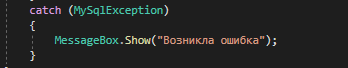
* 1. **Тестирование и документирование программного продукта**
     1. **Обоснование методов и средств тестирования**

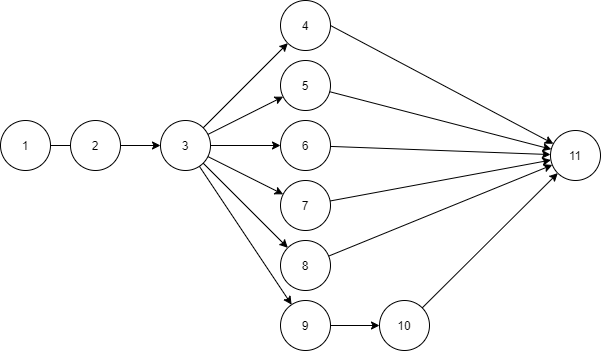
Разработанный проект имеет довольно большой объем кода. Так как математических вычислений в программе не предусмотрено, модульное тестирование посредством встроенных средств Visual Studio будет очень проблематичным.

В связи с этим было принято решение добавить в код программы специальные строки кода – исключительные ситуации. Данные ситуации могут возникать при каких-либо ошибках во время работы программы. Пример блока кода для модуля «Авторизации»

  
Рисунок 1 – Пример блоков для кнопки «Авторизации»

Полные блоки кода для всех модулей можно увидеть в листинге программы.

Чтобы последовательно и безошибочно тестировать программу, необходимо составить тестовый граф:

  
Рисунок 2 – тестовый граф

Пояснение по графу:

1. Открытие программы
2. Регистрация
3. Авторизация
4. Открытие формы VR
5. Открытие формы AR
6. Открытие формы телестоматологии
7. Открытие формы зубной щетки
8. Открытие формы 3D печати
9. Открытие формы записи к врачу
10. Запись к врачу
11. Выход

Для работы по тестовому графу необходимо составить тестовые пути, по которым будет тестироваться программа. Тестовые пути по разработанной программе выглядят следующим образом:

* 1-2-3-4-11
* 1-2-3-5-11;
* 1-2-3-6-11;
* 1-2-3-7-11;
* 1-2-3-8-11;
* 1-2-3-9-10-11;

После добавления и использования специальных блоков кода для «перехвата» исключительных ситуаций, необходимо провести интеграционное тестирование.

Интеграционное тестирование – вид тестирования, которое подразумевает собой тестирование связей между различными модулями программы, их взаимодействие. Данные для интеграционного тестирования можно представить в виде таблицы:

Тест кейс для формы авторизации:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Исходные данные | Цель | Описание | Ожидаемый результат | Результат |
| 1 | Логин – test  Пароль – test ( Рисунок 3) | Тестирование регистрации | Тестирование взаимодействия модуля регистрации с базой данных | Запись введенных данных в базу данных | Сообщение об успешном добавлении данных |
| 2 | Логин – test Пароль – test ( Рисунок 4) | Тестирование авторизации | Тестирование взаимодействия модуля авторизации с базой данных | Поиск данных в таблице БД. При совпадении введенных данных и найденных производится переход на главную | Совпадение введенных данных и БД. Переход на главную форму |

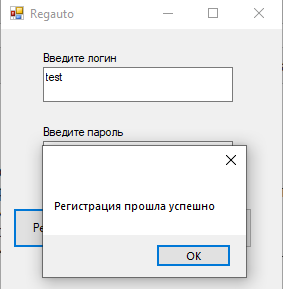


Рисунок 3 – Тестирование регистрации

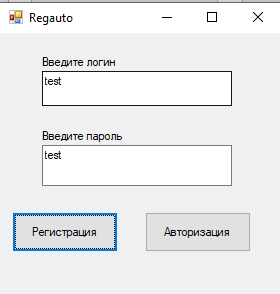


Рисунок 4 ­­– Тестирование авторизации

* + 1. **Результаты тестирования**

Для отладки было применено модульное, интеграционное, системное и регрессионное тестирование. Это делалось для как можно большего охвата программы различными тестами.

Как можно видеть в предыдущем разделе, система успешно прошла все этапы тестирования, критических багов и недочетов обнаружено не было.

Успешно проведенное тестирование дает гарантию работы программы в самых разнообразных условиях использования. Исключается аварийное завершение работы программы. При возникновении ошибки программа сообщит пользователю, что именно послужило причиной остановки работы.

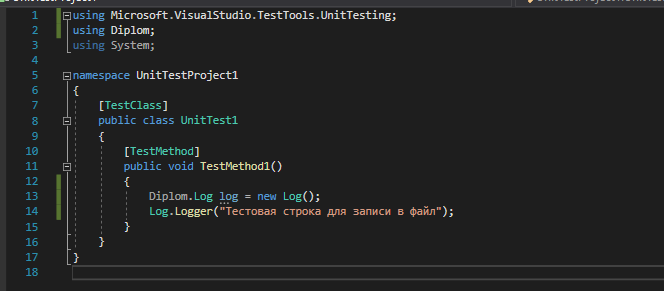
Модульное тестирование производилось с помощью try catch блоков. Подобные блоки были добавлены во все модули кода, что поможет в будущем избежать критических ситуаций в работе программы.

