

МИНОБРНАУКИ РОССИИ  
федеральное государственное автономное образовательное учреждение высшего образования  
«Санкт-Петербургский политехнический университет Петра Великого»  
Институт дополнительного образования  
Высшая инженерная школа

Выпускная квалификационная работа  
по программе профессиональной переподготовки  
«Разработчик прикладного программного обеспечения (язык C#)»  
«Веб-приложение для создания и прохождения тестов»

Выполнил: Григорьев В. А.  
Руководитель: Щукин А. В.

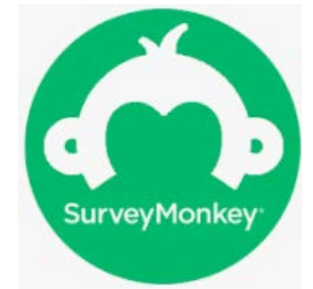
Санкт-Петербург  
2021

# Предметная область

Тестирование (от слова test — испытание, проверка) применяется для определения пригодности объекта тестирования для выполнения тех или иных функций.

Традиционный тест - метод диагностики испытуемых, в котором они отвечают на одни задания, в одинаковое время, в одинаковых условиях и с одинаковой оценкой.

В заданиях открытого типа со свободным изложением испытуемый должен самостоятельно сформулировать ответ, никакие ограничения на них в задании не накладываются.



# Цели и задачи

## Цель ВКР:

- Подготовить веб приложение для создания и прохождения традиционных тестов с вопросами открытого типа со свободным изложением.

