выбор нужной дисциплины.
2. Учащийся может посмотреть сроки сдачи заданий и результаты собственных сдач.

Предусловия.

Если пользователь находится в системе, он может посмотреть журнал.

Постусловия

Отсутствуют.

### Сценарий событий для прецедента «Изменить данные».

Краткое описание

Используется сотрудником учебного отдела для добавления/удаления/изменения данных

Основной поток событий

1. Выбрать нужного пользователя
2. Удалить пользователя или дисциплину, а также изменить данные учетной записи существующего пользователя или дисциплины

Предусловия.

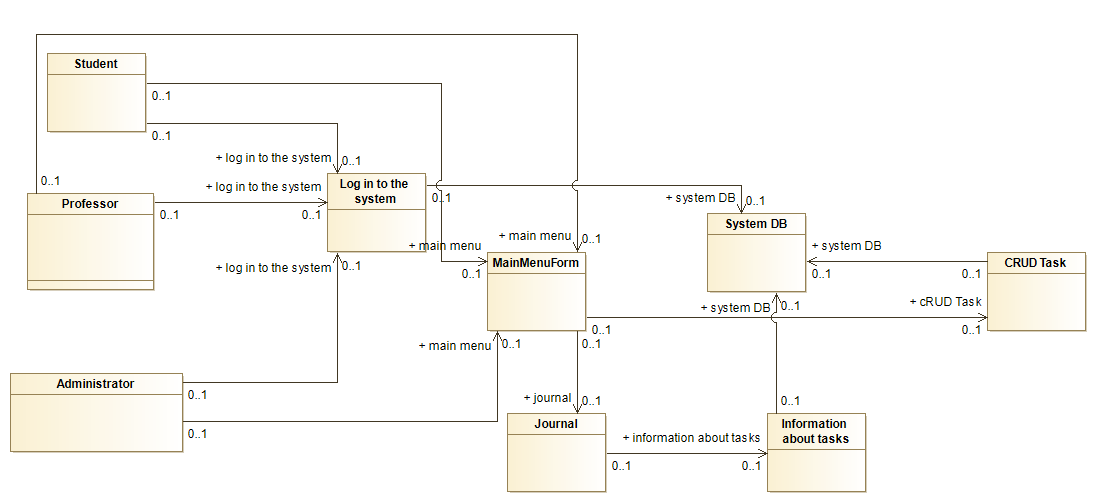
Необходимость изменить данные.

Постусловия

Все изменения сохраняются в БД.

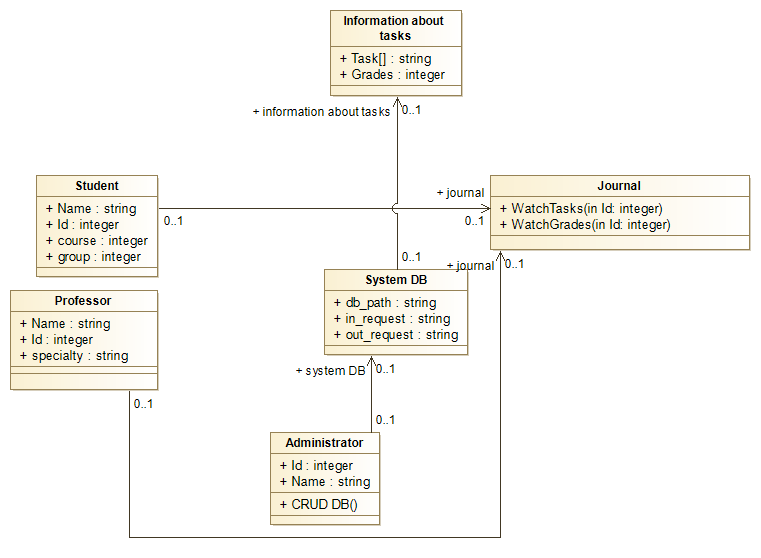
# Глава 3. Анализ вариантов использования