Интеграционное тестирование производилось с помощью тестового графа и тестовых путей. Все тестовые пути были пройдены успешно.

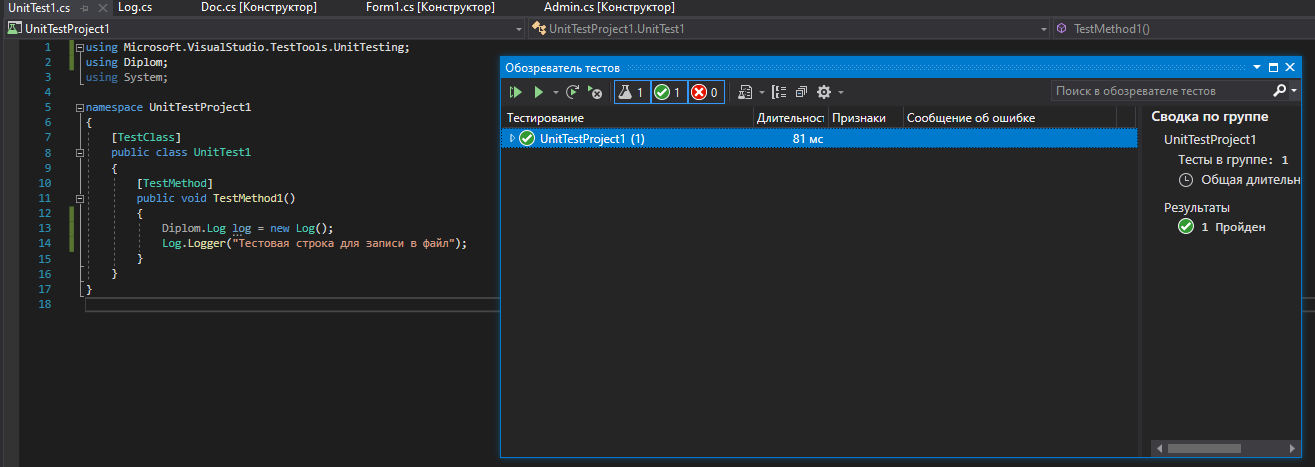
Последним этапом было проведение системного тестирования. В нем тестировались производительность программы, работа в стрессовых условиях и регрессионное тестирование. Программа успешно прошла все вышеперечисленные методики тестирования.

Из этого следует, что программный продукт полностью готов к работе в реальных условиях.

Unit тестирование проводилось с помощью встроенных средств Visual Studio. Был создан отдельный проект, который по ссылке подключается к приложению. Для тестирования была выбрана функция логгирования. В тестовом местоде был написан следующий код:

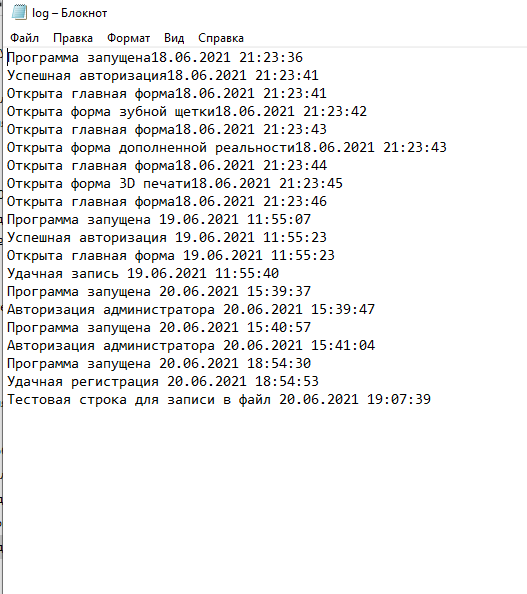


После написания и подключения кода можем запустить данный тест:

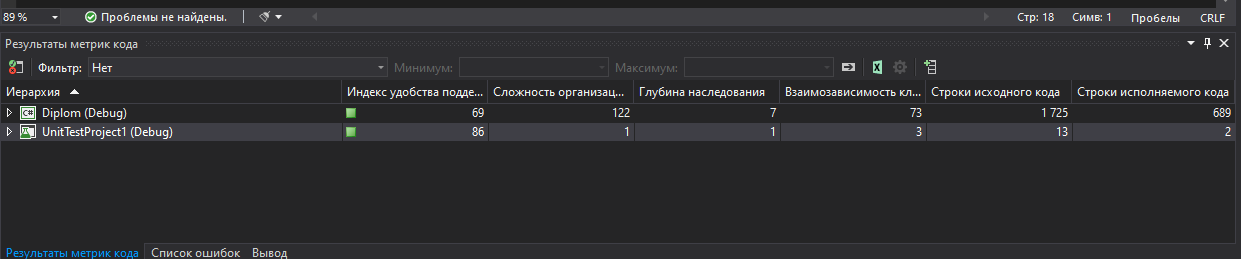


Как видно на скриншоте, тест успешно пройден.

Заходим в файл логгирования и убеждаемся, что строка успешно записана:



Метрики кода:



Степень покрытия кода выяснить не представляется возможным, так как эта функция доступна только в Enterprise версии Visual Studio