## Задачи:

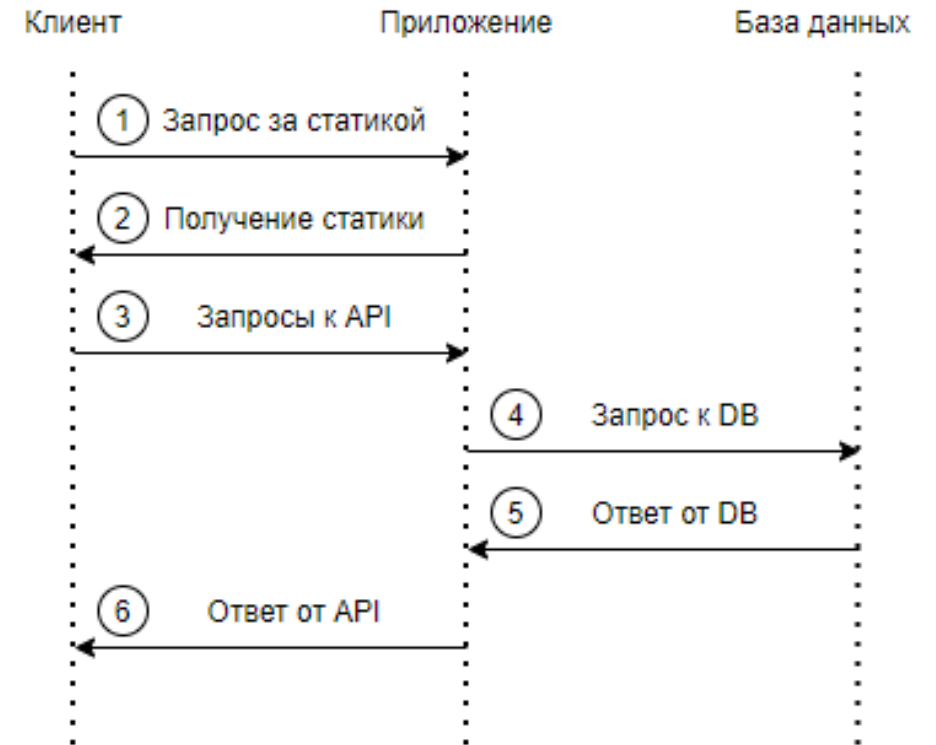
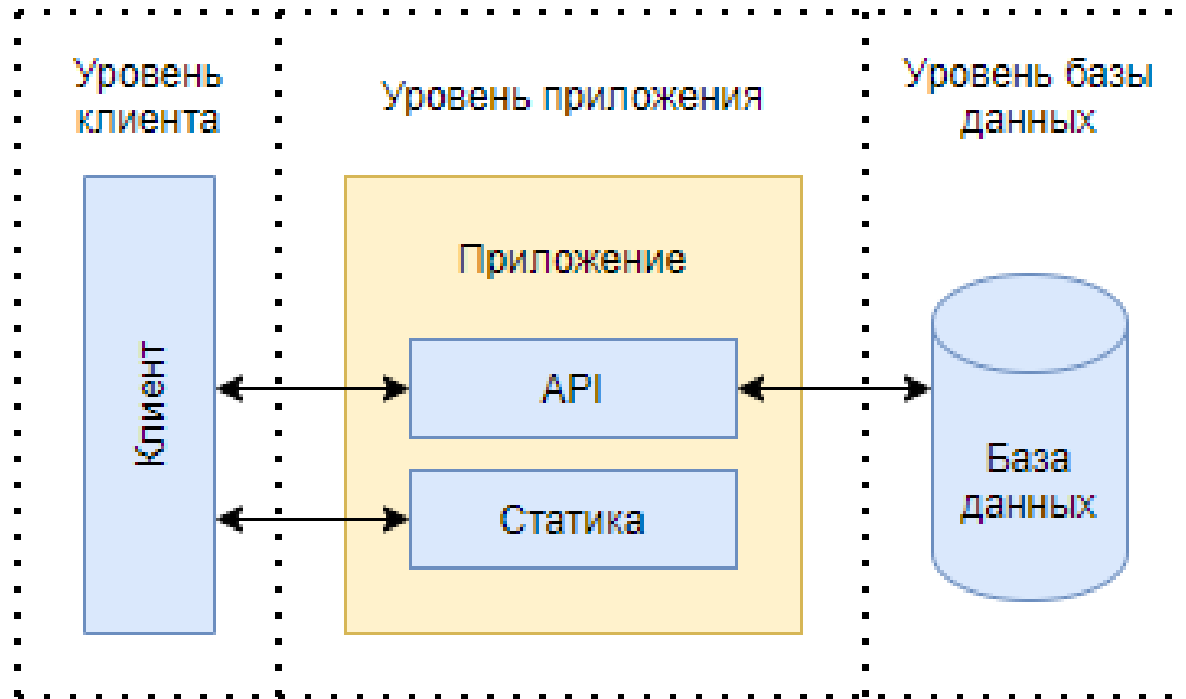
- 1. Разработать структуру базы данных для хранения данных.
- 2. Разработать backend часть приложения на ASP.NET Core WebApi.
- 3. Разработать frontend часть приложения в виде html страниц.
- 4. Провести тестирование.

# Требования

Приложение должно обеспечивать следующую функциональность:

- Авторизация и аутентификация.
- Получение списка тестов, созданных авторизованным пользователем.
- Создание и редактирование тестов.
- Прохождение существующих тестов авторизованным пользователем.
- Получение статистики по выполненным тестам.
- Вывод логов приложения в консоль и в файл.
- Сбор и отправка метрик о состоянии приложения.

# Архитектура



3-х уровневая клиент-серверная модель

# База данных

ORM - EF Core.

СУБД - SQLite.

Таблицы: Users, Tests, Questions, Answers и Results.



# Аутентификация и авторизация

| Имя метода               | Тип  | Адрес     | Параметры | Тело запроса | Тело ответа |
|--------------------------|------|-----------|-----------|--------------|-------------|
| <b>RegisterUserAsync</b> | POST | /register | -         | UserDto      | (Guid)Token |
| <b>LoginAsync</b>        | POST | /login    | -         | UserDto      | (Guid)Token |
| <b>LogoutAsync</b>       | GET  | /logout   | -         | -            | -           |

