Министерство образования и науки Российской Федерации

Волжский политехнический институт (филиал)

федерального государственного бюджетного образовательного учреждения высшего образования

«Волгоградский государственный технический университет»

(ВПИ (филиал) ВолгГТУ)

|  |  |
| --- | --- |
| Факультет | Инженерно-экономический факультет |
| Кафедра | Информатика и технология программирования |

**КОНТРОЛЬНАЯ РАБОТА**

|  |  |
| --- | --- |
| по дисциплине: | Технология командной разработки программных систем |
| на тему: | Разработка и тестирование автоматизированной информационной системы |
| Вариант № 3 | |

|  |  |
| --- | --- |
| Студент | БОРЗИН Роман Юрьевич, ЧУРЗИНА Екатерина Олеговна |
|  | (фамилия, имя, отчество) |

|  |  |
| --- | --- |
| Группа | ВИП-408 |
|  |  |

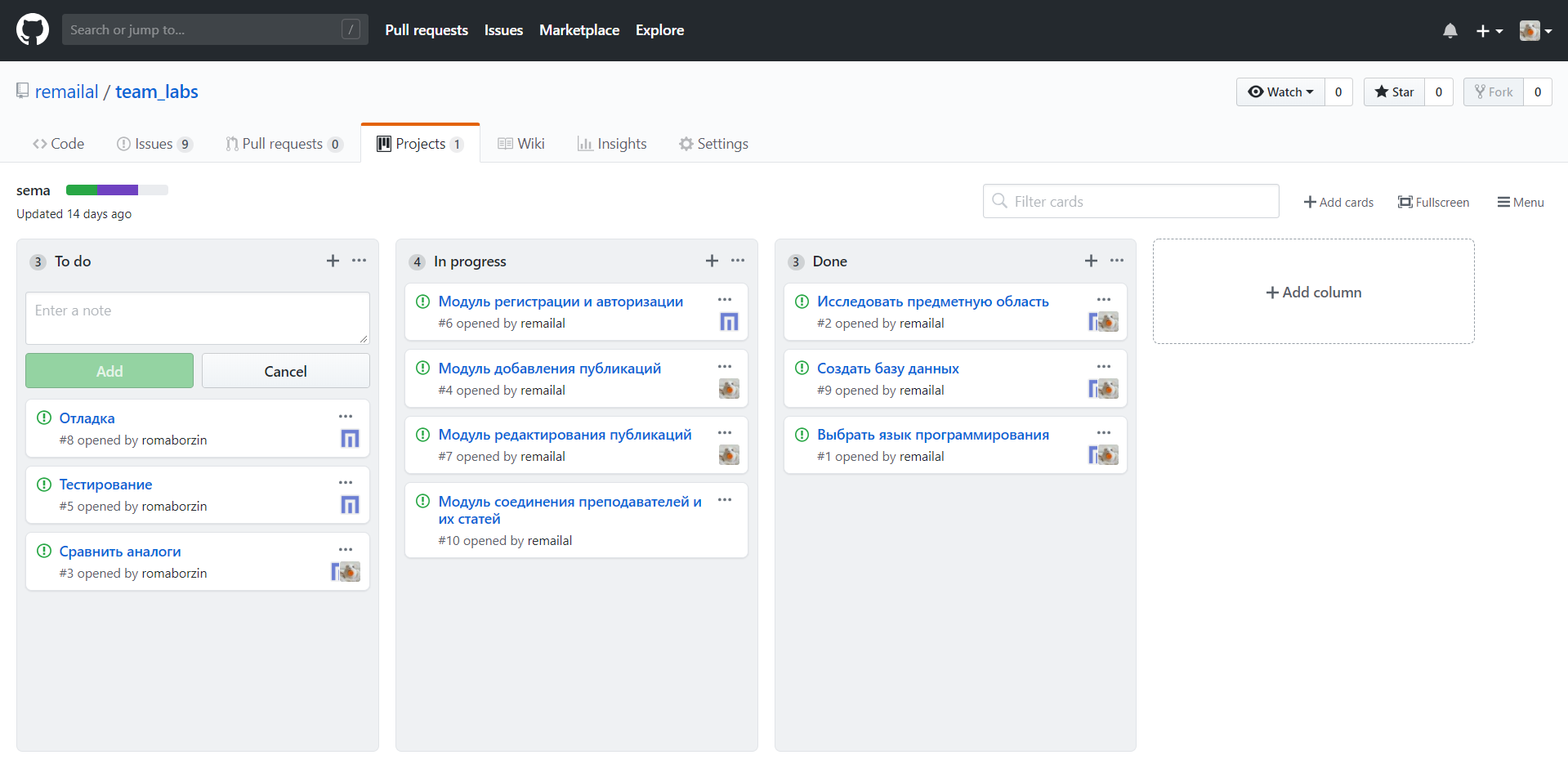
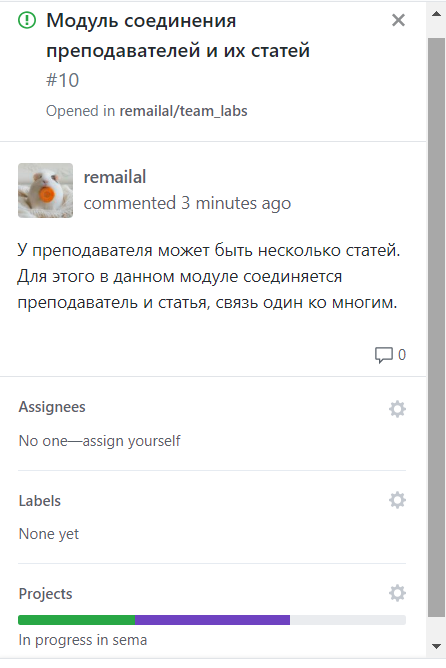
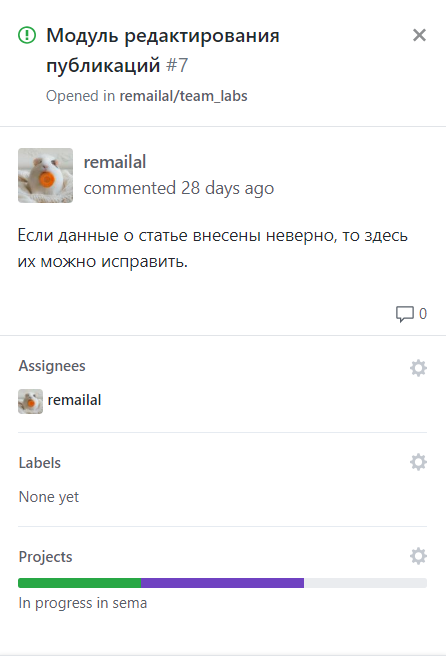
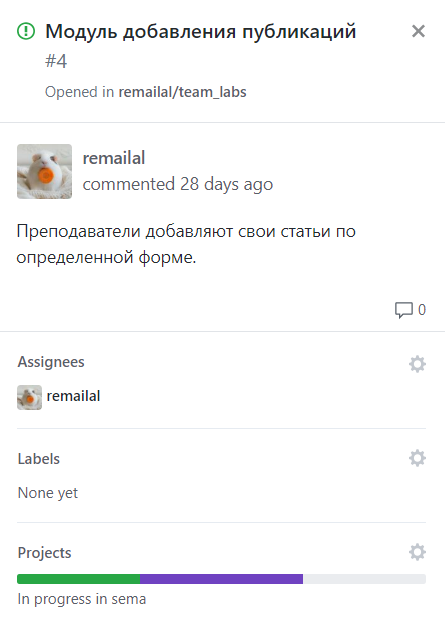
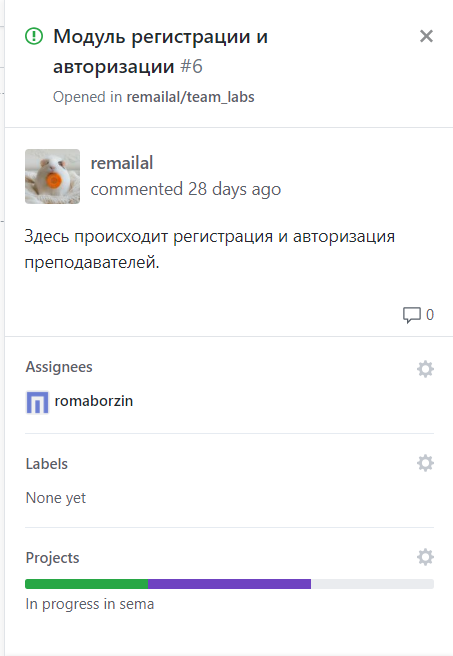
|  |  |
| --- | --- |
| Оценка |  |
|  | (баллы) |

