



**МИНОБРНАУКИ РОССИИ
ФГБОУ ВО
РТУ МИРЭА
Колледж программирования и кибербезопасности**

**Отчет №5
«Задание УП»**

По УП.01.01 Учебная практика

Специальность 09.02.07 Информационные системы и программирование

Выполнил студент
Группы ПКС-35
Неструев О. Д.
Проверил преподаватель
Стоколос М. Д.

Москва
2024

Цель: Редактирование окна регистрации пользователя, создание класса для обработки исключительных ситуаций, авторизация пользователя в базе данных.

1. Редактирование окна регистрации пользователя

Для редактирования окна необходимо добавить в него поля для ввода «ФИО» и «Пароль» пользователя, а также убрать поле для ввода «Почта» пользователя. Пример нового окна регистрации представлен на рисунке 1.

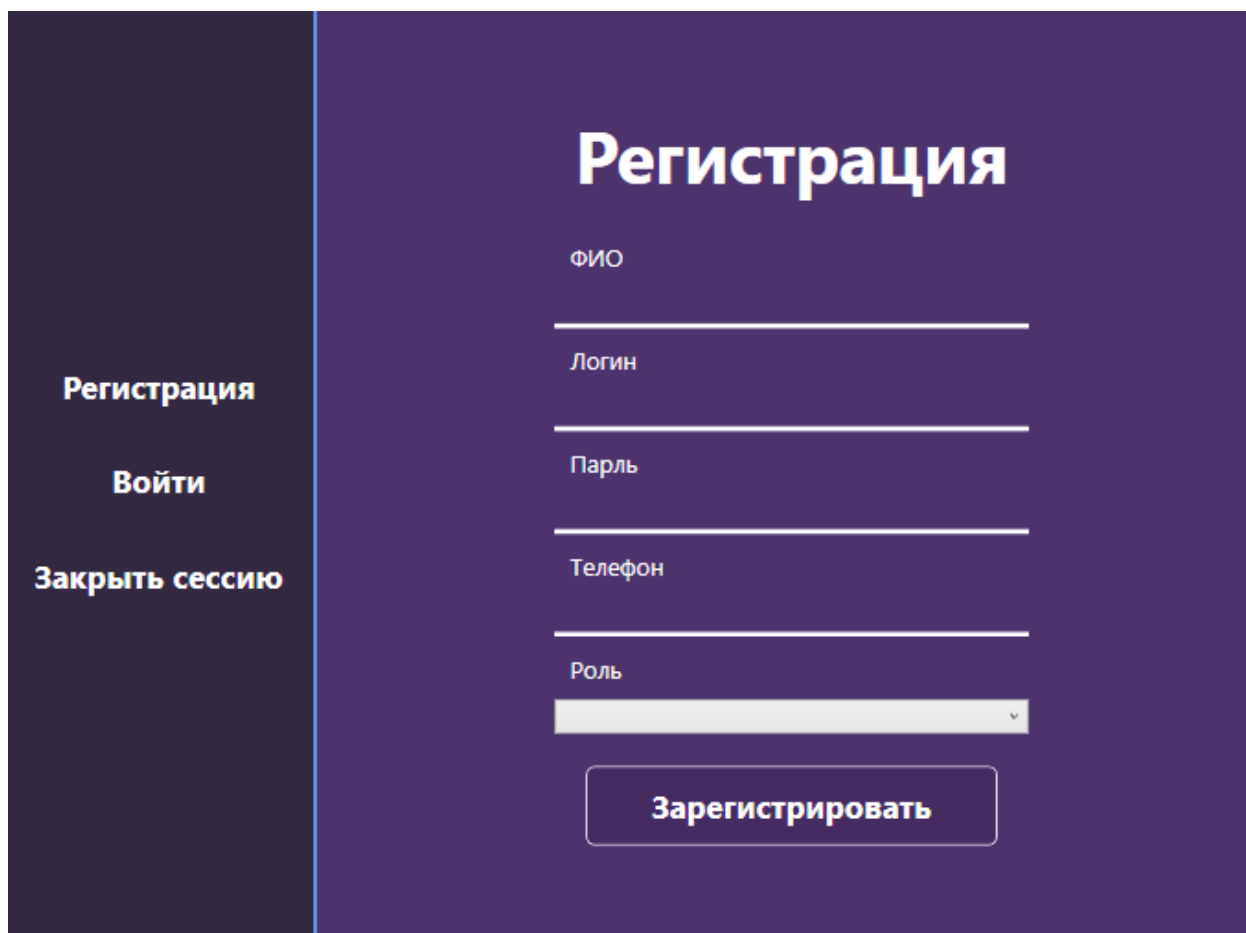
The image shows a web interface for user registration. On the left is a dark purple sidebar with three white text links: 'Регистрация', 'Войти', and 'Заккрыть сессию'. The main area has a purple background with the title 'Регистрация' in large white font. Below the title are five input fields with white labels: 'ФИО', 'Логин', 'Парль', 'Телефон', and 'Роль'. The 'Роль' field is a dropdown menu. At the bottom is a white button with a purple border and the text 'Зарегистрировать' in purple.