# Работа с сущностью теста

| Имя метода          | Тип    | Адрес     | Параметры    | Тело запроса | Тело ответа   |
|---------------------|--------|-----------|--------------|--------------|---------------|
| <b>GetListAsync</b> | GET    | /List     | -            | -            | List<TestDto> |
| <b>AddAsync</b>     | GET    | /AddEmpty | -            | -            | (Guid)TestId  |
| <b>UpdateAsync</b>  | PATCH  | /         | -            | TestDto      | -             |
| <b>GetAsync</b>     | GET    | /         | (Guid)testId | -            | TestDto       |
| <b>RemoveAsync</b>  | DELETE | /         | (Guid)testId | -            | -             |



# Работа с сущностью вопроса

| Имя метода          | Тип    | Адрес     | Параметры            | Тело запроса | Тело ответа       |
|---------------------|--------|-----------|----------------------|--------------|-------------------|
| <b>GetListAsync</b> | GET    | /List     | (Guid) testId        | -            | List<QuestionDto> |
| <b>AddAsync</b>     | GET    | /AddEmpty | (Guid) testId        | -            | (Guid)QuestionId  |
| <b>UpdateAsync</b>  | PATCH  | /         | -                    | QuestionDto  | -                 |
| <b>RemoveAsync</b>  | DELETE | /         | (Guid)<br>questionId | -            | -                 |

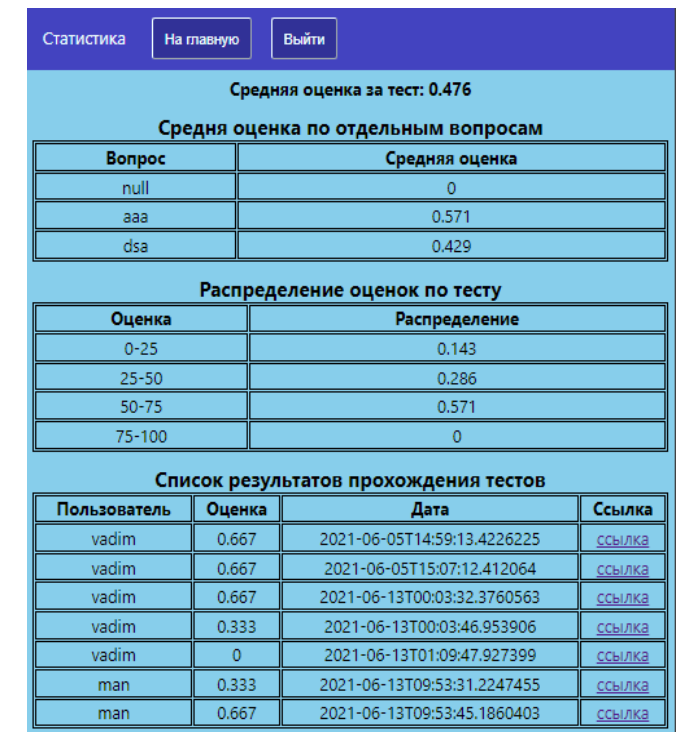
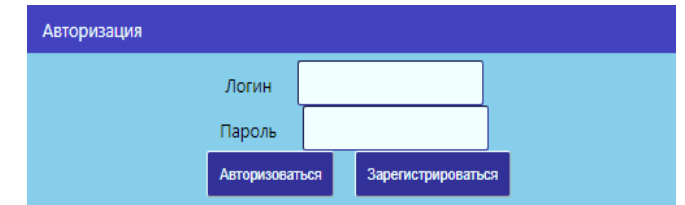
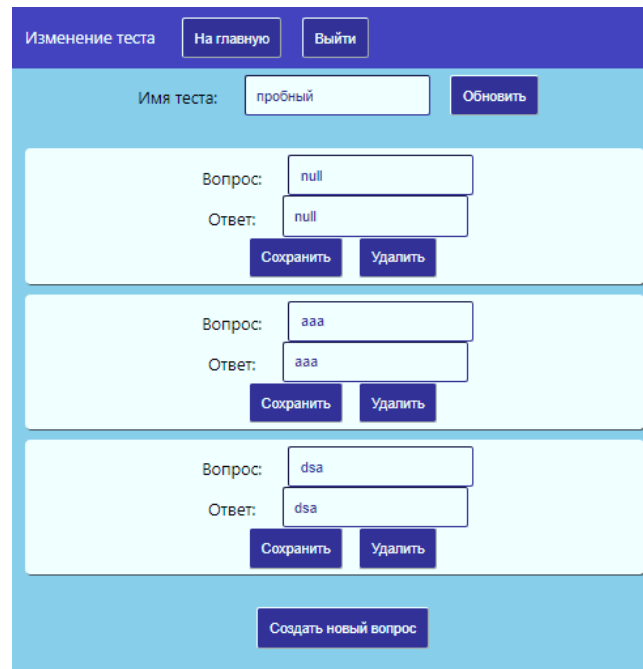
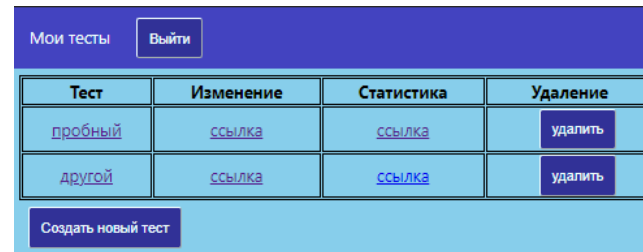
# Работа с сущностью ответа