|  |  |  |  |
| --- | --- | --- | --- |
| Проверил |  |  | доцент, Свиридова О.В. |
|  | (подпись и дата подписания) |  | (долж., фамилия и инициалы) |

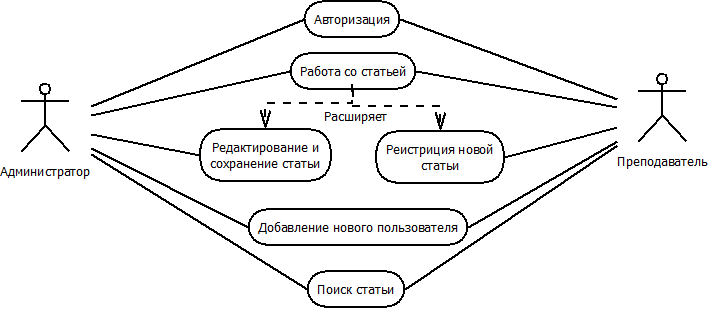
|  |  |  |  |
| --- | --- | --- | --- |
| Специалист по УМР |  |  |  |
|  | (подпись и дата подписания) |  | (фамилия и инициалы) |

Волжский, 2018

**Требования к программному приложению**

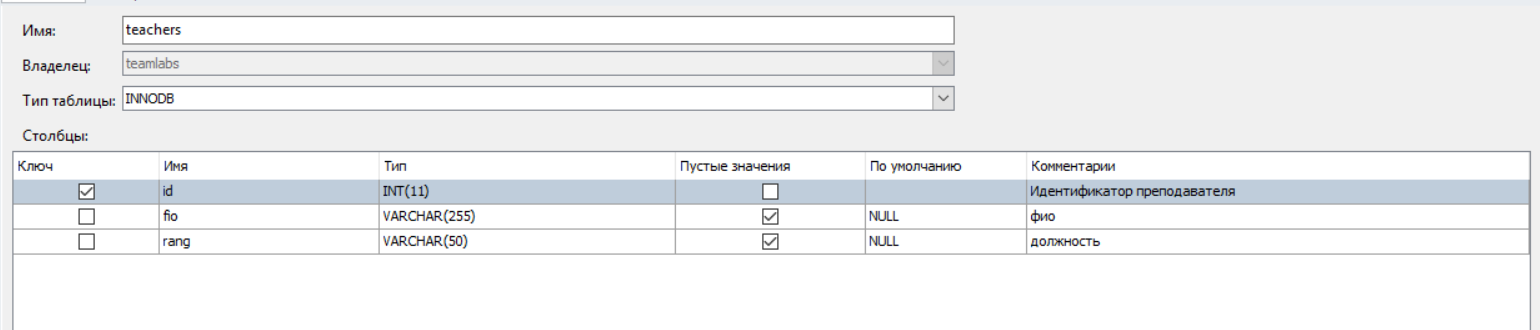


Use case диаграмма вариантов использования системы



**Таблицы базы данных системы**

Таблица преподаватели

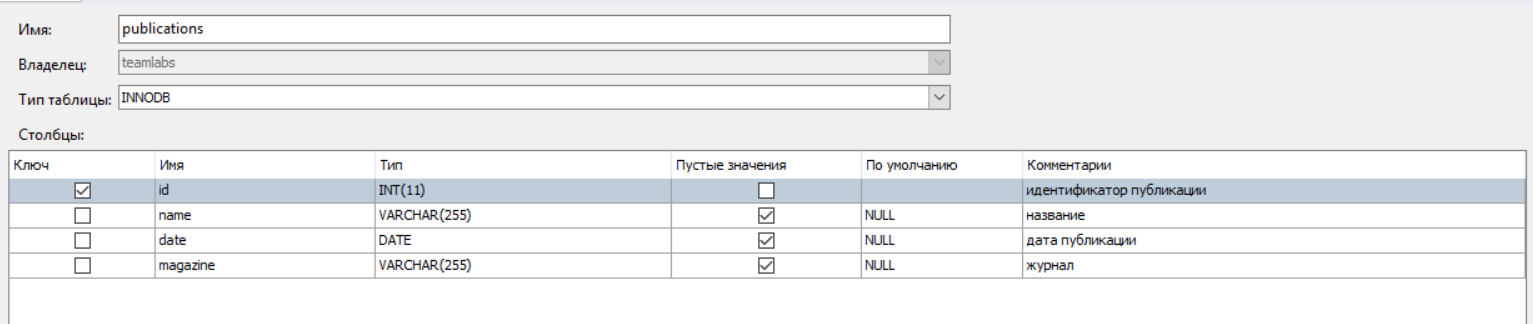


Id – Идентификационный номер преподавателя

Fio – ФИО преподавателя.

