Цель: Освоение методики тестирования разработанной программной системы в ручном режиме.

Были протестированы следующие функции системы:

* Добавление, изменение и удаление работников
* Добавление, изменение и удаление рецепта
* Добавление, изменение и удаление пациентов
* Добавление, изменение и удаление счета
* Добавление, изменение и удаление заказа
* Создание записи в журнал через Excel

В результате ручного тестирования были найдены следующие ошибки: 1. Отсутствие наличие ограничений в полях Даты рождения у Работника(worker), Пациента (patient). См. Рисунок 1

Последовательность для обнаружения: выбрать текущую дату в качестве значения. Внешнее проявление ошибки отсутствует. Способ ее устранения: введение триггеров, содержащих ограничение.

2. Отсутствие проверки логина и пароля у работника(worker). См. Рисунок 2

Последовательность для обнаружения: ввод полей login и parol. Внешнее проявление ошибки отсутствует. Способ ее устранения: введение БД, содержащее данные всех логинов и паролей различных работников, а также триггеров, осуществляющих проверку на наличие введенной информации.

3. Проведение записи в журнал только одного клиента в связи с записью только одной выбранной даты. См. Рисунок 3

Последовательность для обнаружения: выбор нескольких клиентов и ввод одной даты. Внешнее проявление ошибки: наличие одинаковых полей даты в excel для различных клиентов. Способ ее устранения: введение отдельной вкладки с полями для записи в журнал, после чего происходило бы формирование отчета для различных клиентов.

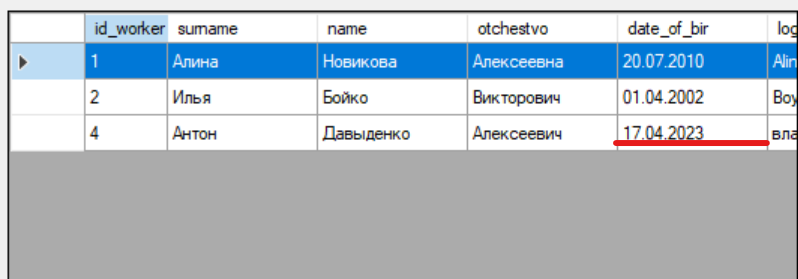


Рисунок 1 – Ошибка ввода даты

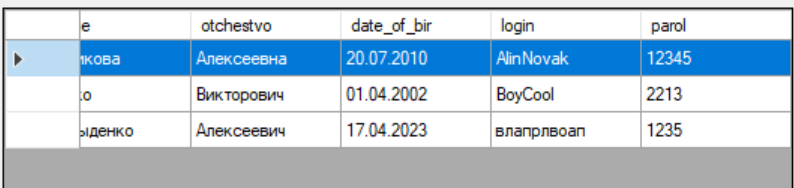


Рисунок 2 – Ошибка проверки login, parol

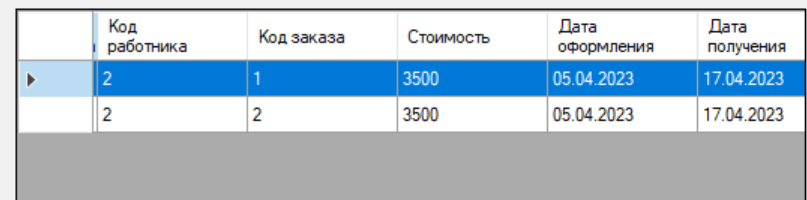


Рисунок 3 – Ошибка одинаковых дат

Технические характеристики машины:

* Процессор – 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
* Оперативная память – 8,00 ГБ
* Жесткий диск, на котором проводилось тестирование – HDD, 250 ГБ

Было проведено нагрузочное тестирование на следующих объемах случайных данных:

N – работников, N – пациентов, N – рецептов, N – счетов, N – заказов

Каждый из N работников записывает N пациентов с вероятностью 0.5 (в среднем N^2/2);

Каждый из N пациентов производят N заказов (в среднем 2\*N^2);

Каждый из N счетов включают в себя препараты из N рецептов с (в среднем 5\*N^2/2);

Для значений N= 10, 50, 100, 250, 700;

Были произведены замеры времени загрузки данных в БД следующих таблиц:

* Список всех докторов
* Список всех пациентов
* Список всех работников

По результатам нагрузочного тестирования, заметно меняется лишь время загрузки информации в БД работников, препаратов и записи в журнал. На Таблице 1 и Рисунке 4 приведены результаты.

Таблица 1 – Результаты нагрузочного тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| БД | N = 10 | N = 50 | N = 100 | N = 250 | N = 700 |
| worker | 0,029 | 0,037 | 0,019 | 0,049 | 0,044 |
| preparate | 0,008 | 0,04 | 0,027 | 0,111 | 0,2234 |
| write\_mag | 0,079 | 0,323 | 0,652 | 1,123 | 9,637 |

Рисунок 4 – Зависимость времени выполнения запроса от N

На Рисунке 5 представлен график зависимости затраченного времени от количества записей при отсутствии сторонней нагрузки на оборудование.

Рисунок 5 – Зависимость времени выполнения запроса от N (log)

Стрессовое тестирование проводилось на случайных данных, сгенерированных N = 1000, после чего было замерено время ответа БД каждой из вкладок программы. Результаты измерений таблиц приведены на Таблице 2.

|  |  |
| --- | --- |
| Вкладка | Среднее время загрузки (секунды) |
| worker | 0,015856 |
| recipe | 0,12133 |
| patient | 1,0239 |
| chet | 0,9384 |
| zakaz | 0,00242 |
| doctor | 0,57894 |
| preparate | 1,49023 |
| state\_zakaz | 1,84832 |
| write\_mag | 0,4576 |
| naklad | 0,5234 |

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 5 – Нагрузка ноутбука при помощи Furmark

Пустой проект Unit test:

[Test]  
public void TestPatientInsertButton\_Click()  
{  
    // Arrange  
    var form = new Form();  
    var namePatBox = new TextBox() { Text = "John" };  
    var surnamePatBox = new TextBox() { Text = "Doe" };  
    var otchestvoPatBox = new TextBox() { Text = "Smith" };  
    var bDateP = new DateTimePicker() { Value = new DateTime(1990, 1, 1) };  
    var idRecipeBox = new TextBox() { Text = "1" };  
    var lgotaBox = new TextBox() { Text = "Yes" };  
    form.Controls.Add(namePatBox);  
    form.Controls.Add(surnamePatBox);  
    form.Controls.Add(otchestvoPatBox);  
    form.Controls.Add(bDateP);  
    form.Controls.Add(idRecipeBox);  
    form.Controls.Add(lgotaBox);  
  
    var patientInsertButton = new Button();  
    var eventArgs = new EventArgs();  
    bool messageBoxShown = false;  
  
    // Act  
    patientInsertButton\_Click(patientInsertButton, eventArgs);  
  
    // Assert  
    if (string.IsNullOrEmpty(namePatBox.Text.Trim())  
        || string.IsNullOrEmpty(surnamePatBox.Text.Trim())  
        || string.IsNullOrEmpty(otchestvoPatBox.Text.Trim())  
        || string.IsNullOrEmpty(lgotaBox.Text.Trim()))  
    {  
        messageBoxShown = true;  
        Assert.IsTrue(messageBoxShown);  
    }  
    else if (string.IsNullOrEmpty(idRecipeBox.Text))  
    {  
        messageBoxShown = true;  
        Assert.IsTrue(messageBoxShown);  
    }  
}

Вывод: освоили методику тестирования разработанной программной системы в ручном режиме.