| Имя метода           | Тип  | Адрес     | Параметры       | Тело запроса    | Тело ответа     |
|----------------------|------|-----------|-----------------|-----------------|-----------------|
| <b>SendListAsync</b> | POST | /SendList | -               | List<AnswerDto> | (Guid)resultId  |
| <b>GetListAsync</b>  | GET  | /List     | (Guid) resultId | -               | List<AnswerDto> |

# Работа по генерации статистики

| Имя метода                       | Тип | Адрес                 | Параметры     | Тело запроса | Тело ответа            |
|----------------------------------|-----|-----------------------|---------------|--------------|------------------------|
| <b>GetResultListAsync</b>        | GET | /GetResultList        | (Guid) testId | -            | List<ResultDto>        |
| <b>GetAverageScoreAsync</b>      | GET | /GetAverageScore      | (Guid) testId | -            | double                 |
| <b>GetScorePerQuestionsAsync</b> | GET | /GetScorePerQuestions | (Guid) testId | -            | List<(string, double)> |
| <b>GetScoreDistributionAsync</b> | GET | /GetScoreDistribution | (Guid) testId | -            | List<double>           |

# Frontend



В качестве Frontend части приложения используются статические страницы (HTML+CSS+JS).

# Swagger

Для документации API  
используется библиотека  
NSwag.

По пути /swagger  
выводится UI для Swagger.



Schemes  
HTTP

**Answers**

- POST /api/v1/Answers/SendList
- GET /api/v1/Answers/List

**Questions**

- GET /api/v1/Questions/List
- GET /api/v1/Questions/AddEmpty
- PATCH /api/v1/Questions
- DELETE /api/v1/Questions

**Statistics**

**Tests**

**Users**

Models

# Логирование

Для логирования используется библиотека NLog.