Rang – должность препродавателя.

Таблица публикации



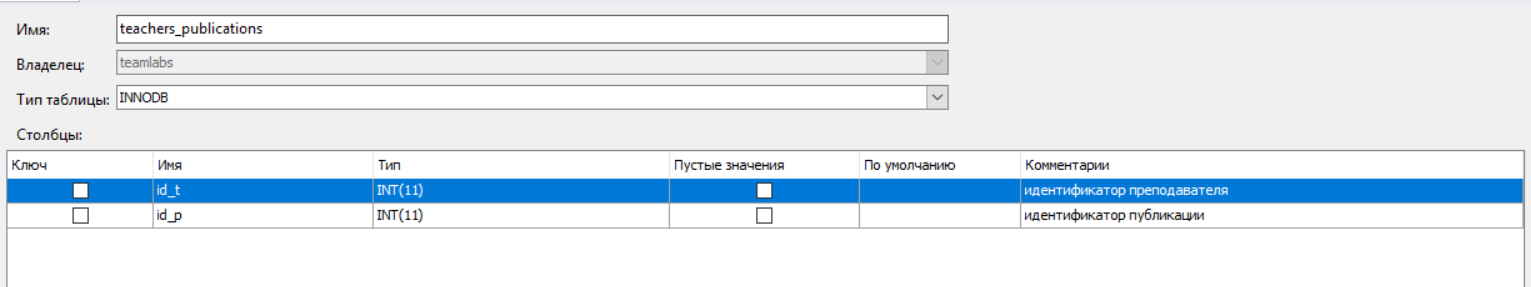
Id – идентификатор публикации.

Name – название.

Date – дата публикации.

Magazine – журнал.

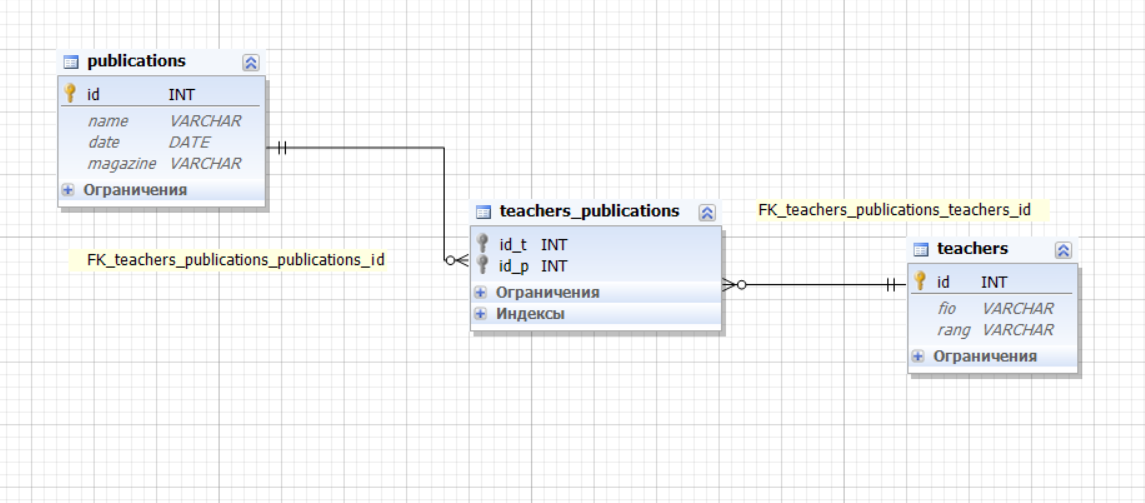
Таблица публикации преподавателей



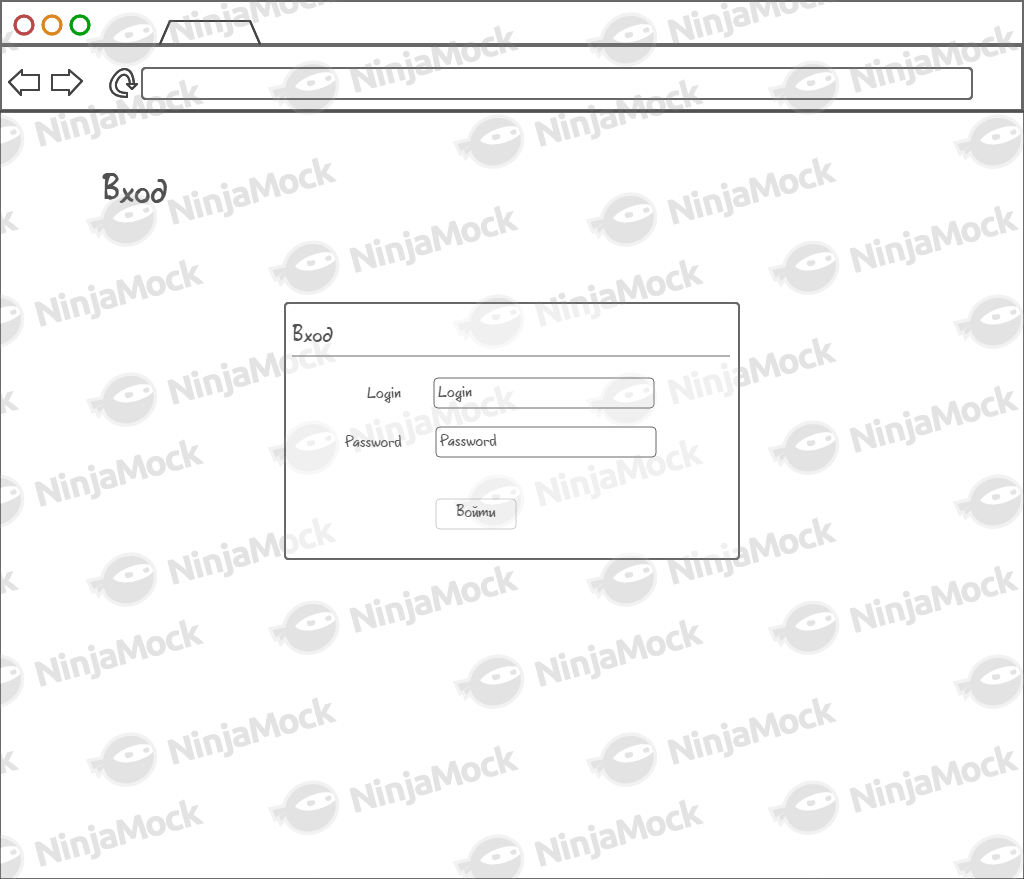
Id\_t – идентификатор преподавателя.

