Аннотация теста:

|  |  |
| --- | --- |
| **Название проекта** | Информационная система управления заявками ремонта автомобилей |
| **Рабочая версия** | 0.1 |
| **Имя тестирующего** | Рудыка Елизавета Александровна |
| **Дата(ы) теста** | 17.10.2024 |

Тест кейс #1:

|  |  |
| --- | --- |
| **Тест кейс #** | 1 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Проверка авторизации с корректными данными |
| **Краткое изложение теста** | Тест проверяет успешный вход пользователя в систему с корректными логином и паролем |
| **Этапы теста** | 1. Запустить приложение 2. Ввести корректные логин и пароль 3. Нажать кнопку "Войти" |
| **Тестовые данные** | Логин: login15  Пароль: pass15 |
| **Ожидаемый результат** | Появляется сообщение об успешной авторизации и открывается окно автомеханика. |
| **Фактический результат** | Появляется сообщение об успешной авторизации и открывается окно автомеханика. |
| **Статус** | Зачет |
| **Предварительное условие** | Логин и пароль пользователя внесены в базу данных |
| **Постусловие** | Пользователю доступен функционал роли автомеханика |
| **Примечания/комментарии** | Нет |

Тест кейс #2:

|  |  |
| --- | --- |
| **Тест кейс #** | 2 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Проверка авторизации с некорректными данными |
| **Краткое изложение теста** | Тест проверяет реакцию системы при вводе неверного логина или пароля |
| **Этапы теста** | 1. Запустить приложение 2. Ввести некорректный логин или пароль 3. Нажать кнопку "Войти" |
| **Тестовые данные** | Логин: login15  Пароль: password15 |
| **Ожидаемый результат** | Появляется сообщение об ошибке "Неверный логин или пароль" и капча. |
| **Фактический результат** | Появляется сообщение об ошибке "Неверный логин или пароль" и капча. |
| **Статус** | Зачет |
| **Предварительное условие** | Логин и пароль пользователя не внесены в базу данных |
| **Постусловие** | Авторизация не выполнена, сообщение об ошибке показано |
| **Примечания/комментарии** | Нет |

Тест кейс #3:

|  |  |
| --- | --- |
| **Тест кейс #** | 3 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Проверка авторизации после блокировки аккаунта |
| **Краткое изложение теста** | Тест проверяет возможность входа после блокировки пользователя из-за многократных неудачных попыток авторизации |
| **Этапы теста** | 1. Запустить приложение 2. Ввести неверные данные 3 раза 3. Нажать кнопку "Войти" 4. Проверить блокировку аккаунта |
| **Тестовые данные** | Логин: login15  Пароль: password15 |
| **Ожидаемый результат** | Появляется сообщение об ошибке "Неверный логин или пароль" и сообщение о блокировки системы на 3 минуты. |
| **Фактический результат** | Появляется сообщение об ошибке "Неверный логин или пароль" и сообщение о блокировки системы на 3 минуты. |
| **Статус** | Зачет |
| **Предварительное условие** | Логин и пароль пользователя не внесены в базу данных |
| **Постусловие** | Вход заблокирован |
| **Примечания/комментарии** | Нет |

Тест кейс #4:

|  |  |
| --- | --- |
| **Тест кейс #** | 4 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Проверка функциональности создания заявки |
| **Краткое изложение теста** | Тест проверяет корректность работы формы создания заявки |
| **Этапы теста** | 1. Открыть вкладку «Добавление заявки» 2. Заполнить данные заявки 3. Нажать кнопку «Создать» |
| **Тестовые данные** | Вид автомобиля: Легковой  Модель автомобиля: Toyota Mark  Проблема: Руль плохо крутится  ФИО: Касаткин Егор Львович  Телефон: 89991234567 |
| **Ожидаемый результат** | Заявка успешно создана, сообщение о добавлении заявки, обновление списка заявок |
| **Фактический результат** | Заявка успешно создана, сообщение о добавлении заявки, обновление списка заявок |
| **Статус** | Зачет |
| **Предварительное условие** | Пользователь должен иметь доступ к созданию заявок |
| **Постусловие** | Заявка добавлена в базу данных |
| **Примечания/комментарии** | Нет |

Тест кейс #5:

|  |  |
| --- | --- |
| **Тест кейс #** | 5 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Проверка удаления заявки заказчиком |
| **Краткое изложение теста** | Тест проверяет возможность удаления заявки заказчиком |
| **Этапы теста** | 1. Открыть список заявок 2. Выбрать заявку 3. Нажать кнопку "Удалить" |
| **Тестовые данные** | Нет |
| **Ожидаемый результат** | Заявка успешно удалена, сообщение о завершении удаления, обновление списка заявок |
| **Фактический результат** | Заявка успешно удалена, сообщение о завершении удаления, обновление списка заявок |
| **Статус** | Зачет |
| **Предварительное условие** | Вход в систему под ролью заказчика, в системе есть заявки вошедшего заказчика |
| **Постусловие** | Заявка удалена из базы данных |
| **Примечания/комментарии** | Заявка удалена |

Библиотека классов:

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Пройденные юнит-тесты:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

using Microsoft.VisualStudio.TestTools.UnitTesting;

using InventoryLibrary;

using System;

using System.Collections.Generic;

namespace InventoryTests

{

[TestClass]

public class InventoryManagerTests

{

private InventoryManager inventoryManager;

[TestInitialize]

public void Setup()

{

inventoryManager = new InventoryManager();

}

[TestMethod]

public void AddItem\_ValidItem\_AddsItemToInventory()

{

var item = new Item(1, "Laptop", 10, 999.99m);

inventoryManager.AddItem(item);

var items = inventoryManager.GetAllItems();

Assert.AreEqual(1, items.Count);

Assert.AreEqual("Laptop", items[0].Name);

}

[TestMethod]

[ExpectedException(typeof(InvalidOperationException))]

public void AddItem\_DuplicateItem\_ThrowsException()

{

var item1 = new Item(1, "Laptop", 10, 999.99m);

var item2 = new Item(1, "Tablet", 5, 499.99m);

inventoryManager.AddItem(item1);

inventoryManager.AddItem(item2);

}

[TestMethod]

public void RemoveItem\_ValidItem\_RemovesItemFromInventory()

{

var item = new Item(1, "Laptop", 10, 999.99m);

inventoryManager.AddItem(item);

inventoryManager.RemoveItem(1);

var items = inventoryManager.GetAllItems();

Assert.AreEqual(0, items.Count);

}

[TestMethod]

[ExpectedException(typeof(KeyNotFoundException))]

public void RemoveItem\_NonExistingItem\_ThrowsException()

{

inventoryManager.RemoveItem(1);

}

[TestMethod]

public void UpdateQuantity\_ValidItem\_UpdatesItemQuantity()

{

var item = new Item(1, "Laptop", 10, 999.99m);

inventoryManager.AddItem(item);

inventoryManager.UpdateQuantity(1, 20);

var updatedItem = inventoryManager.GetItemById(1);

Assert.AreEqual(20, updatedItem.Quantity);

}

}

}