Министерство образования и науки РФ

ФГБОУ ВПО «Омский государственный технический университет»

Кафедра «Автоматизированные системы обработки информации и управления»

**ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ ПО ДИСЦИПЛИНЕ УПРАВЛЕНИЕ ПРОГРАМНЫМИ ПРОЕКТАМИ**

|  |  |
| --- | --- |
| Старший преподаватель | Купш А.Г. |

подпись, дата

|  |  |
| --- | --- |
| Студент | Попов В.З. |
| Студент | Зарков А.Д. |
| Студент | Штырц Е.В. |

подпись, дата

Омск 2016

Содержание

[Планирование работ по выполнению проекта 3](#_Toc470186300)

[Оценка трудозатрат на выполнение работ 5](#_Toc470186301)

[Использование систем контроля версий исходного кода программ 9](#_Toc470186302)

[Использование средств автоматизации тестирования программного обеспечения 10](#_Toc470186303)

[Сервис записи на СТО. 11](#_Toc470186304)

[Список используемых источников 12](#_Toc470186305)

# Планирование работ по выполнению проекта

На основе технического задания на разработку автоматизированной информационной системы, была выбрана спиральная модель жизненного цикла разработки ПО. Так как эта модель обеспечивает быстрое получение результатов и изменение требований не влечет за собой весомых проблем.

На основе спиральной модели был сформирован календарный план выполнения работ по проекту. Так же были выделены основные этапы работ – это проектирование и разработка.

Были выделены основные задачи внутри отдельных этапов работ – это такие задачи как: распределение задач, и разработка модулей приложения.

При создании задач определялись зависимости между ними. Например задача по оцениванию СТО не могла выполниться раньше чем задачи по регистрации СТО и пользователя, так как при оценке СТО необходимы данные зарегистрированного СТО и пользователя.

На основе зависимостей и важности задач формировался порядок выполнения задач. На задачи назначались исполнители, так что бы нагрузка была сбалансированной.

Все работы по планированию выполнялись в программе Microsoft Project 2007. Для автоматизации управления проектом по разработке использовался Битрикс. На рисунке 1 показан проект полученный в Microsoft Project 2007, а на рисунке 2 календарный план в Битрикс.