Id\_p – идентификатор публикации.

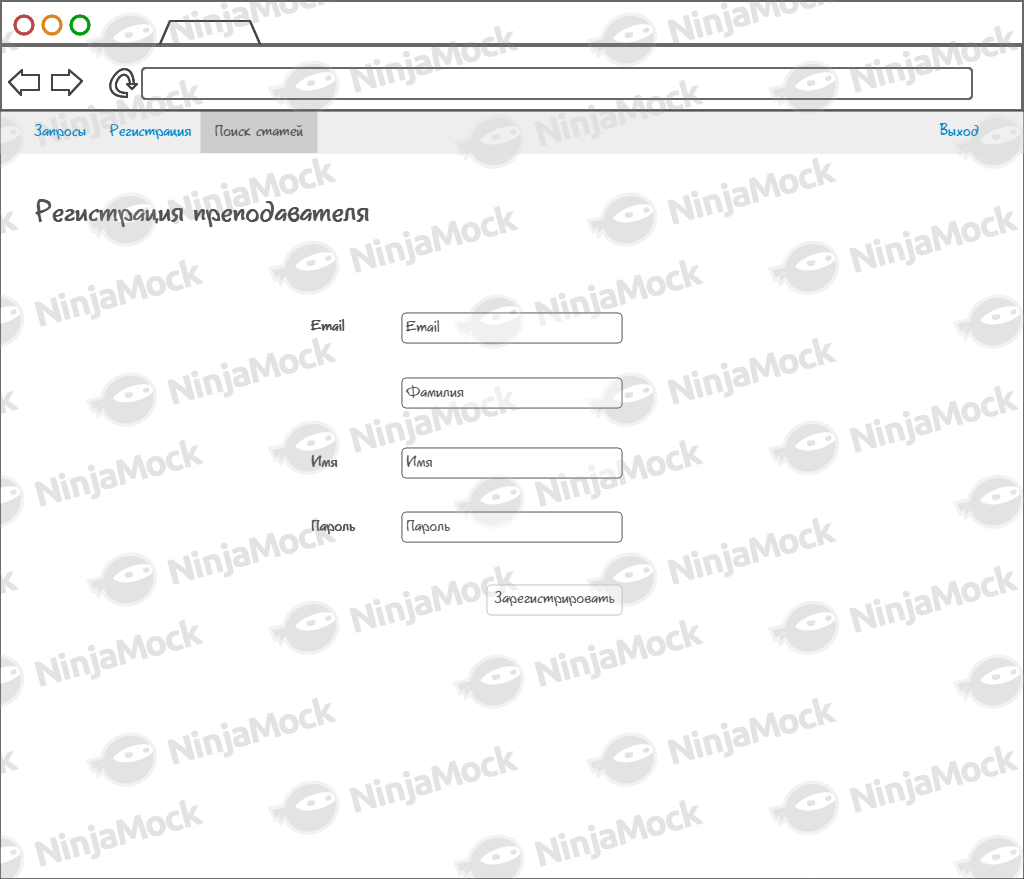
Диаграмма базы данных



**Модель интерфейса системы**



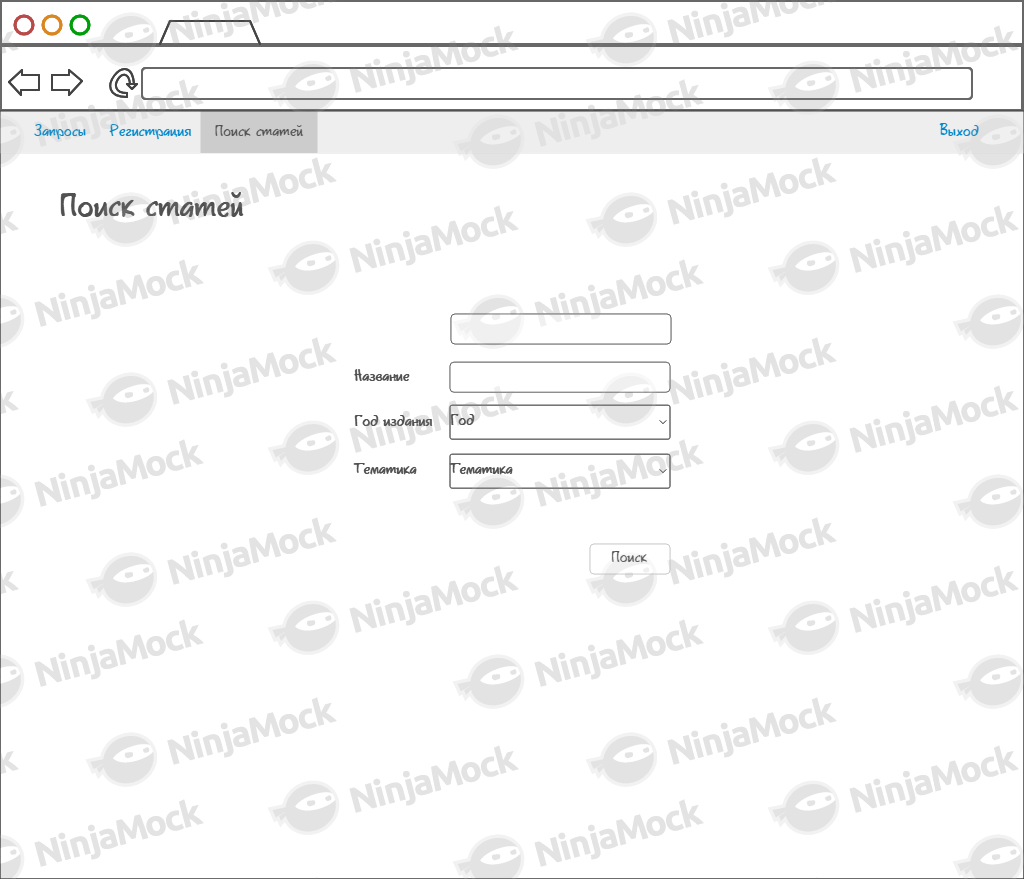
Авторизация в системе



Регистрация преподавателя администратором



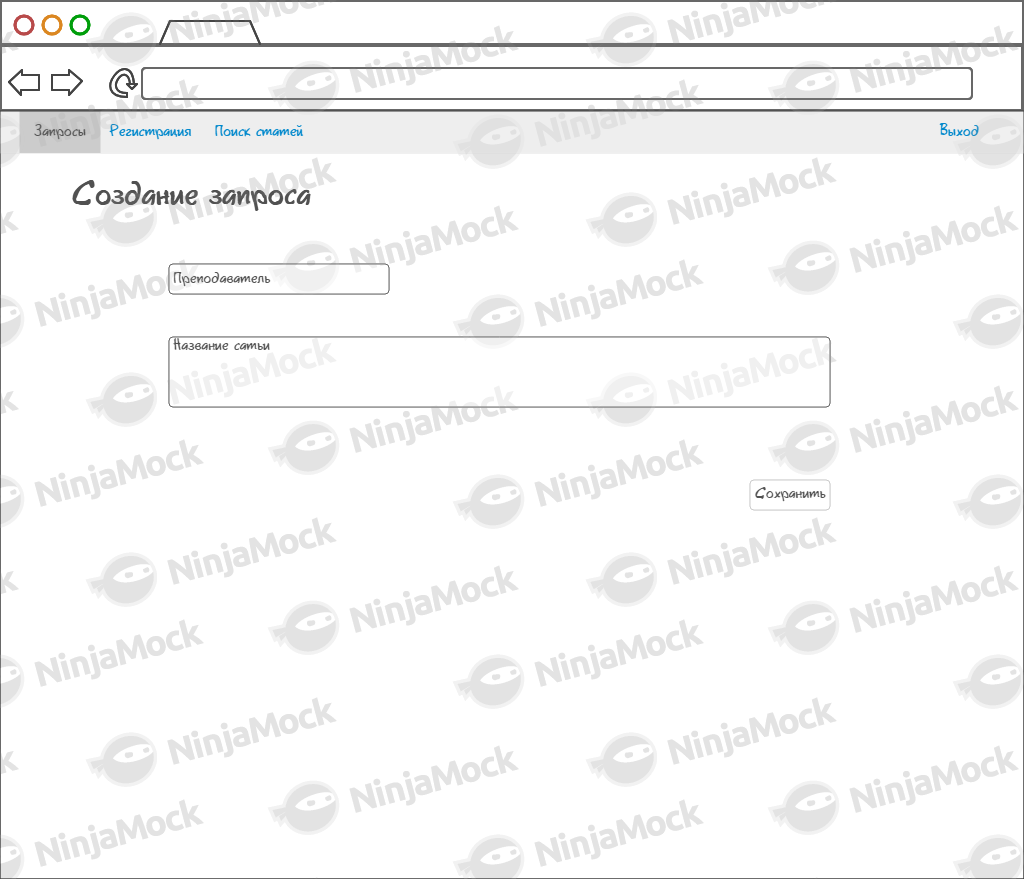
Запросы на размещение статей от преподавателей