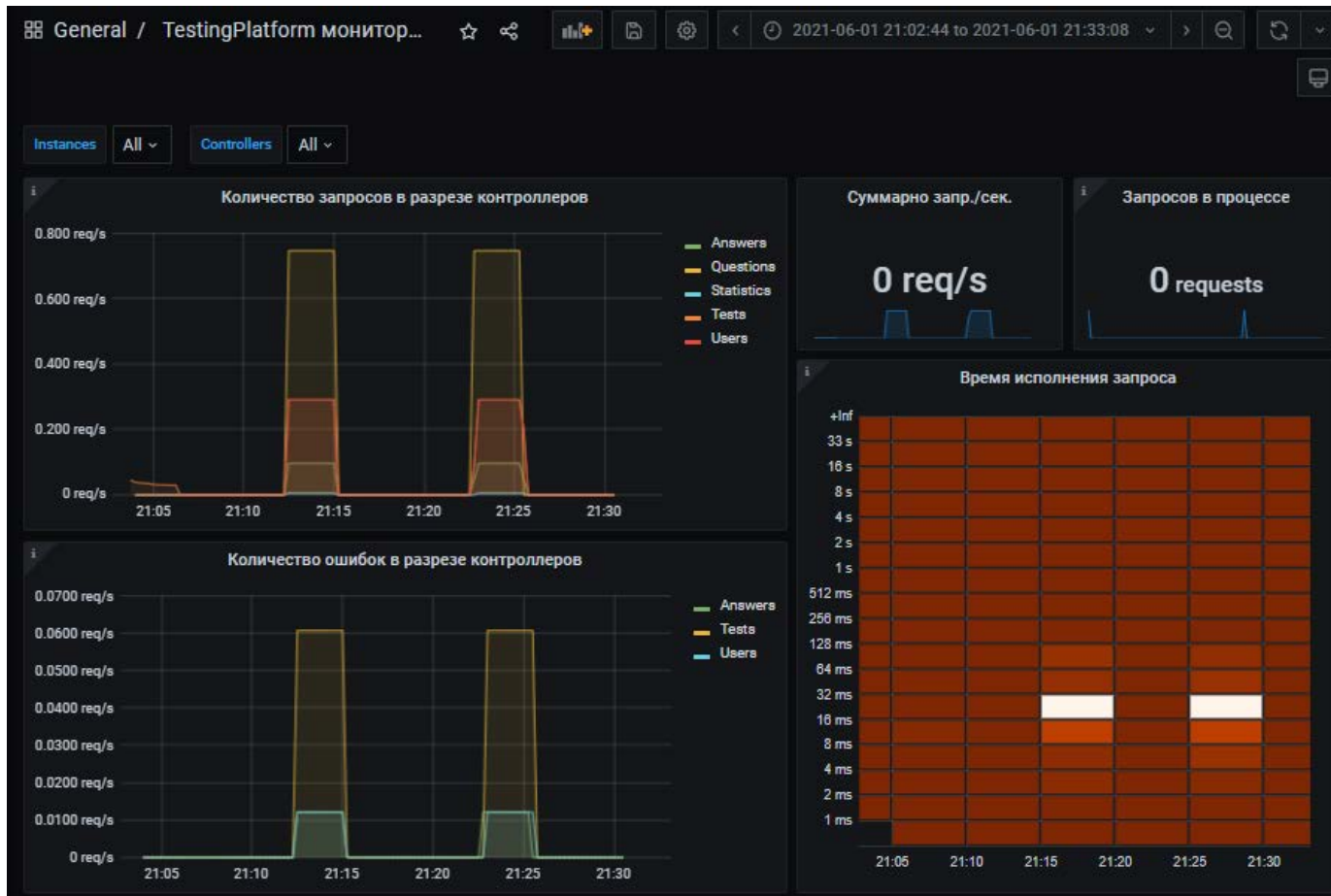
Логи выводятся в файл и в консоль.

Конфигурирование логирования находится в файле nlog.config в корне приложения.

A screenshot of a code editor showing the configuration for NLog in an XML file named nlog.config. The file is located in the 'nlog.config' tab. The configuration includes a root element 'nlog' with namespaces for the NLog schema and XSI. It defines two targets: 'allfile' (File target) and 'allconsole' (Console target). The 'allfile' target is configured with a file name and a layout. The 'allconsole' target is configured with a layout. The configuration also includes two rules: one for logging to 'allfile' with a minimum level of 'Warn', and another for logging to 'allconsole' with a minimum level of 'Info'.