## Общая диаграмма

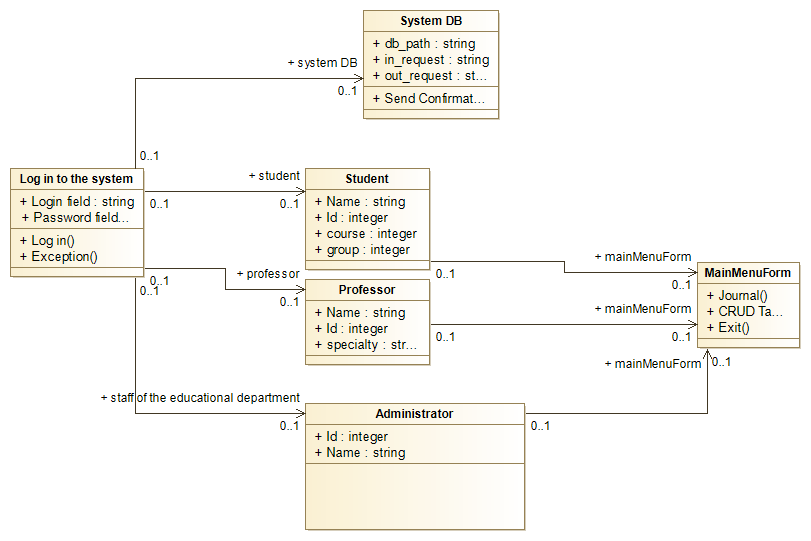


Данная диаграмма описывает работу программы, она является главной

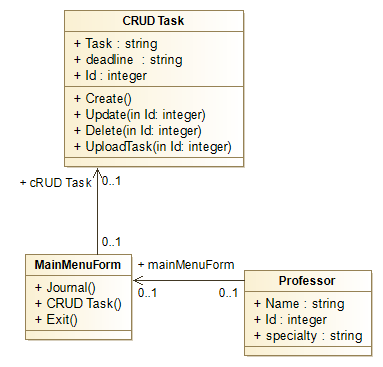
## Диаграммы классов



Данная диаграмма описывает использование журнала

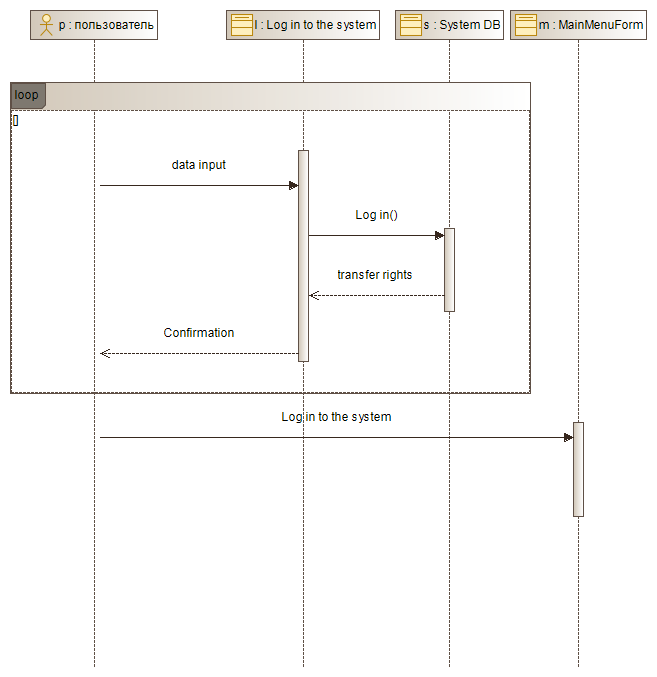


Данная диаграмма описывает вход пользователя в систему

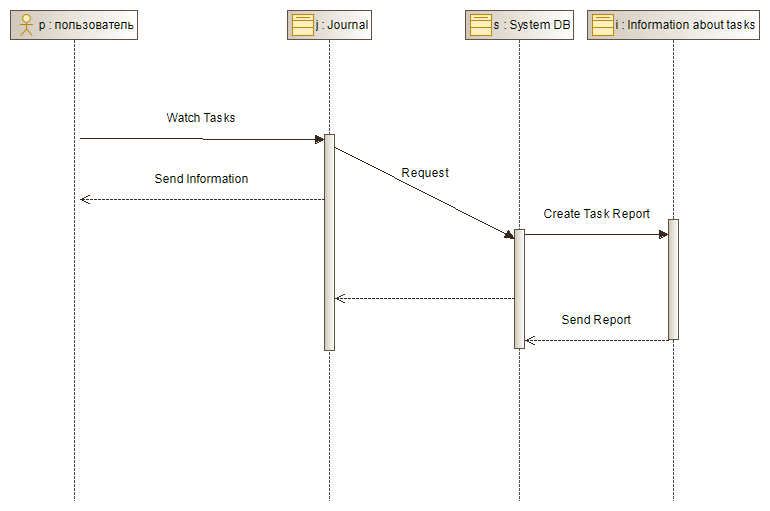


Данная диаграмма описывает добавление задания

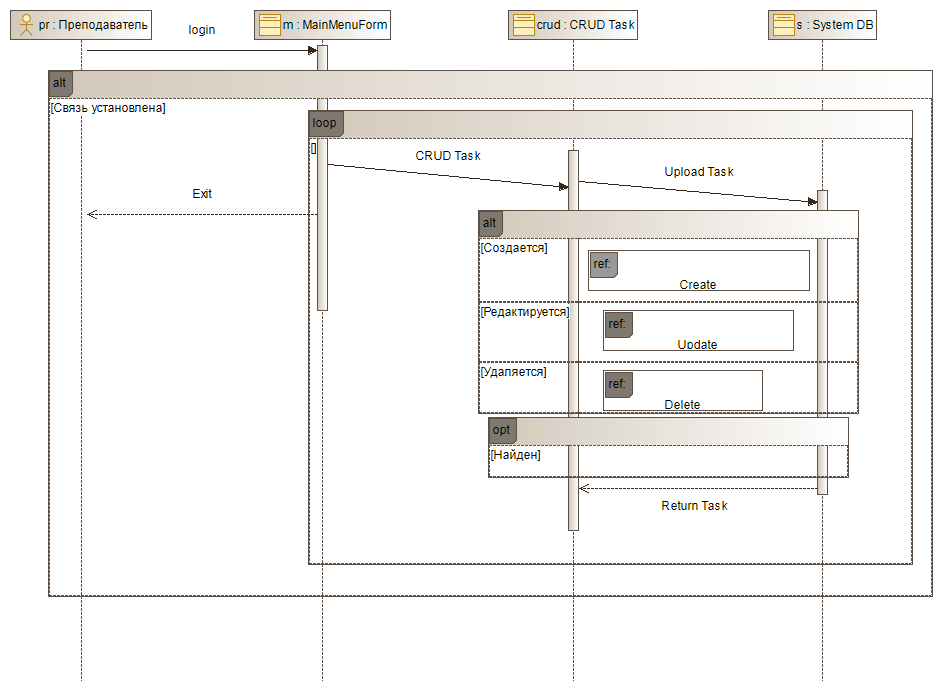
## Диаграммы последовательности



Данная диаграмма описывает вход пользователя в систему

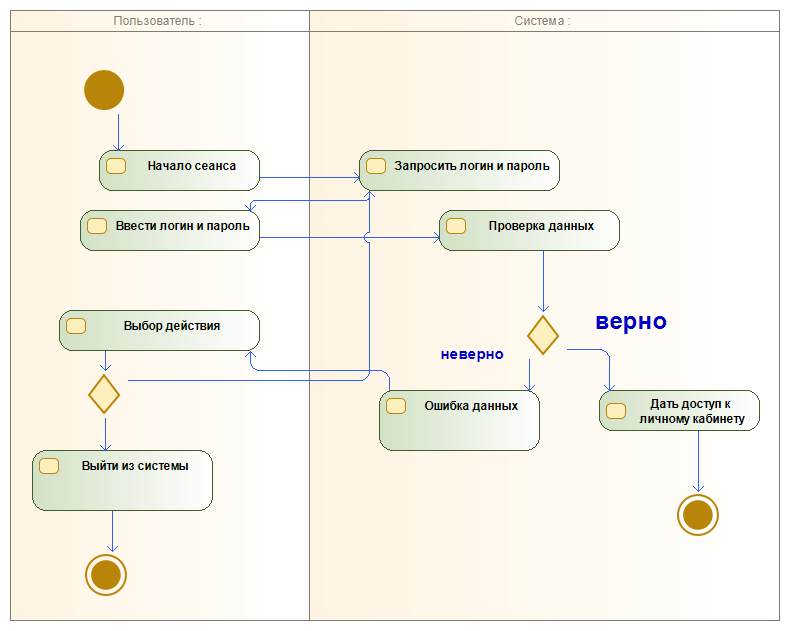


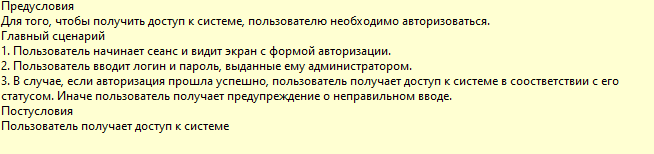
Данная диаграмма описывает использование журнала