Поиск статей

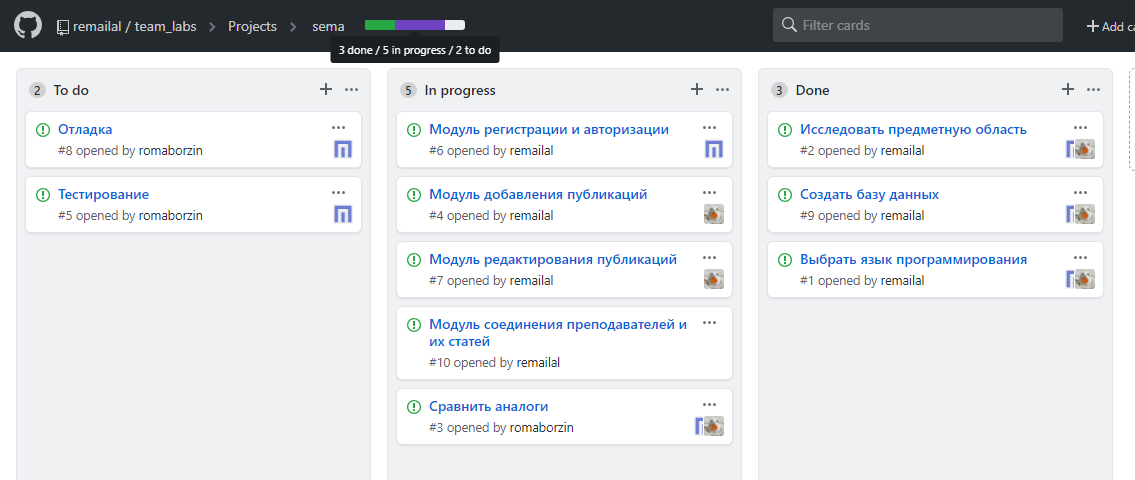


Результаты поиска

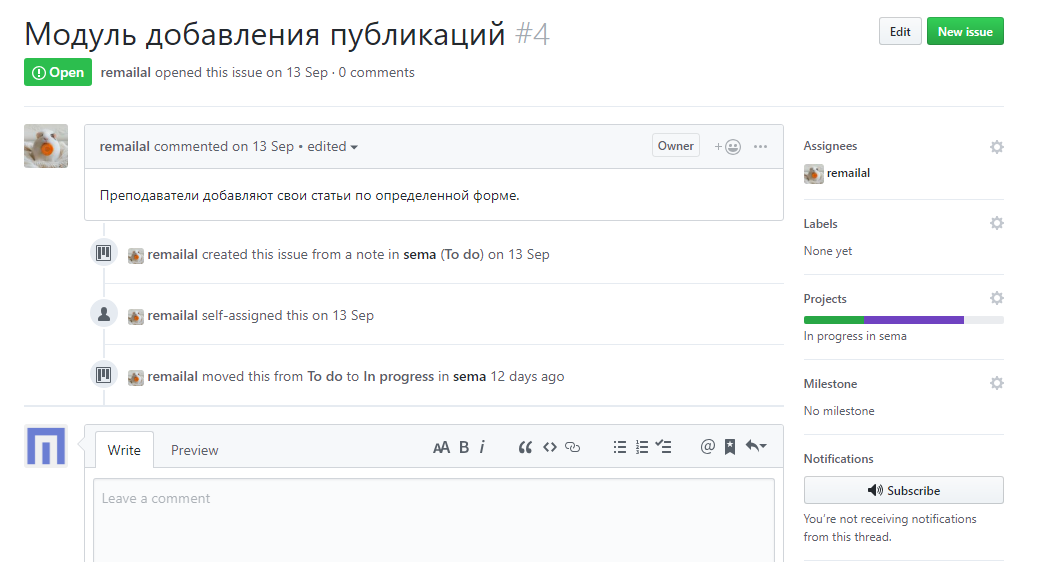


Создание запроса

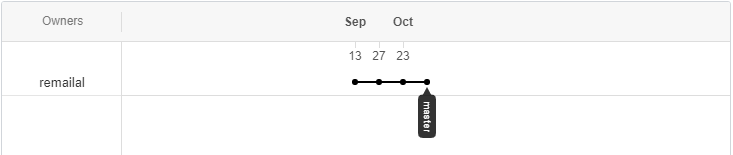
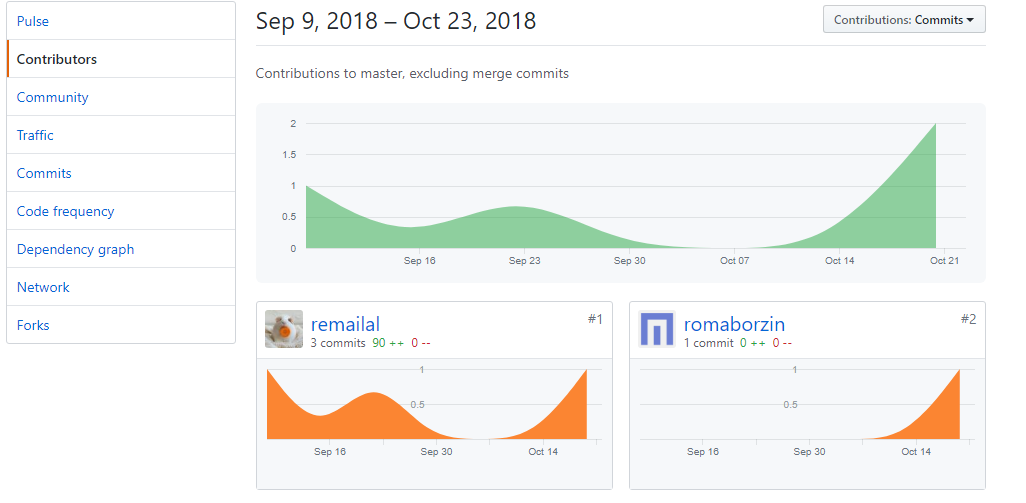
**Планирование времени работы членов команды**



Состояние выполнения работы



Описание модуля с отметками о выполнении



Отслеживание выгрузки промежуточных результатов

**Тестирование разработанного приложения**

Тестирование класса St

Класс St реализует метод вставки addSt(). Данный метод реализуют размещение статьи в системе.

Спецификация данного тестового случая для тестирования метода addSt() на основе спецификации этого класса:

Название класса: addSt

Название тестового случая: add\_st\_test

Описание тестового случая: Тест проверяет правильность работы метода addSt() – добавление статьи на основе введенных пользователем данных.

Ожидаемый результат:

Перечисленным входным значениям должны соответствовать следующие выходные значения:

|  |  |  |  |
| --- | --- | --- | --- |
| Тест 1 | Тест 2 | Тест 3 | Тест 4 |
| Система добавила статью и продолжила работу | Система выдала сообщение: «Вы заполнили не все поля!» | Система выдала сообщение: «Вы заполнили не все поля!» | Система выдала сообщение: «Вы заполнили не все поля!» |

Код теста:

На основе спецификации был создан тестовый драйвер - метод order\_tester.

NSArray \*testCasesText = @[@" Название статьи ", @" Название статьи ", @"", @" Название статьи "];

NSArray \*testCasesZakazchik = @[@"Иванов Иван,", @"", @"Смирнов Дмитрий", @"Сидоров Петр"];

for (int i = 0; i < 4; i++) {

self.taskField.text = [testCasesText objectAtIndex:i];

self.stField.text = [testCasesSt objectAtIndex:i];

[self doneButton:nil];

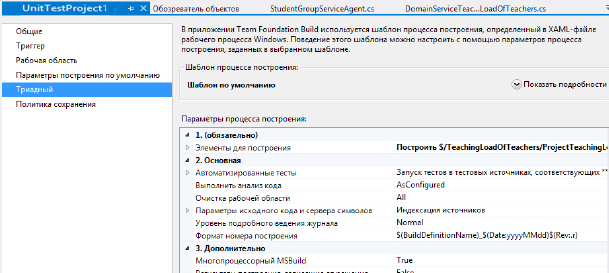
}

В методе st\_tester реализована вся функциональность теста. В данном случае для покрытия спецификации достаточно перебрать следующие комбинации входных данных: все данные введены корректно, ошибка в одной из составляющей входного массива, и получить соответствующие им сообщения с помощью метода add\_st(). Пары соответствующих значений заносятся в NSLog-файл для последующей проверки на соответствие спецификации.

Результаты тестирования формируются классом add\_st\_test, который производит сравнение ожидаемых результатов с фактическими.

После построения тестового проекта созданный тест будет отображаться в обозревателе тестов





Политика сохранения



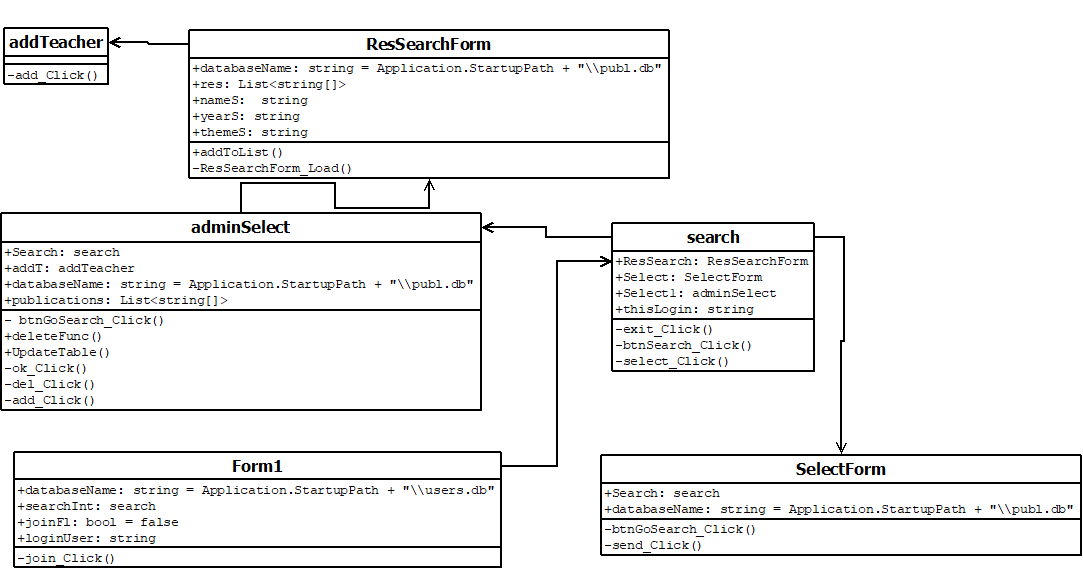
Триадный определения построения

**Руководство системного администратора**

1) Для установки необходимо перенести файл exe на персональный компьютер.