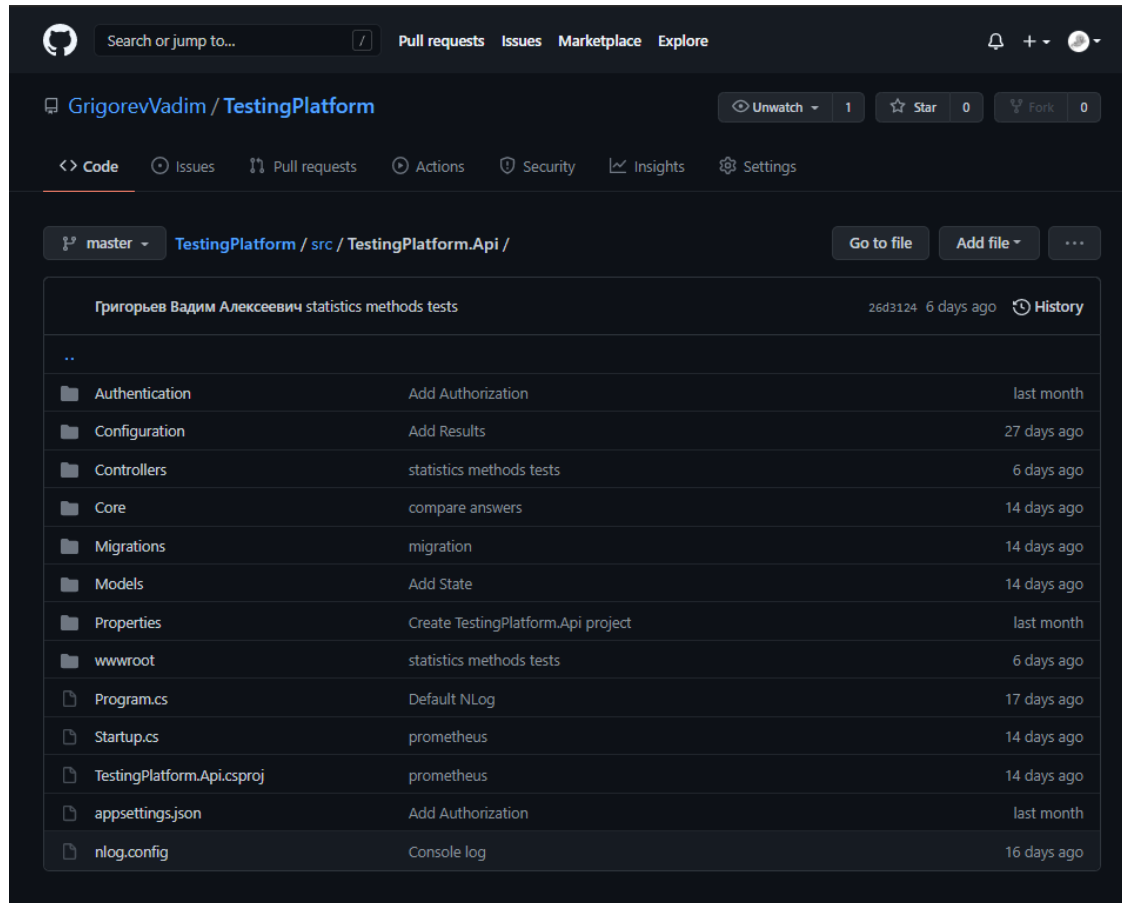
```
<?xml version="1.0" encoding="utf-8" ?>
<nlog xmlns="http://www.nlog-project.org/schemas/NLog.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <targets>
    <target xsi:type="File" name="allfile" fileName=".\logs\nlog-${
      layout="${longdate} | ${event-properties:item=EventId_Id:
    <target xsi:type="Console" name="allconsole"
      layout="${longdate} | ${event-properties:item=EventId_Id:
    </targets>
  <rules>
    <logger name="*" minlevel="Warn" writeTo="allfile" />
    <logger name="*" minlevel="Info" writeTo="allconsole" />
  </rules>
</nlog>
```