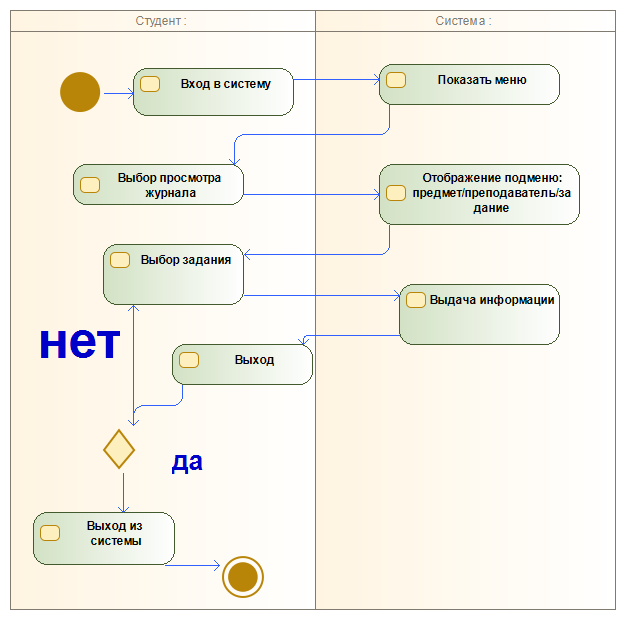
Данная диаграмма описывает добавление задания

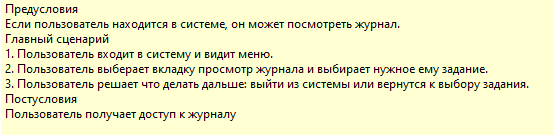
## Для действия “Войти в систему”



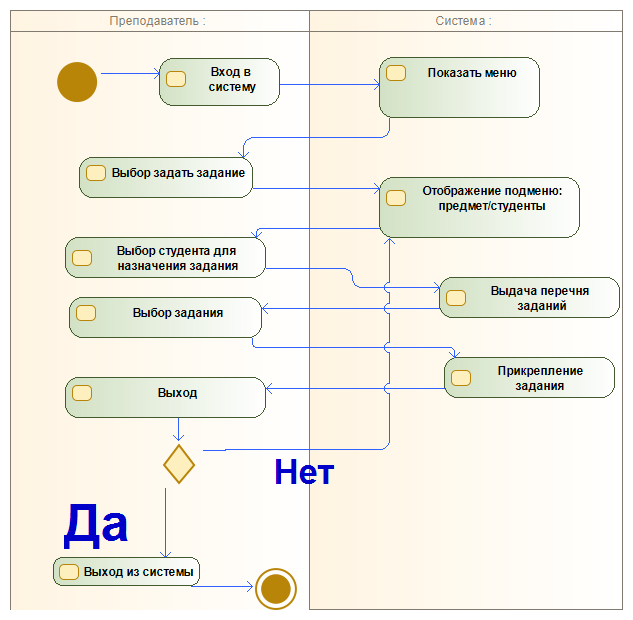


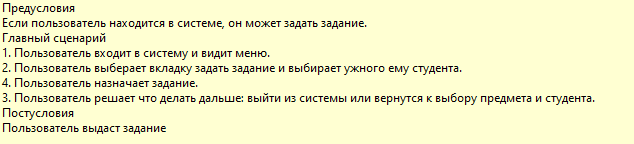
## Для действия “Посмотреть журнал”





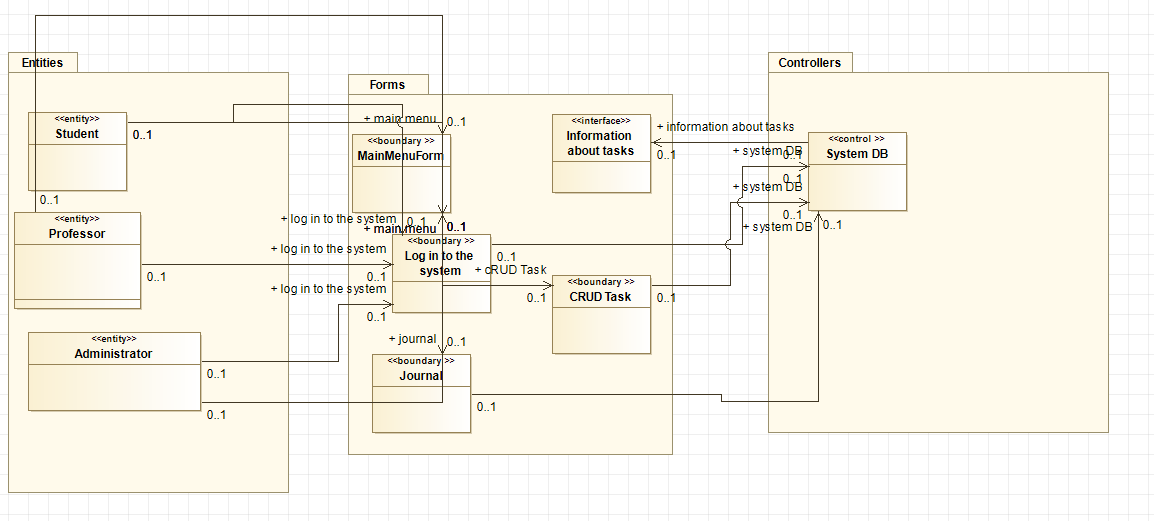
## Для действия “ Выдача заданий”