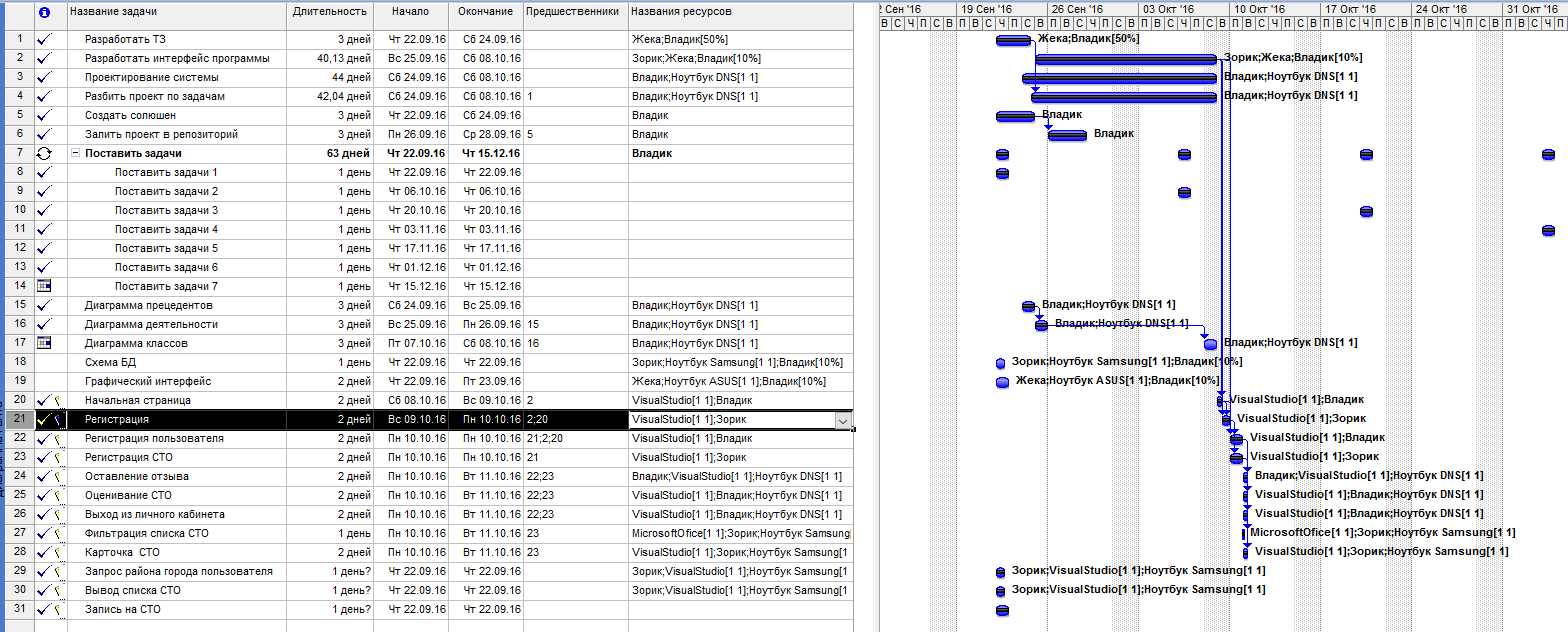


Рисунок 1 – Планирование в программе Microsoft Project 2007

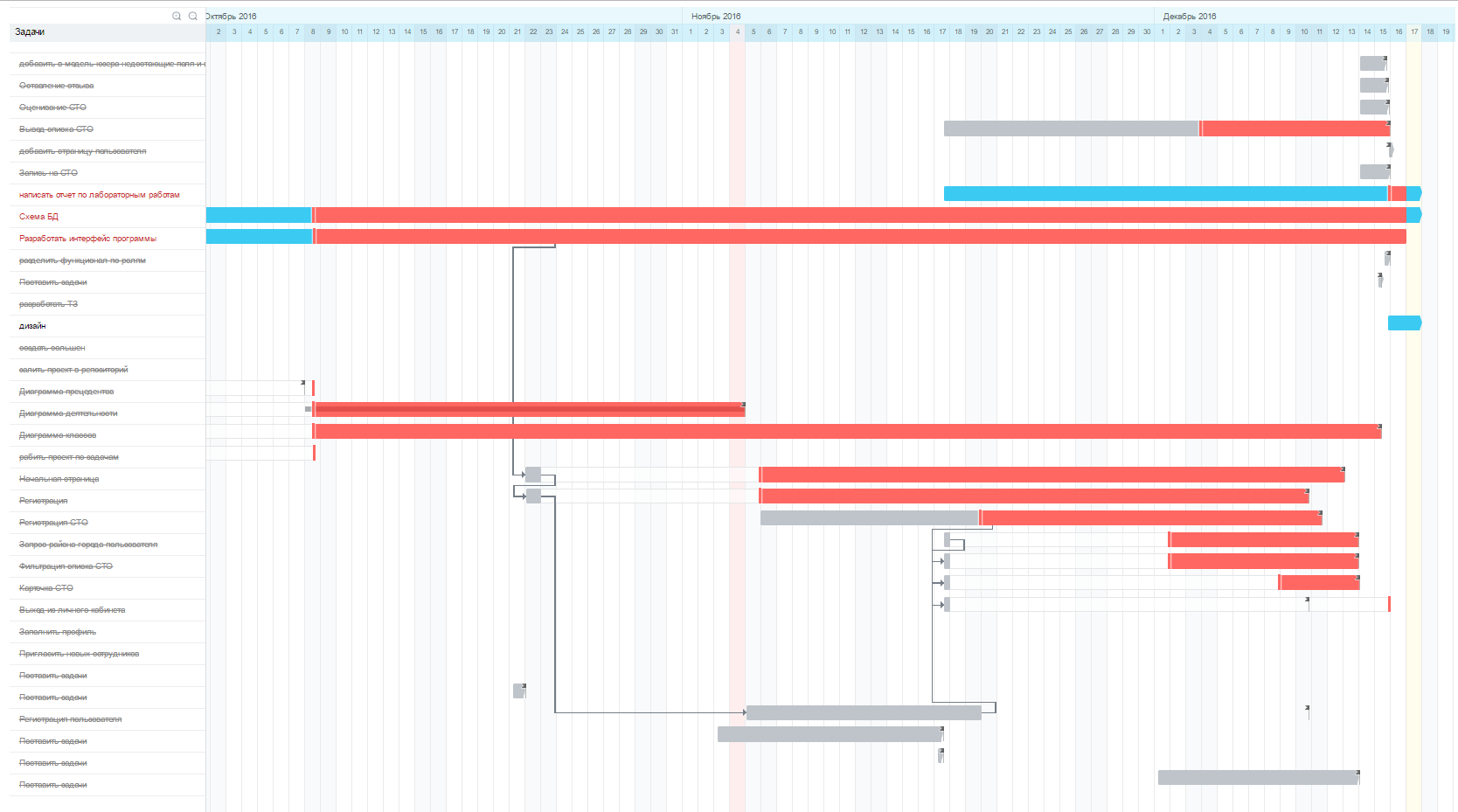


Рисунок 2 – Календарный план в Битрикс

# Оценка трудозатрат на выполнение работ

Расчет трудоемкости и стоимости выполнения работ по проекту, для которого был составлен календарный план в процессе выполнения лабораторной работы № 1. Расчет трудоемкости проводился методом функциональных точек. Средства реализации: ограничений не накладывается, рекомендуется использование MicrosoftExcel (рисунок 6). Функционально-ориентированные метрики косвенно измеряют программный продукт и процесс его разработки. Вместо подсчета LOC-оценки при этом рассматривается не размер, а функциональность или полезность продукта. В качестве количественной характеристики применяется понятие количества функциональных точек FP (function points).Для получения функционально-ориентированных метрик (FP-метрик) используются функциональные и концептуальные модели будущей системы(Рисунок 3-5). Основывается процесс получения функционально-ориентированных метрик на функциональной модели системы. Рассматриваются только наиболее значимые процессы, соответствующие основным функциям разрабатываемого программного продукта. Каждый бизнес-процесс имеет входные и выходные данные, находящие свое отражение в концептуальной модели системы.

