**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Отчет о выполнении лабораторной работы №7**

**по дисциплине**

**«Технологии проектирования программного обеспечения»**

Направление подготовки 01.03.02 Прикладная математика и информатика

Работу выполнил студент группы 4ИТ/2 И.А. Молчанов

(подпись)

Работу проверил доц. каф. ИТ, к.т.н., доц. А.Н. Полетайкин

(подпись)

Краснодар

2023

**ВВЕДЕНИЕ**

**Тема**: Тестирование программной системы

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

**Задание:**

**1.** Выполнить системное пользовательское тестирование работоспособности ПС посредством воздействия на её интерфейсную часть, разработанную при выполнении лабораторной работы №6. При возникновении несоответствий задокументировать их, произвести поиск логических ошибок, разработать тесты для их обнаружения, устранить ошибки.

**2.** Выполнить нагрузочное тестирование программы и оценить эффективность разработанных при выполнении лабораторной работы №6 автоматизированных функций ПС, запросов к БД. В качестве критерия эффективности использовать время выполнения функции. Предварительно подготовить 5 массивов исходных данных на 10, 50, 100, 500 и 1000 записей. Построить графики зависимости времени вычислений от объема исходных данных.

**3.** Выполнить стрессовое тестирование ПС. В случае обнаружения некорректных ситуаций описать их, выявить причины и принять меры к их устранению.

**4.** Разработать модульный тест для проверки работоспособности кода, реализующего автоматизированные функции ПС.

**Индивидуальная тема:** программное средство для анализа новостных сообщений на фейк.

**1. Системное (ручное) пользовательское тестирование ПС.**

Системное тестирование – это тестирование всей системы в целом, как правило, через ее пользовательский интерфейс. Следует проверить соответствует ли разработанная ПС заявленным требованиям.

Система адаптивна под различные разрешения экрана, интерфейс простой и незапутанный.

Для корректного тестирования необходимо провести тест со стороны пользователя.

**1.1 Тест №1 (Пользователь)**

Состоит из проверки функционала пользователя в следующих действиях:

1. Зайти на страницу формы заявки
2. Заполнить поля некорректными данными
3. Проверить валидацию данных
4. Зайти на страницу формы ввода текста
5. Ввести текст для анализа
6. Проверить валидацию данных

Ожидается, что все элементы интерфейса корректно работают. Этот тест успешно пройден.

Скриншоты срабатывания некоторых пунктов проверки:

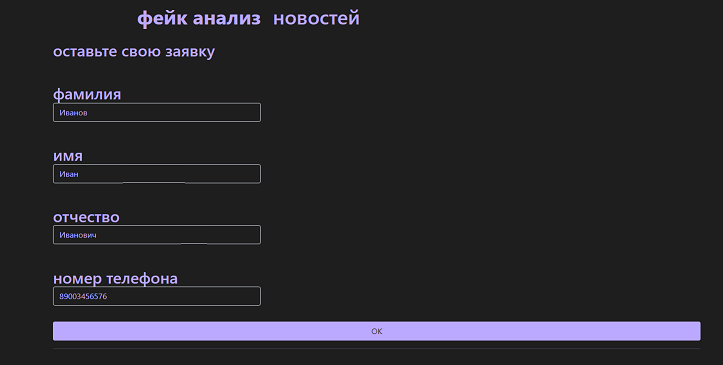


Рисунок 1 – окно формы заявки

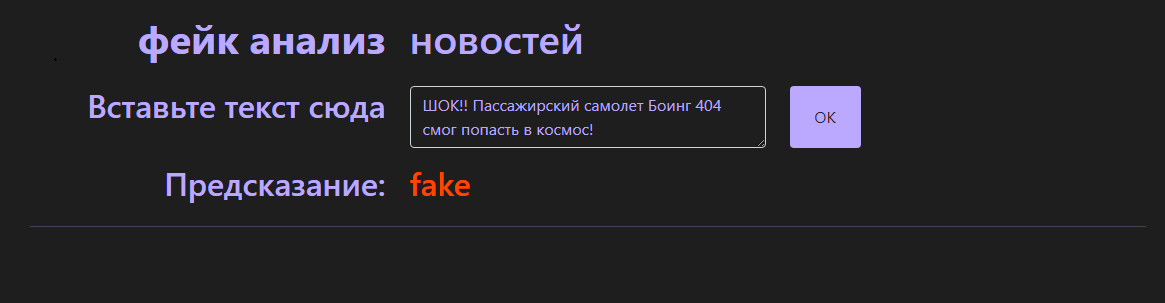


Рисунок 2 – окно при фейковой новости

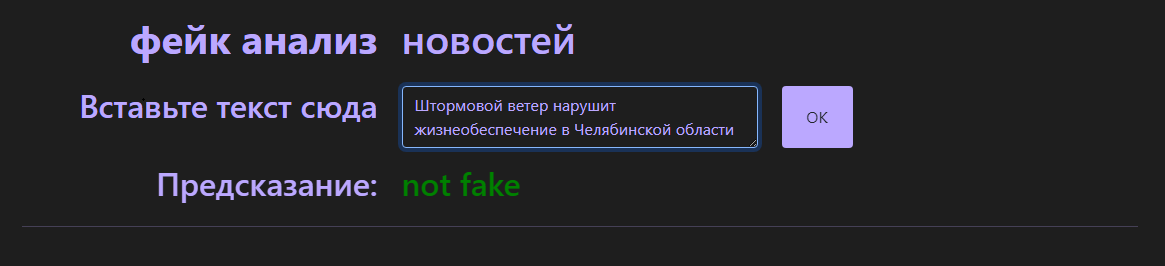


Рисунок 3 – окно при правдивой новости

**2. Нагрузочное тестирование**

Выполним нагрузочное тестирование программы и оценим эффективность автоматизированных функций ПС, запросов к БД. В качестве критерия эффективности используем время выполнения функции. Для тестирования подготовим 5 массивов исходных данных на 10, 50, 100, 500 и 1000. Тестирование проводилось на таблице обработки заявок всех клиентов. Результаты тестирования представлены в таблице 1.

Технические характеристики используемой машины:

* Оперативная память: 8 Гб
* Процессор: 4 ядра с тактовой частотой 2.333 ГГц, 8 потоков
* Разрядность Windows: 64-разрядная операционная система
* Размер жёсткого диска: 256 Гб

|  |  |  |  |
| --- | --- | --- | --- |
| № | Кол-во записей | Операция | Результат (сек.) |
| 1 | 10 | Вставка | 0,115 |
| 2 | 50 | Вставка | 0,212 |
| 3 | 100 | Вставка | 0,289 |
| 4 | 500 | Вставка | 0,422 |
| 5 | 1000 | Вставка | 0,619 |

Таблица 1 – Сравнительная таблица нагрузочного тестирования

При данном тестировании в базу данных загружается сначала 10 записей об объектах, далее 50, 100, 500 и 1000. Далее строится проект, тем самым проверяется скорость работы ПС с различными объёмами данных

После проведения теста мы получили следующие данные:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количество исходных данных | 10 | 50 | 100 | 500 | 1000 |
| Время вычислений | 1550 мс | 2120 мс | 2890 мс | 4220 мс | 6190 мс |

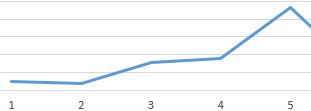


Рисунок 8 – График работы программы

**3. Стрессовое тестирование**

В этом виде тестирования необходимо проверить работу системы при чрезмерной нагрузке на неё. Стрессовое тестирование ПС будет проведено посредством отправки многократных запросов к БД PostgreSQL. Во время тестирования было обнаружено, что даже в стрессовых ситуациях приложение продолжает функционировать, БД быстро обрабатывает запросы. Могут возникать небольшие задержки в работе интерфейса, но после выполнения череды запросов задержки перестают себя проявлять.

**4. Модульное тестирование**

Для тестирования работы вставки была установлена библиотека jest и написан следующий тест:

module.exports = { presets: [['@babel/preset-env', {targets: {node: 'current'}}]], };

module.exports = { testEnvironment: 'jsdom',

moduleFileExtensions: ['js', 'json', 'vue'],

transform: { '^.+\\.js$': 'babel-jest',

'^.+\\.vue$': '@vue/vue3-jest' }

}

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 - Результат модульного теста

**Вывод:** в рамках данной работы была протестирована программная система различными способами, освоены методики тестирования разработанной программной системы в ручном режиме, проведено успешное системное, нагрузочное, стрессовое и модульное тестирования, также в работе приведён код модульных тестов.