# Мониторинг



Метрики собираются в Prometheus затем выводятся в Grafana

# GitHub

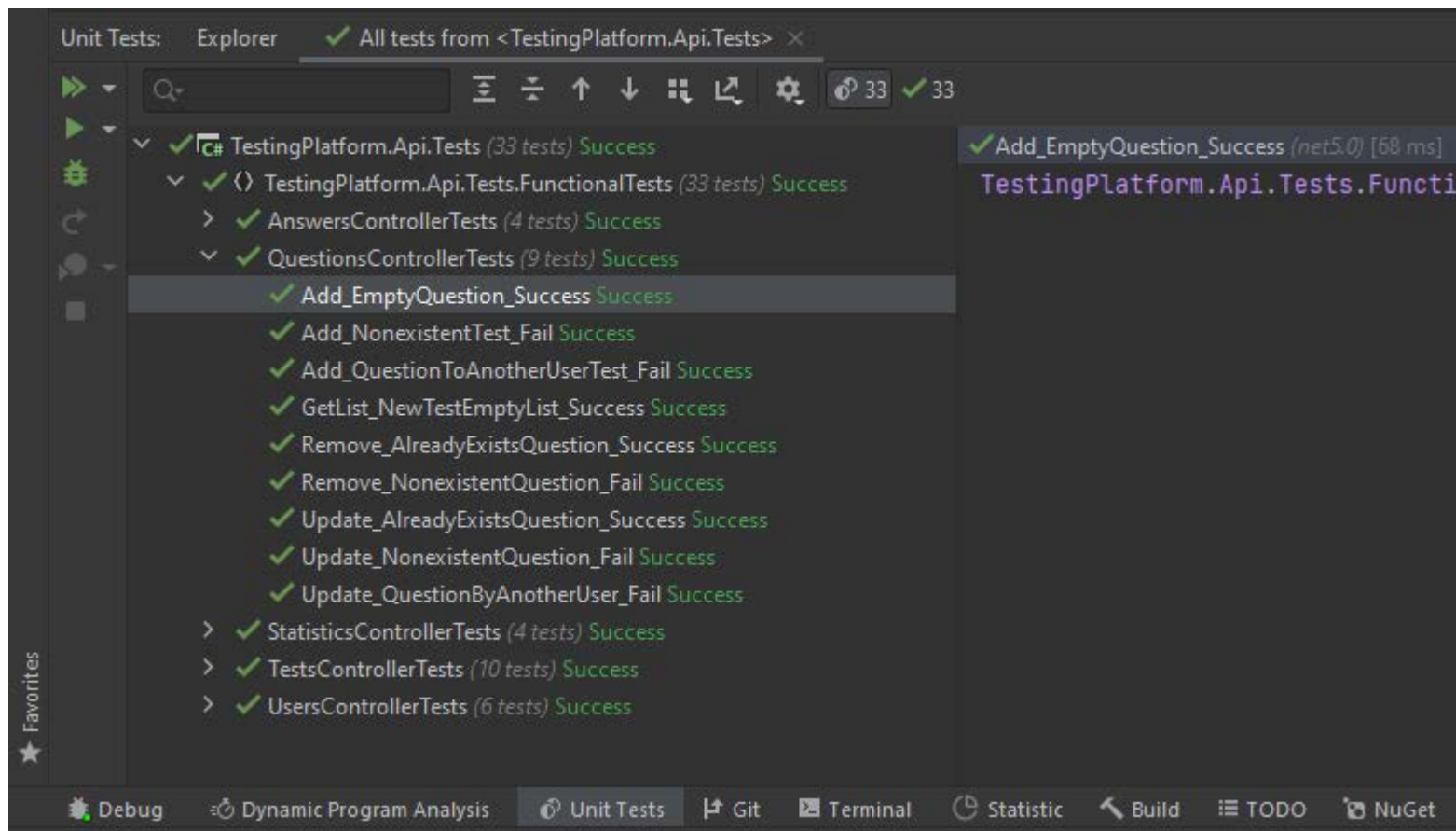


Исходный код приложения располагается по адресу <https://github.com/GrigorevVadim/TestingPlatform>.



# Тестирование

На API приложение  
написано 33  
функциональных теста.  
Все тесты стабильно  
выполняются.



# Заключение

Задачи:

1. Разработана структура базы данных для хранения данных.
2. Разработана backend часть приложения на ASP.NET Core WebApi.
3. Разработана frontend часть приложения в виде html страниц.
4. Проведено тестирование.

Цель ВКР:

Веб-приложение для создания и прохождения тестов подготовлено.

Спасибо за внимание!