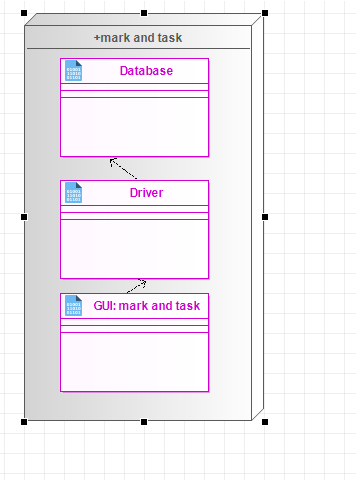


# Глава 4. Проектирование архитектуры системы

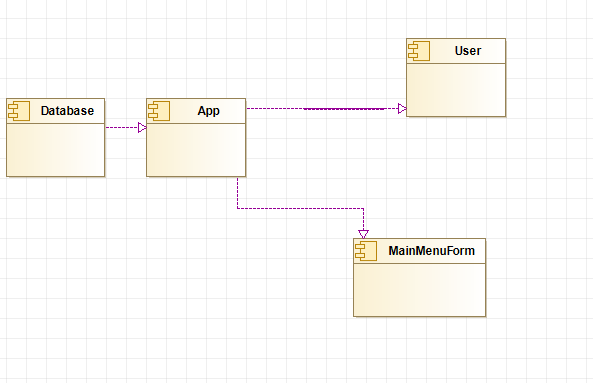
## Диаграмма пакетов



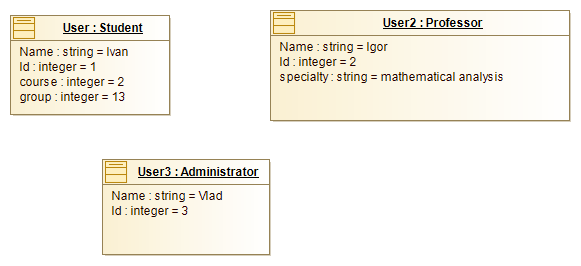
## Диаграмма развертывания



## Диаграмма компонентов

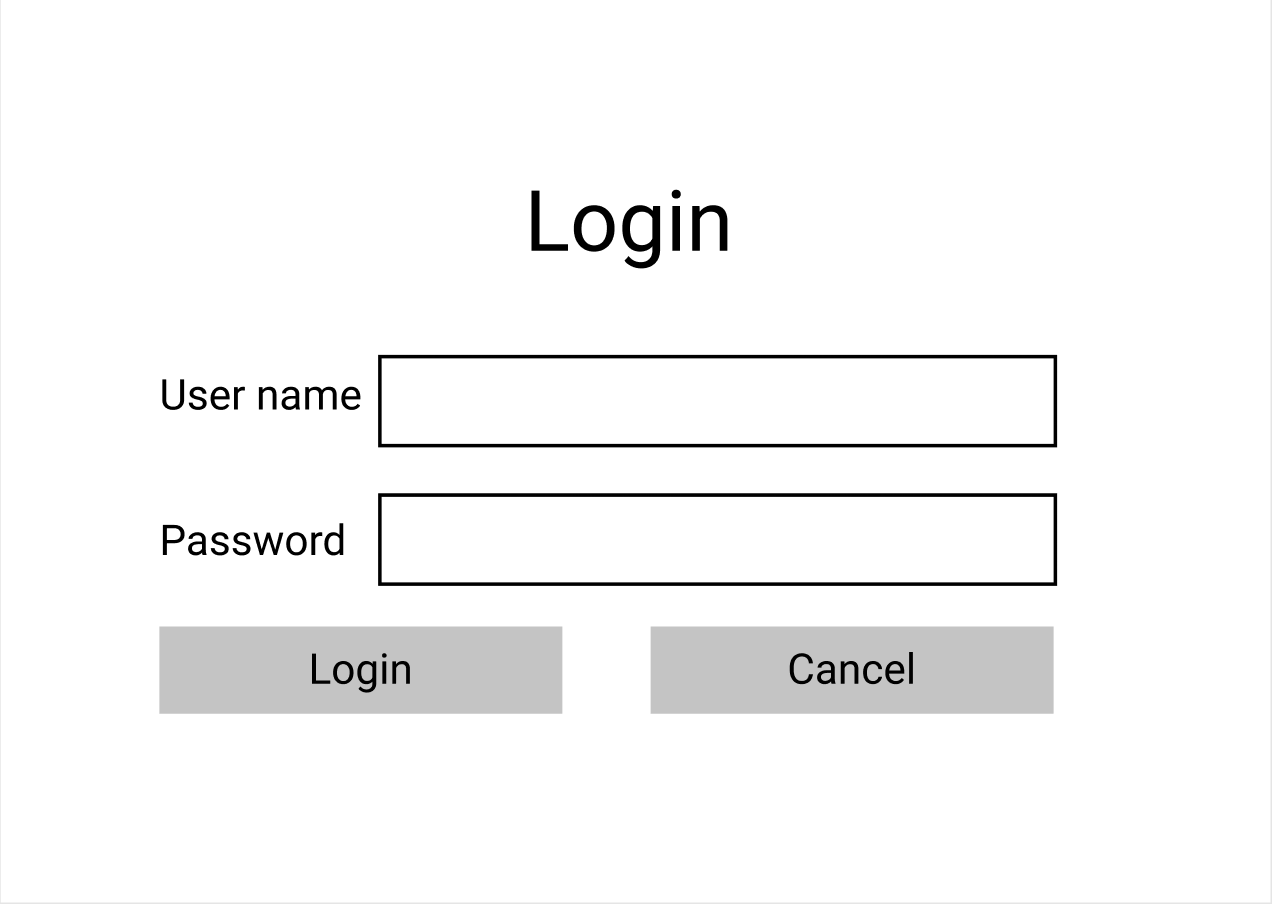


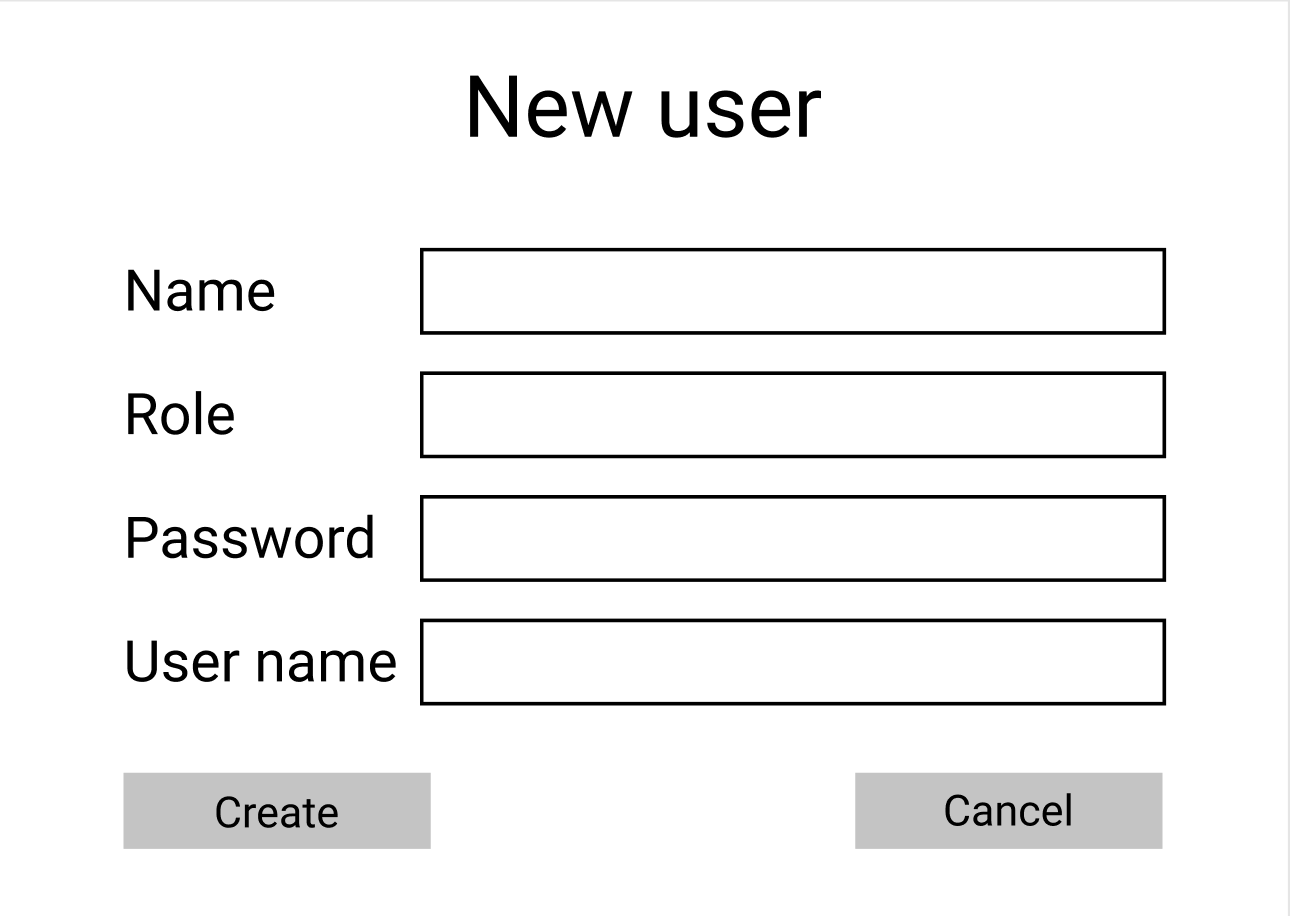
## Диаграммы объектов



Данная диаграмма описывает создание объектов

## Макеты интерфейса









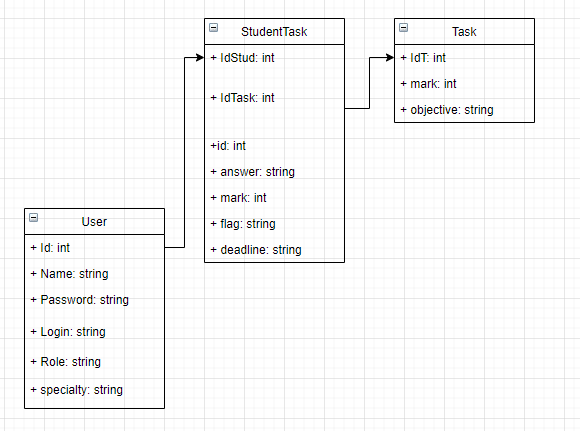
# Глава 5. Проектирование элементов системы

## Описание проектных классов системы

Приложение состоит из следующих классов:

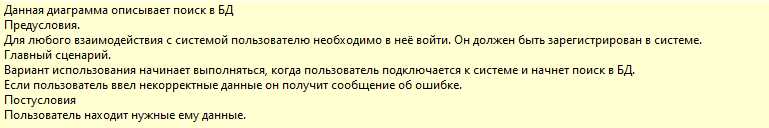
различные графические формы (AddTask, AddUser, EditJournal, GiveTask, JournalBD, Login, MenuForAdmin, MenuForProfessor, MenuForStudent, MenuJournal, SubmitTask) для взаимодействия с пользователем, классы различных сущностей (Users).

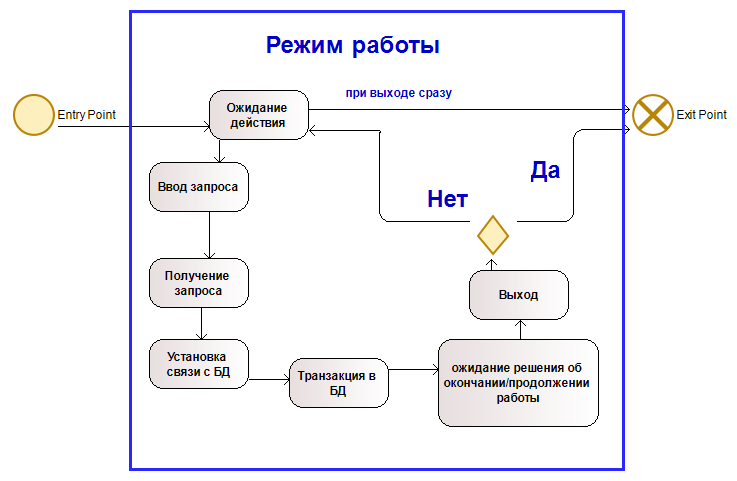
Модель базы данных

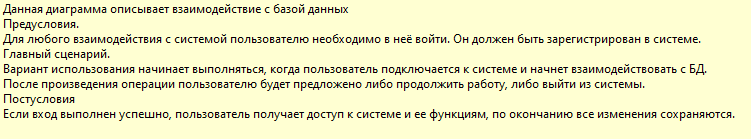


## Диаграммы состояний









# Глава 6. Разработка приложения

Я не использовал некоторые поля в классах, так как в ходе создания приложения они не понадобились, в общем именно функционал везде совпадает.

