**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

**ОТЧЕТ**

**по производственной практике**

**Тема: Покрытие тестами программного обеспечения.**

| Студент гр. 1303 |  | Чубан Д.В. |
| --- | --- | --- |
| Руководитель |  | Никитенко Д.А. |

Санкт-Петербург

2024

**ЗАДАНИЕ**

**НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ**

| Студент Чубан Д.В. | | |
| --- | --- | --- |
| Группа 1303 | | |
| Тема практики: Покрытие тестами программного обеспечения. | | |
| Задание на практику:  Необходимо покрыть юнит-тестами программу для радиомониторинга летательных аппаратов. Также необходимо автоматизировать их запуск. | | |
| Сроки прохождения практики: 26.06.2024 – 09.07.2024 | | |
| Дата сдачи отчета: 09.07.2024 | | |
| Дата защиты отчета: 09.07.2024 | | |
|  | | |
| Студент |  | Чубан Д.В. |
| Руководитель |  | Никитенко Д.А. |

**АННОТАЦИЯ**

Нужно создать юнит-тесты для программы радиомониторинга летательных аппаратов, проверяющих работоспособность общей логики программы, математической части, высчитывающей передвижение объектов в зоне радиоприемников, графического интерфейса программы, а также серверной части, обменивающейся данными между радиоприемником и программой на стороне конечного пользователя.

**SUMMARY**

It is necessary to create unit tests for an aircraft radio monitoring program that checks the operability of the general logic of the program, the mathematical part that calculates the movement of objects in the radio area, the graphical interface of the program, as well as the server part that exchanges data between the radio receiver and the program on the end user side.

**СОДЕРЖАНИЕ**

|  | Введение | 5 |
| --- | --- | --- |
| 1. | Решение задачи | 6 |
| 1.1. | Разработка юнит-тестов | 6 |
| 1.2. | Автоматизация системы проверок | 6 |
| 2. | Использованные технологии | 7 |
| 3. | Скан отзыва руководителя с оценкой, подписью и печатью | 8 |
|  | Заключение | 9 |

**ВВЕДЕНИЕ**

Целью практики является создание юнит тестов, покрывающих полный функционал программы и автоматизация проверок работоспособности.

Для достижения цели ставятся следующие задачи: разработка юнит-тестов для каждого модуля программы; автоматизация системы проверок.

В конечном счете, тестирование поможет находить сбои в работе приложения при модификации исходного кода.

**1. РЕШЕНИЕ ЗАДАЧИ**

**1.1. Разработка юнит-тестов для каждого модуля программы**

Для разработки тестов сначала был проанализирован программный код проекта и принцип его работы для выявления возможных критических и пограничных значений для функций.

Был произведен рефакторинг кода: божественные классы разделены на отдельные для удобства тестирования.

Также были написаны mock-классы, имитирующие поведение основных систем приложения, чтобы исключить возможность ошибки в системах, используемых тестируемым классом.

Далее, для каждого класса был создан отдельный проект, вынесенный в отдельную директорию для тестов, в котором описаны все тесты для его методов, предусматривающие критические и пограничные значения, а также штатную работу. Каждый такой проект собирается в отдельный исполняемый файл. Данный вариант нужен для автоматизации тестовой проверки при сборке проекта.

Также по запросу был создан отдельный класс, вызывающий эти тесты внутри основного кода приложения для более быстрой ручной проверки при отсутствии необходимости полной сборки проекта, если изменения затрагивают не все приложение.