Рис. 1 Измененное окно регистрации пользователя

2. Создание класса для обработки исключительных ситуаций

Для обработки исключительных ситуаций были созданы два класса:

- «Instruments.cs»;
- «CheckInputData.cs».

Класс «Instruments.cs» хранит в себе паттерны создания сообщений для пользователя в случае нарушения работы или уточнения действий. Пример скрипта для создания сообщения представлен ниже.

```
// Вывод информационного сообщения
public object createInformationMessageBox(string message) =>
MessageBox.Show(message, "Tech Support", MessageBoxButtons.OK,
MessageBoxImage.Information);
// Вывод уточняющих сообщений
public bool createQuestionMessageBox(string message) =>
(MessageBox.Show(message, "Tech Support",
MessageBoxButton.YesNo, MessageBoxImage.Question) ==
MessageBoxResult.Yes);
```

Класс «CheckInputData.cs» проверяет вводимые пользователем данные на корректность и наличие требуемой информации. Пример скрипта представлен ниже.

```
// Защита от SQLI
public bool checkSQLI(string example)
{
if (example.Length == 0)
return false;
string[] incorrectSymbols = new string[3] { "", "-", ";" };
for(int symbolIndex = 0; symbolIndex < 3; symbolIndex++)
{
if (example.Contains(incorrectSymbols[symbolIndex]))
return false;
}
return true;
}

// Проверка номера телефона
public bool controlTelephoneNumbers(string telephoneNumberData)
{
Regex outPlusPhoneNumberController = new Regex(
"[8][0-9]{10}");
```

```

Console.WriteLine("Results 1: " +
outPlusPhoneNumberController.IsMatch(telephoneNumberData));
return
outPlusPhoneNumberController.IsMatch(telephoneNumberData);
}

```

3. Авторизация пользователя в базе данных

Для авторизации пользователя в базе данных используется класс «DatabaseConnection.cs», который хранит в себе скрипты для работы с базой данных и алгоритмы подключения. За непосредственную авторизацию пользователя отвечает метод «signInAccount()» который делает запрос в таблицу пользователя и ищет введенные им логин и пароль. Пример скрипта авторизации представлен ниже.

```

// Авторизация пользователя
public bool signInAccount(string[] userData, string userRole,
string idName)
{
string login = userData[0],
password = userData[1];
using (SqlConnection conn = new SqlConnection(url))
{
conn.Open();
using (SqlCommand cmd = new SqlCommand())
{
cmd.Connection = conn;
cmd.CommandText = $"
Use RepairService
SELECT [{idName}], Логин, Пароль
FROM {userRole};
";
SqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
if (login.Trim() == (" + dr["Логин"]).Trim() &&

```

```
password.Trim() == ("" + dr["Пароль"]).Trim())
{
    CacheWork cacheWork = new CacheWork();
    // Регистрация пользователя в КЕШ
    cacheWork.AddUserID(Convert.ToUInt64(("" + dr[idName]).Trim()));
    cacheWork.AddUserRole(userRole);
    return true;
}
}
dr.Close();
}
conn.Close();
}

return false;
}
```