**Руководство системного программиста**

Диаграмма классов



**Исходный код**

**SelectForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Data.SQLite;

using System.Data.Common;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class SelectForm : Form

{

search Search;

string databaseName = Application.StartupPath + "\\publ.db";

public SelectForm()

{

InitializeComponent();

}

private void SelectForm\_Load(object sender, EventArgs e)

{

}

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

{

Search = new search("no admin");

this.Hide();

Search.Show();

}

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

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("INSERT INTO publ (name, year, theme, flag) " +

"values('" + name.Text + "', '" + year.Text + "'," +

"'" + theme.Text + "', 'нет')", connection);

command.ExecuteNonQuery();

connection.Close();

MessageBox.Show("Публикация добавлена!", "Уведомление");

}

}

}

**Search.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Data.SQLite;

using System.Data.Common;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class search : Form

{

ResSearchForm ResSearch;

SelectForm Select;

adminSelect Select1;

string thisLogin;

public search(string login)

{

thisLogin = login;

InitializeComponent();

}

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

{

Application.OpenForms[0].Show();

Application.OpenForms[1].Hide();

}

private void search\_Load(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

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

{

ResSearch = new ResSearchForm(name.Text, year.Text, theme.Text);

this.Hide();

ResSearch.Show();

}

private void search\_FormClosed(object sender, FormClosedEventArgs e)

{

}

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

{

if (thisLogin == "admin")

{

Select1 = new adminSelect();

this.Hide();

Select1.Show();

}

else

{

Select = new SelectForm();

this.Hide();

Select.Show();

}

}

}

}

**ResSearchForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Data.SQLite;

using System.Data.Common;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class ResSearchForm : Form

{

string databaseName = Application.StartupPath + "\\publ.db";

List<string[]> res = new List<string[]>();

string nameS, yearS, themeS;

public ResSearchForm(string N, string Y, string T)

{

nameS = N;

yearS = Y;

themeS = T;

InitializeComponent();

}

void addToList(SQLiteDataReader reader)

{

res.Add(new string[3]);

res[res.Count - 1][0] = reader[1].ToString();

res[res.Count - 1][1] = reader[2].ToString();

res[res.Count - 1][2] = reader[3].ToString();

}

private void ResSearchForm\_Load(object sender, EventArgs e)

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("SELECT \* FROM 'publ';", connection);

SQLiteDataReader reader = command.ExecuteReader();

res.Clear();

foreach (DbDataRecord record in reader)

{

if (nameS != "")

{

if (reader[4].ToString() == "да" && nameS == reader[1].ToString())

{

addToList(reader);

}

}

if (yearS != "")

{

if (reader[4].ToString() == "да" && yearS == reader[2].ToString())

{

addToList(reader);

}

}

if (themeS != "")

{

if (reader[4].ToString() == "да" && themeS == reader[3].ToString())

{

addToList(reader);

}

}

}

if (res.Count != 0)

{

foreach (string[] s in res)

{

if (nameS != "")

{

if (nameS != s[0])

res.Remove(s);

}

if (yearS != "")

{

if (yearS != s[1])

res.Remove(s);

}

if (themeS != "")

{

if (themeS != s[2])

res.Remove(s);

}

}

}

reader.Close();

connection.Close();

foreach (string[] s in res)

resSearchData.Rows.Add(s);

}

}

}

**Form1.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.Common;

using System.Data.SQLite;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class Form1 : Form

{

string databaseName = Application.StartupPath + "\\users.db";

search searchInt;

bool joinFl = false;

string loginUser;

public Form1()

{

InitializeComponent();

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

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

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("SELECT \* FROM 'users';", connection);

SQLiteDataReader reader = command.ExecuteReader();

foreach (DbDataRecord record in reader)

{

if (login.Text == reader[0].ToString() && pas.Text == reader[1].ToString())

{

loginUser = login.Text;

searchInt = new search(loginUser);

this.Hide();

searchInt.Show();

joinFl = true;

}

}

if (joinFl == false)

{

MessageBox.Show("Неверный логин/пароль", "Ошибка входа");

}

reader.Close();

connection.Close();

}

}

}

**AdminSelect.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Data.SQLite;

using System.Data.Common;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class adminSelect : Form

{

search Search;

addTeacher addT;

string databaseName = Application.StartupPath + "\\publ.db";

List<string[]> publications = new List<string[]>();

public adminSelect()

{

InitializeComponent();

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("SELECT \* FROM 'publ';", connection);

SQLiteDataReader reader = command.ExecuteReader();

publications.Clear();

foreach (DbDataRecord record in reader)

{

if (reader[4].ToString() == "нет")

{

publications.Add(new string[4]);

publications[publications.Count - 1][0] = reader[0].ToString();

publications[publications.Count - 1][1] = reader[1].ToString();

publications[publications.Count - 1][2] = reader[2].ToString();

publications[publications.Count - 1][3] = reader[3].ToString();

}

}

reader.Close();

connection.Close();

foreach (string[] s in publications)

resSearchData.Rows.Add(s);

}

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

{

Search = new search("admin");

this.Hide();

Search.Show();

}

public void deleteFunc()

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("DELETE FROM 'publ' WHERE id ='" + resSearchData.CurrentRow.Cells[0].Value + "';", connection);

command.ExecuteNonQuery();

connection.Close();

}

public void UpdateTable()

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("SELECT \* FROM 'publ';", connection);

SQLiteDataReader reader = command.ExecuteReader();

publications.Clear();

foreach (DbDataRecord record in reader)

{

if (reader[4].ToString() == "нет")

{

publications.Add(new string[4]);

publications[publications.Count - 1][0] = reader[0].ToString();

publications[publications.Count - 1][1] = reader[1].ToString();

publications[publications.Count - 1][2] = reader[2].ToString();

publications[publications.Count - 1][3] = reader[3].ToString();

}

}

reader.Close();

connection.Close();

foreach (string[] s in publications)

resSearchData.Rows.Add(s);

}

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

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

foreach (string[] s in publications)

{

SQLiteCommand command = new SQLiteCommand("INSERT INTO publ (name, year, theme, flag) " +

"values('" + resSearchData.CurrentRow.Cells[1].Value + "', '" + resSearchData.CurrentRow.Cells[2].Value + "'," +

"'" + resSearchData.CurrentRow.Cells[3].Value + "', 'да')", connection);

command.ExecuteNonQuery();

}

connection.Close();

deleteFunc();

resSearchData.Rows.Clear();