## Пример генерации кода Login:

**import** com.modeliosoft.modelio.javadesigner.annotations.objid;

*@objid ("a5f220b9-9bb5-4b5b-94c6-91dcf76f1c26")*

**public** **class** Log in to the system {

*@objid ("51c9cbe7-2ed7-4c30-9253-4c3f761da630")*

**public** String Login field;

*@objid ("b9a7d3f2-82fc-48b8-88dd-9f8a8ae0295f")*

**public** String Password field;

*@objid ("5058712c-720a-42ef-a4ec-1cb4e62b9f4a")*

**public** Student student;

*@objid ("d12a5e15-0f56-4016-98ff-60156a7d00e0")*

**public** Professor professor;

*@objid ("bd1650e1-f219-4ad9-a7b9-24d6ce60dd3c")*

**public** Administrator staff of the educational department;

*@objid ("7bf64c91-31d5-4a58-bb88-8374d2384ae0")*

**public** System DB system DB;

*@objid ("1e30aba0-9267-4c3c-8983-2db797f52eff")*

**public** **void** Log in() {

}

*@objid ("ea370c11-a985-4d78-9f66-ab28a436f35d")*

**public** **void** Exception() {

}

}

## Пример кода класса Login:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SQLite;

using System.IO;

using System.Diagnostics;

namespace PPS\_Lab

{

public partial class Login : Form

{

SQLiteConnection Connect;

string name;

string password;

public Login(SQLiteConnection Connect)

{

InitializeComponent();

this.Connect = Connect;

}

private void loginB\_Click(object sender, EventArgs e)

{

User user = new User();

MenuForStudent menuForStudent;

MenuForAdmin menuForAdmin = new MenuForAdmin(Connect);

MenuForProfessor menuForProfessor = new MenuForProfessor(Connect);

if ((name == "" || name == null) || (password == "" || password == null))

{

MessageBox.Show("Все поля должны быть заполнены!");

}

else if (password.Length > 20)

{

MessageBox.Show("Пороль не должен содержать больше 20 символов!");

}

else

{

SQLiteCommand Command = new SQLiteCommand

{

Connection = Connect,

CommandText = String.Format("SELECT \* FROM Users WHERE login = '{0}' and password = '{1}'", name, password)

};

SQLiteDataReader sqlReader = Command.ExecuteReader();

sqlReader.Read();

if (sqlReader.HasRows)

{

user = new User(sqlReader);

if (sqlReader.GetString(3) == "0")

{

menuForAdmin.Show();

this.Hide();

}

else if (sqlReader.GetString(3) == "2")

{

menuForStudent = new MenuForStudent(Connect, user);

menuForStudent.Show();

this.Hide();

}

else if (sqlReader.GetString(3) == "1")

{

menuForProfessor.Show();

this.Hide();

}

else

{

MessageBox.Show("Неверные данные!");

}

}

else

{

MessageBox.Show("Неверные данные, попробуте войти еще раз!");

}

}

}

private void cancelB\_Click(object sender, EventArgs e)

{

Close();

}

private void userNameL\_TextChanged(object sender, EventArgs e)

{

name = userNameL.Text;

}

private void passwordL\_TextChanged(object sender, EventArgs e)

{

password = passwordL.Text;

}

}

}

## Пример создания таблицы Users:

CREATE TABLE Users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name VARCHAR(20) NOT NULL,

login VARCHAR(20) NOT NUll UNIQUE,

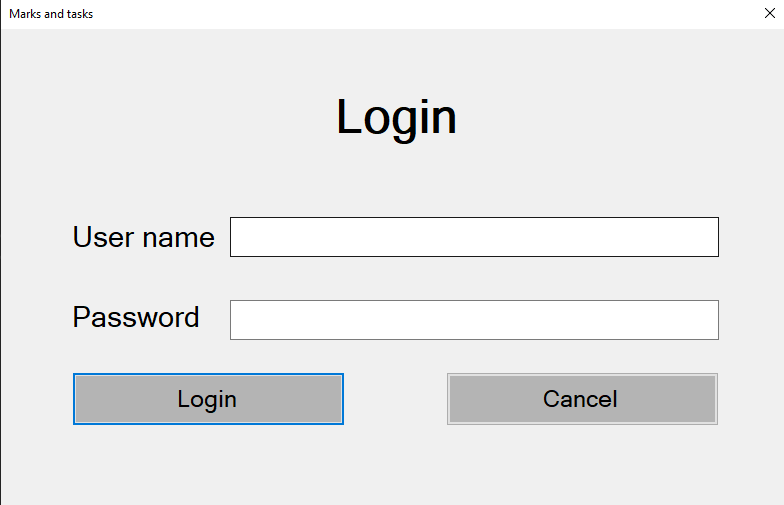
role VARCHAR(20) NOT NULL CHECK(role =='0' or role =='1' or role=='2'),