**1.2. Автоматизация системы проверок**

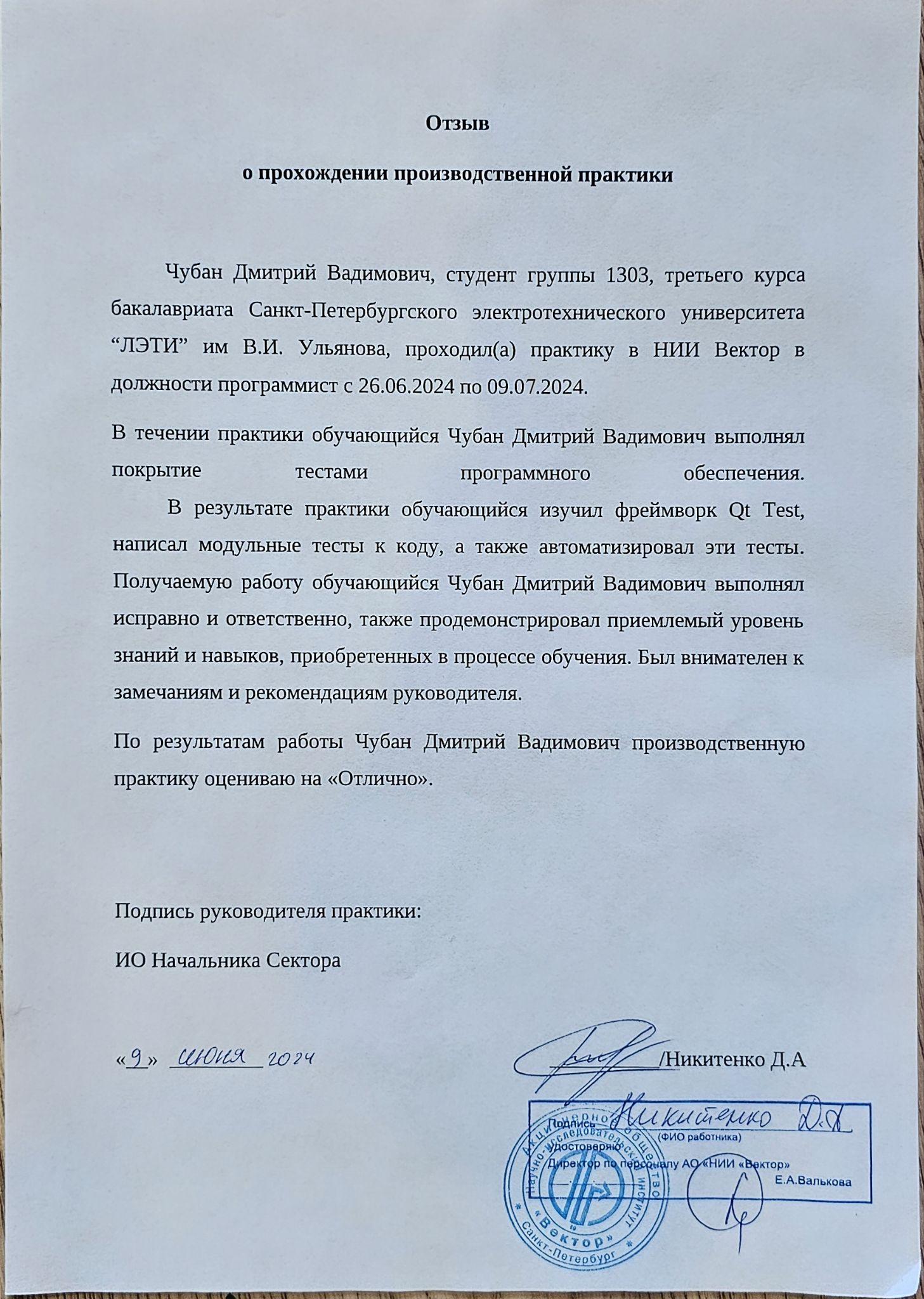
Для автоматизации системы проверок были изменены параметры сборки всего проекта. После его компиляции вызываются описанные ранее исполняемые файлы тестов. Если проверка в каком либо из тестов не была пройдена, разработчик получит сообщение об ошибке, указывающее ожидаемое и полученное значение функции при тестировании или сообщение о критической ошибке (например, segmentation fault). В таком случае, дальнейшая сборка проекта будет прервана.

**2. ИСПОЛЬЗОВАННЫЕ ТЕХНОЛОГИИ**

Для реализации юнит-тестов и автоматизации их применения использовались следующие технологии:

C++, Qt, QTest, QTCreator, qmake

**3. СКАН ОТЗЫВА РУКОВОДИТЕЛЯ С ОЦЕНКОЙ, ПОДПИСЬЮ И ПЕЧАТЬЮ**



**ЗАКЛЮЧЕНИЕ**

В результате был успешно реализовано юнит-тестирование проекта и автоматизация процесса тестирования.

Для этого были выполнены все поставленные задачи: разработаны юнит-тесты для каждого модуля проекта и автоматизировано их применение.

Полученный результат полностью соответствует поставленной цели.