UpdateTable();

MessageBox.Show("Публикация добавлена!", "Уведомление");

}

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

{

deleteFunc();

resSearchData.Rows.Clear();

UpdateTable();

}

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

{

addT = new addTeacher();

this.Hide();

addT.Show();

}

}

}

**addTeacher.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Data.SQLite;

using System.Data.Common;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Командная\_разработка\_сема

{

public partial class addTeacher : Form

{

string databaseName = Application.StartupPath + "\\users.db";

public addTeacher()

{

InitializeComponent();

}

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

{

SQLiteConnection connection =

new SQLiteConnection(string.Format("Data Source={0};", databaseName));

connection.Open();

SQLiteCommand command = new SQLiteCommand("INSERT INTO users (login, pas) " +

"values('" + log.Text + "', '" + pas.Text + "')", connection);

command.ExecuteNonQuery();

connection.Close();

MessageBox.Show("Преподаватель добавлен!", "Уведомление");

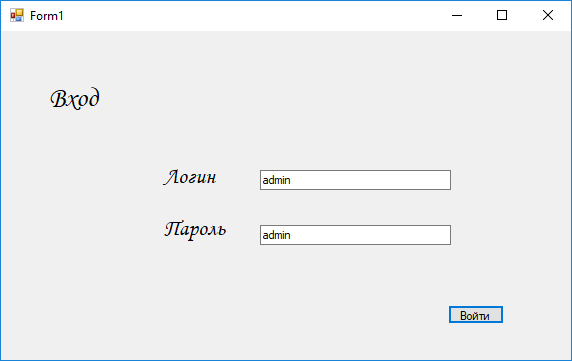
}

}

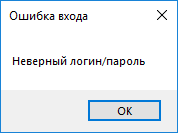
}

**Руководство пользователя**

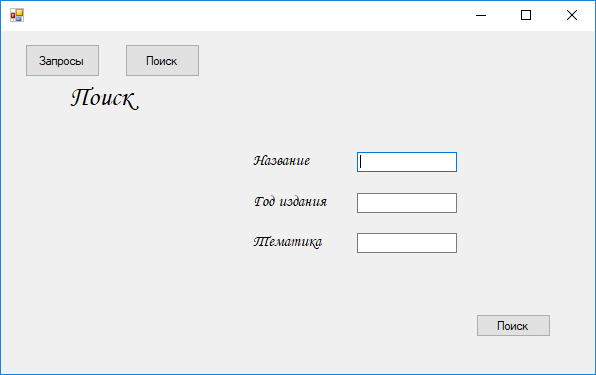
При запуске программы открывается окно авторизации пользователя



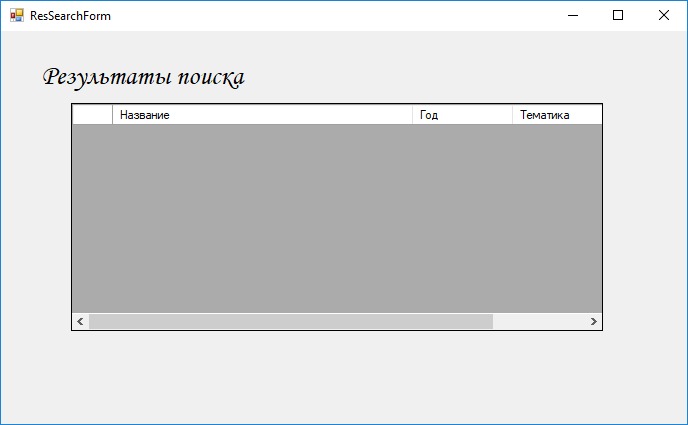
При вводе неверных данных для входа будет выведено сообщение об ошибке



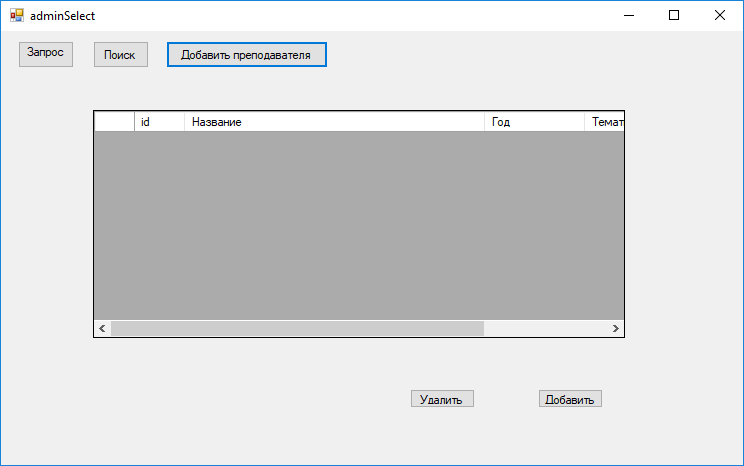
На экране администратора отображается окно поиска статей.



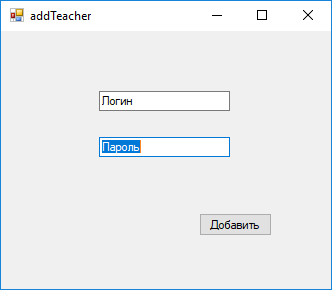
По заданным параметрам осуществляется поиск статей



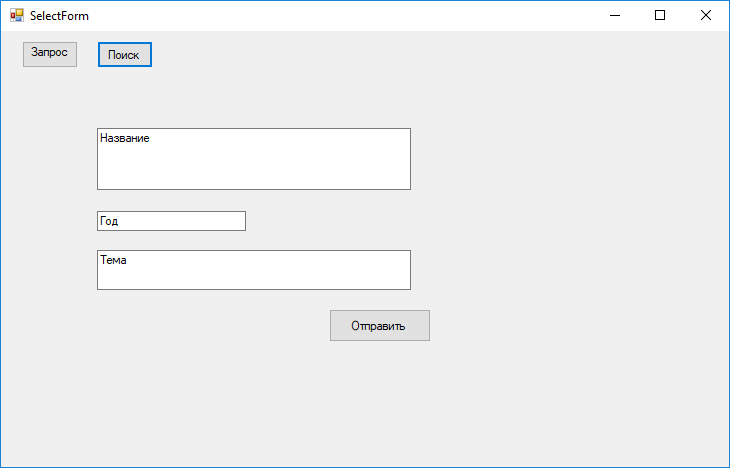
Администратору доступно редактирование и сохранение статей, отправленных преподавателем на модерацию.



Администратор может создавать пользователей.



Преподаватель после авторизации может отправить статью на модерацию.



Так же преподавателю доступен поис публикаций

