|  |  |
| --- | --- |
| ***Название проекта*** | Медицинская лаборатория |
| ***Номер версии*** | №1 |
| ***Имя тестера*** | Максим |
| ***Даты тестирования*** | 16.05.2024 |

*Тестирование приложения (Медицинская лаборатория)*

*Тестирование приложения с использованием TestCase*

|  |  |
| --- | --- |
| ***TestCase #*** | 1 |
| ***Приоритет теста*** | Высокий |
| ***Название тестирования/Имя*** | Авторизация пользователя |
| ***Резюме испытания*** | Проверить процесс авторизации пользователя в приложении медицинская лаборатория. |
| ***Шаги тестирования*** | 1. Открыть приложение медицинская лаборатория. 2. Нажать кнопку «Войти» в верхней части экрана. 3. Ввести имя пользователя и пароль. 4. Нажать кнопку «Войти». |
| ***Данные тестирования*** | * имя пользователя: test\_user * пароль: password |
| ***Ожидаемый результат*** | Пользователь успешно авторизуется в системе. |
| ***Фактический результат*** | Пользователь успешно авторизуется в системе. |
| ***Предпосылки*** | Пользователь ранее зарегистрировался в приложении медицинская лаборатория. |
| ***Постусловия*** | Пользователь успешно авторизован в системе. |
| ***Статус (Pass/Fail)*** | Pass |
| ***Комментарии*** | Тест пройден успешно, ожидаемый результат достигнут. |

Визуальное тестирование приложения

|  |  |  |
| --- | --- | --- |
| ***Подготовка к тестированию*** | Убедитесь, что приложение установлено и запущено на устройстве, на котором будет проводиться тестирование. |  |
| ***Проверка главной страницы*** | Откройте приложение и проверьте, что все основные элементы, такие как заголовок, поиск, кнопка добавления контакта, список контактов, отображаются корректно и имеют соответствующий дизайн. |  |
| ***Проверка добавления пациента*** | Нажмите на кнопку добавления пациента и заполните все поля формы. Убедитесь, что все поля имеют правильные метки, их содержимое сохраняется и новый пациент появляется в списке. |  |
| ***Проверка редактирования пациента*** | Выберите пациента из списка и нажмите на кнопку редактирования. Измените информацию и сохраните. Убедитесь, что изменения сохранены, и пациент обновлен в списке. |  |
| ***Проверка удаления пациента*** | Выберите пациента и нажмите на кнопку удаления. Подтвердите удаление и убедитесь, что пациент удален из списка. |  |
| ***Проверка работы приложения на разных устройствах и в разных разрешениях экрана*** | Убедитесь, что приложение корректно отображается и функционирует на разных устройствах с разными размерами экрана и ориентацией. |  |
| ***Проверка состояния загрузки*** | Убедитесь, что при загрузке или обновлении данных приложение отображает соответствующую индикацию, сообщение о загрузке. |  |
| ***Проверка состояния ошибки*** | Попытайтесь вызвать ошибку, например, удалив всех пациентов или введя некорректные данные. |  |
| ***Проверка работы с пробелами и специальными символами*** | Введите пробелы и специальные символы в поля формы и убедитесь, что приложение корректно обрабатывает такие данные. |  |
| ***Вывод*** | Вывод по работе приложения | Приложение работает успешно. Без видимых ошибок и вылетов. |

*Тестирование с помощью UnitTest*

**Код:**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using Moq;

namespace UnitTestProject1

{

[TestClass]

public class UnitTest1

{

private Mock<ISampleService> \_mockSampleService;

private SampleManagementViewModel \_sampleManagementViewModel;

[TestMethod]

public void TestMethod1()

{

\_mockSampleService = new Mock<ISampleService>();

\_sampleManagementViewModel = new SampleManagementViewModel(\_mockSampleService.Object);

}

public void AddSample\_WithValidData\_AddsSampleToList()

{

// Arrange

var sample = new Sample { Id = 1, Name = "Test Sample" };

\_mockSampleService.Setup(s => s.AddSample(sample));

// Assert

Assert.AreEqual(1, \_sampleManagementViewModel);

Assert.AreEqual(sample, \_sampleManagementViewModel);

}

public void RemoveSample\_WithExistingSample\_RemovesSampleFromList()

{

// Arrange

var sample = new Sample { Id = 1, Name = "Test Sample" };

\_mockSampleService.Setup(s => s.GetSampleById(1));

// Act

\_sampleManagementViewModel.SelectedSample = sample;

\_sampleManagementViewModel.RemoveSampleCommand(null);

// Assert

Assert.AreEqual(0, \_sampleManagementViewModel);

}

}

internal class Sample

{

public int Id { get; set; }

public string Name { get; set; }

}

internal interface ISampleService

{

void AddSample(Sample sample);

void GetSampleById(int v);

}

internal class SampleManagementViewModel

{

internal object AddSampleCommand;

internal object Samples;

private ISampleService @object;

public SampleManagementViewModel(ISampleService @object)

{

this.@object = @object;

}

public Sample SelectedSample { get; internal set; }

internal void RemoveSampleCommand(object value)

{

throw new NotImplementedException();

}

}

}

***Результат:***