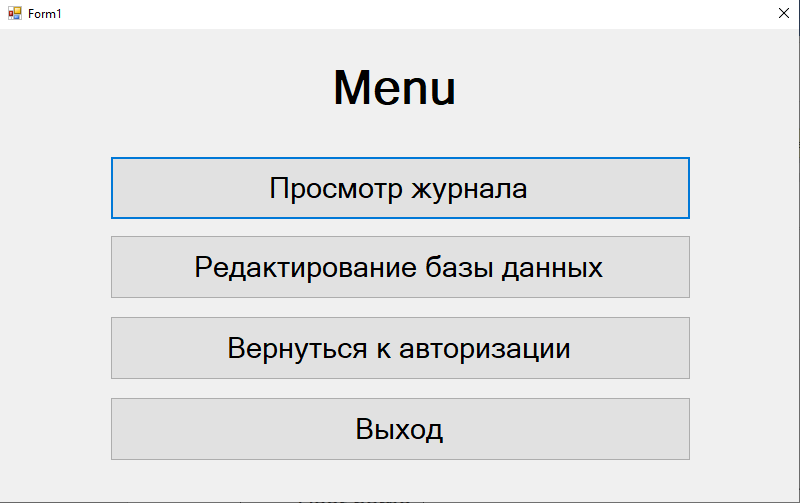
specialization VARCHAR(20),

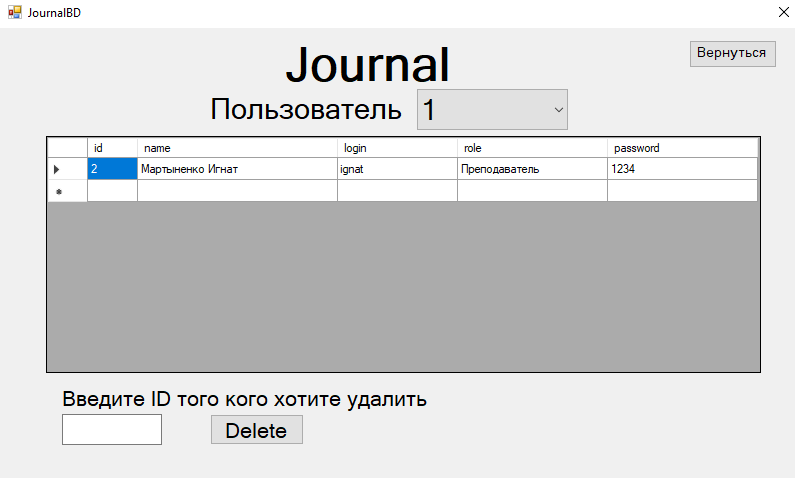
password INT NOT NULL

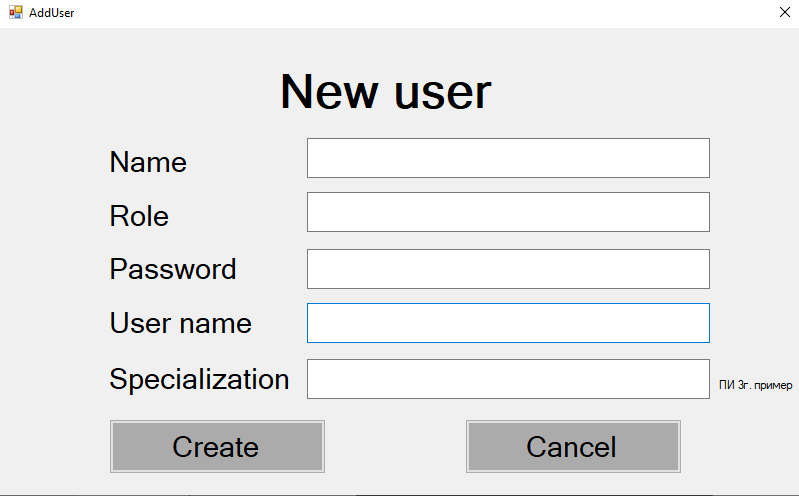
);

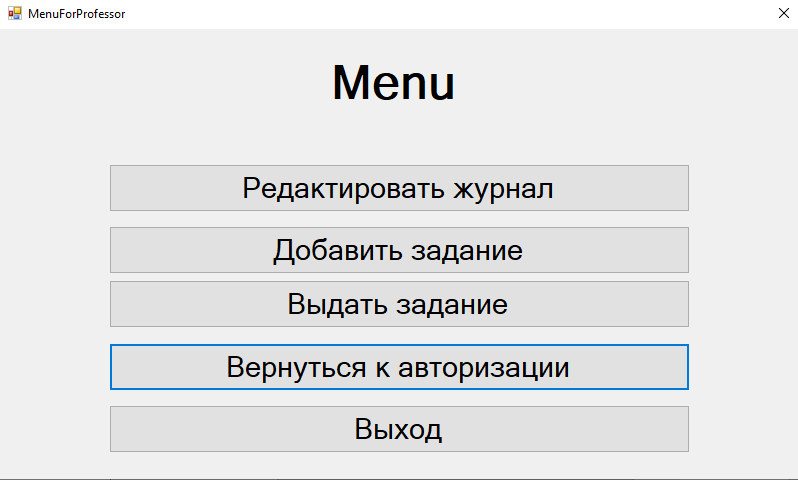
## Скриншоты приложения:

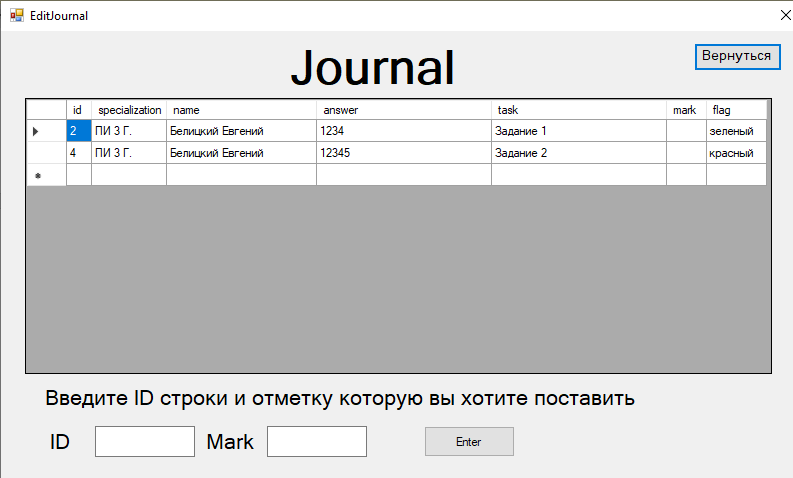


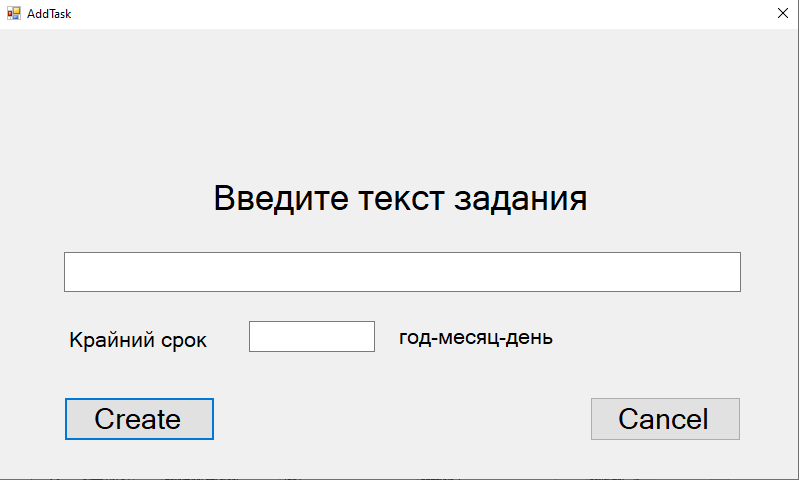


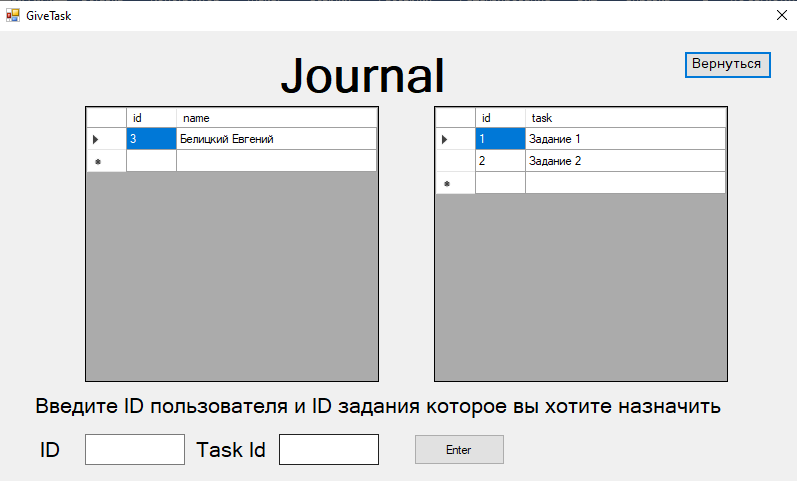


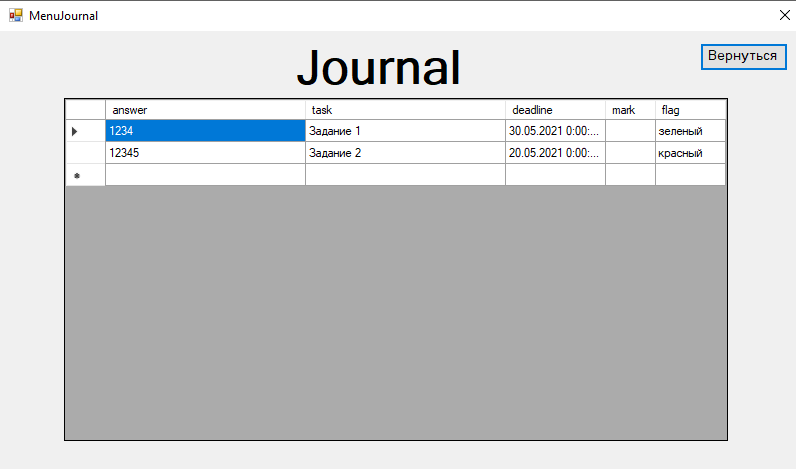


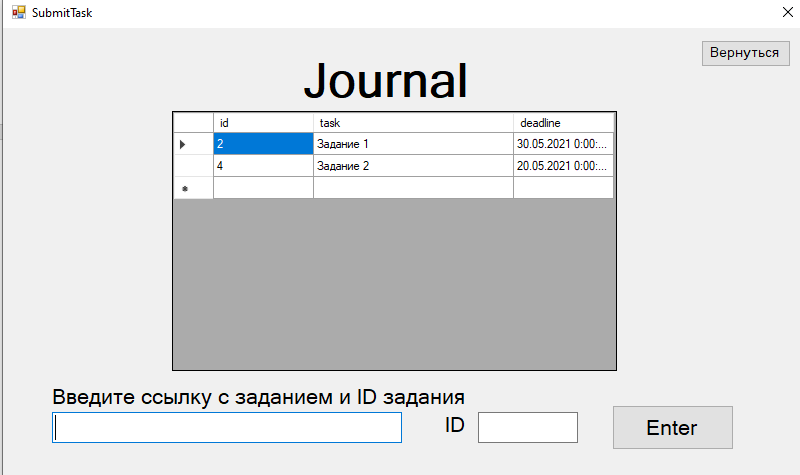












# Глава 7. Результаты тестирования системы

## Unit testing

1. Проведем тест на добавление имени пользователя администратором. Имя должно содержать только буквы (A-Z, А-Я).  Тестируем метод createUserB\_Click(object sender, EventArgs e) класса AddUser.

Вход 123 получим false. Вход Иван получим true.

1. Тест на правильность ввода придуманного пароля. Пароль не может превышать 20 знаков. Тестируем метод loginB\_Click(object sender, EventArgs e) класса Login.

Вход qwerty получим true. Вход 123456789012345678901 false.

1. Тест на правильность выставление флага. При выставлении флага будем сравнивать время сдачи и последний срок. Тестируем метод enterB\_Click(object sender, EventArgs e) класса SubmitTask.

Вход 2021-05-25 дедлайн 2021-05-25 флаг зеленый. Вход 2021-05-25 дедлайн 2021-05-26 флаг красный.

## Integration testing

Произведем проверку на подключение к нашей базе данных.

## System testing

Так как для работы с приложением пользователю необходимо авторизироваться, то необходима проверить корректность ввода логина и пароля, а также подключения к БД. При входе система будет искать введенные данные в БД и если найдет совпадение, то даст полагающиеся права.

## Acceptance testing

Проверяем, имеет ли система следующий функционал:

1. Возможность получить доступ к таблице пользователей
2. Возможность получить доступ к таблице журнала
3. Возможность добавлять и изменять информацию о пользователях
4. Возможность добавлять и изменять задания

## Scenario testing

**Сценарий событий для прецедента «Войти в систему».**

Предусловия

Для входа в систему пользователь должен быть зарегистрирован в системе.

Главный сценарий

Вначале пользователь при входе в систему вводит свой пароль и логин. Если пароль и логин верные он получает доступ к системе, если нет, то пользователю предлагают проверить введенные данные или же выйти из системы.

Альтернативные сценарии

Неверные данные: введено неправильное имя или пароль. Пользователь должен повторить ввод или выйти из системы.

Постусловие

Если вариант использования выполнен успешно, система предоставляет доступ к главному меню пользователю, сообщившему верную комбинацию имени и пароля. В противном случае система гарантирует, что пользователю, сообщившему неверную комбинацию имени и пароля, доступ к меню не будет предоставлен.

Тестирование

Пользователь ввел логин и пароль (Иван, 1111). Логин: только буквы и пробелы, Пароль: цифры, длинна менее 20 и более 0. После поиска данных в БД они оказались верными, был предоставлен доступ к системе. Или же после поиска данных в БД они оказались не верными, было предложено войти снова.

**Сценарий событий для прецедента «Сдать задание».**

Предусловия

Для входа в систему пользователь должен быть зарегистрирован в системе и быть авторизирован как студент.

Главный сценарий.

Пользователь, вошедший в систему, открывает раздел сдать задание, сбрасывает выполненное задание в систему. При правильной отправке в журнале помечается, что студент сдал задание. В зависимости от времени сдачи студент получает различные пометки

Постусловия

Если все прошло успешно преподаватель может проверить задание и поставить балл студенту.

Тестирование

Пользователь входит в систему и нажимает на вкладку сдать задание. Во вкладке он добавляет ответ на свое задание в систему. Ответ: ссылка на решенное задание, он добавляется к конкретному заданию по ID, также система проверяет, когда ты сдал задание до дедлайна или после. Студент проверяет в журнале какая пометка стоит.

# Список литературы

Документация Modelio: <https://github.com/ModelioOpenSource/Modelio/wiki/Modelio-User-Documentation>

Р. Лафоре «Объектно-ориентированное программирование в C++», Глава 16