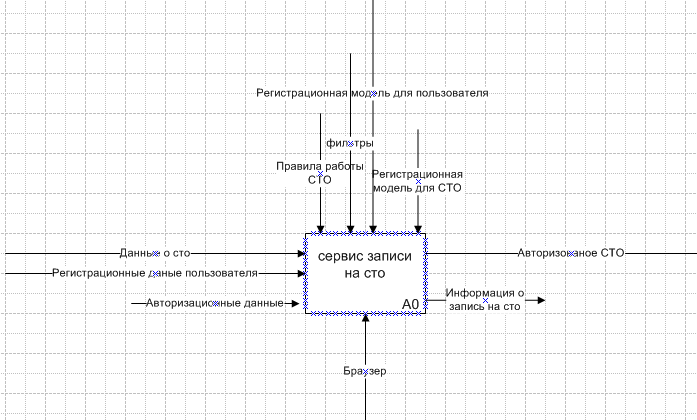


Рисунок 3 – Диаграмма IDEF0

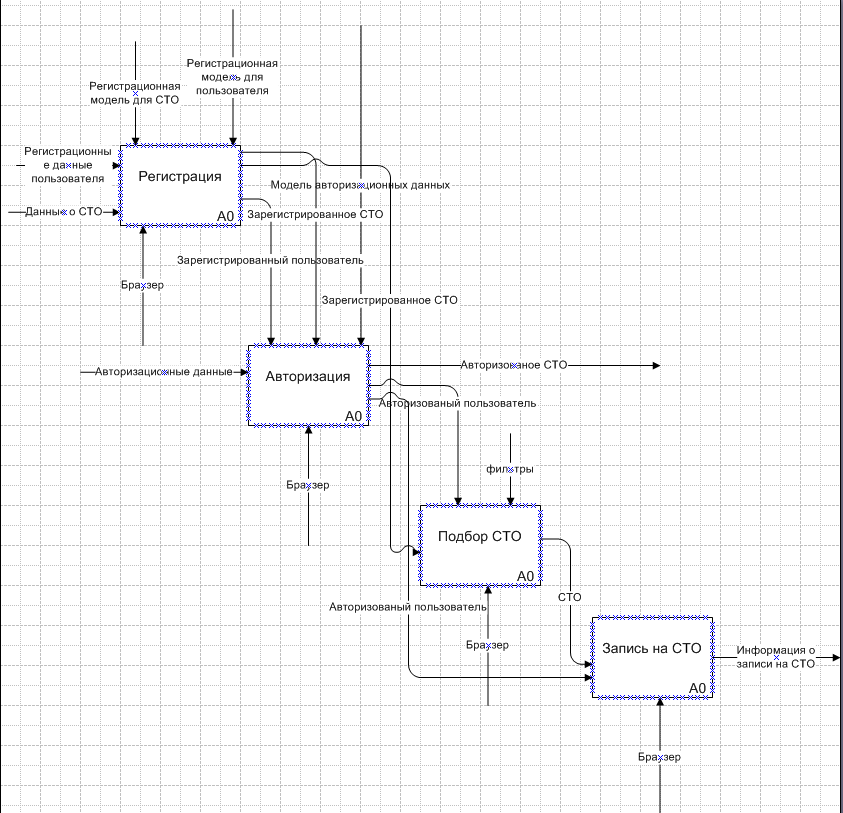


Рисунок 4 – Диаграмма IDEF0

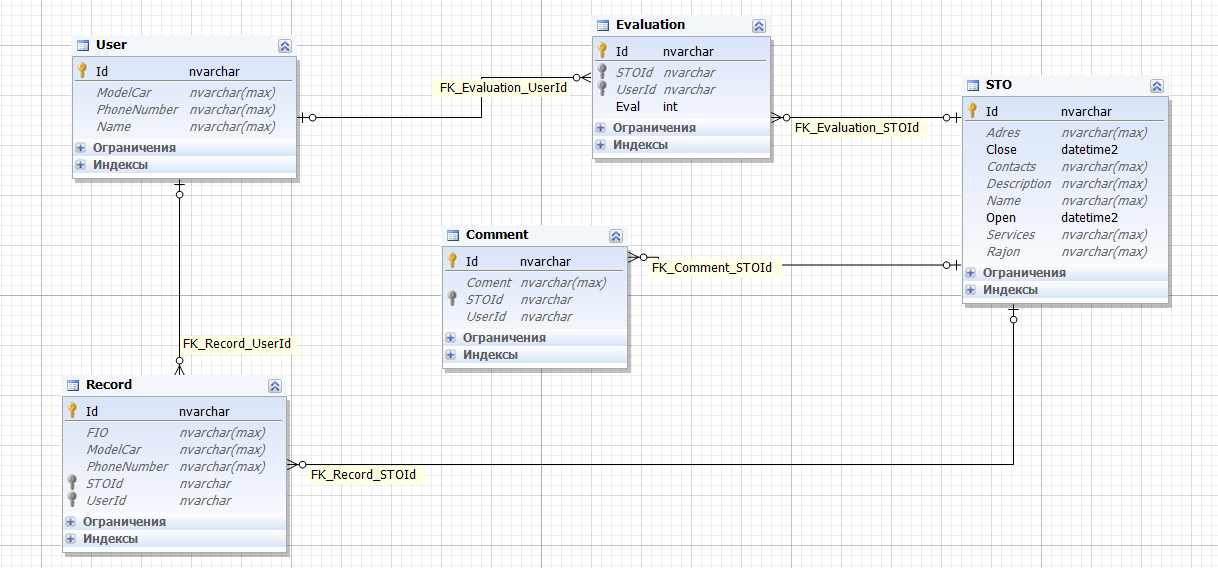


Рисунок 5 – Диаграмма IDEF1x

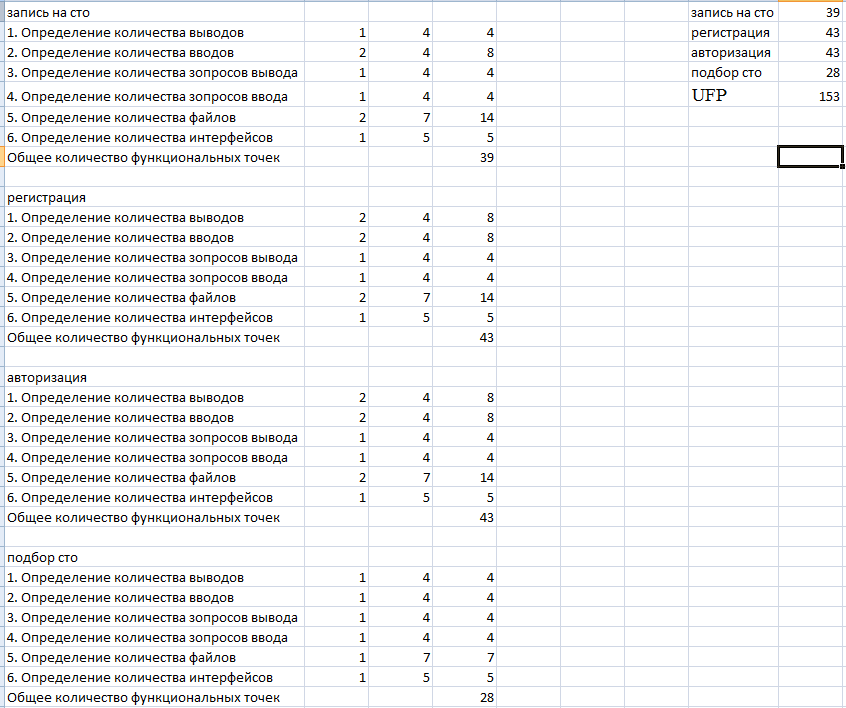


Рисунок 6 –Расчеты количества функциональных точек

Расчет оценки функциональных требований приведен в таблице 1.

Таблица 1 **—** Оценка функциональных требований

|  |  |
| --- | --- |
| **Фактор** | **Оценка**  **DI, degree of influence** |
| 1. Передача данных | 2 |
| 2. Распределенная обработка данных | 1 |
| 3. Производительность | 1 |
| 4. Распространенность используемой  конфигурации | 1 |
| 5. Частота транзакций | 2 |
| 6. Оперативный ввод данных | 4 |
| 7. Эффективность работы конечного  пользователя | 3 |
| 8. Оперативное обновление | 3 |
| 9. Сложность обработки | 3 |
| 10. Повторная используемость | 2 |
| 11. Простота инсталляции | 1 |
| 12. Простота эксплуатации | 3 |
| 13. Разнообразные условия  размещения | 3 |
| 14. Простота изменений | 3 |
| ***TDI = ∑ DI= TDI*** | 32 |

Расчет значения фактора выравнивания производится по формуле 1.

*VAF = (TDI \*0.01) + 0.65 (1)*

Подсчитаем значение фактора выравнивания для нашей системы:

*VAF = 0.65 + (32\*0.01) = 0.97 (2)*

Дальнейшая оценка в выровненных функциональных точках зависит от типа оценки. Начальное оценка количества выровненных функциональных точек для программного приложения определяется по следующей формуле:

*AFP = UFP \* VAF. (3)*

Таблица 2 **—** Оценка выровненных функциональных точках

|  |  |
| --- | --- |
| **Информационные объекты** | **AFP** |
| 1 Регистрация | 43 \* 0.97 = 41.71 |
| 2 Подбор СТО | 28 \*0.97= 27.16 |
| 3 Авторизация | 43 \*0.97= 41.71 |
| 4 Запись СТО | 39 \*0.97= 37.83 |
| ∑AFP | 148.41 |

# Использование систем контроля версий исходного кода программ

При разработке использовалась система контроля версий Git и хостинг GitHub и программа SouceTree.

На локальной машине был создан новый проект под название STO после чего проект был экспортирован на хостинг GitHub . При создании репозитория в нем по умолчанию существует только 1 ветвь под название master. Для разработки была создана ветвь dev , а master используется для показа проекта заказчику.

При разработке было создано 3 ветки сделано 27 комитов 1 реверт 1 мерж.

Использование репозитория очень удобно тем что: можно отменить ненужные изменения и всегда вернуться к старой версии проекта, разрабатывая в разных ветках заказчик видит только конечную версию проекта.

На рисунке 7 представлена работа над проектом в программе SouceTree. На рисунке 8 представлен проект на удаленном хостинге GitHub.

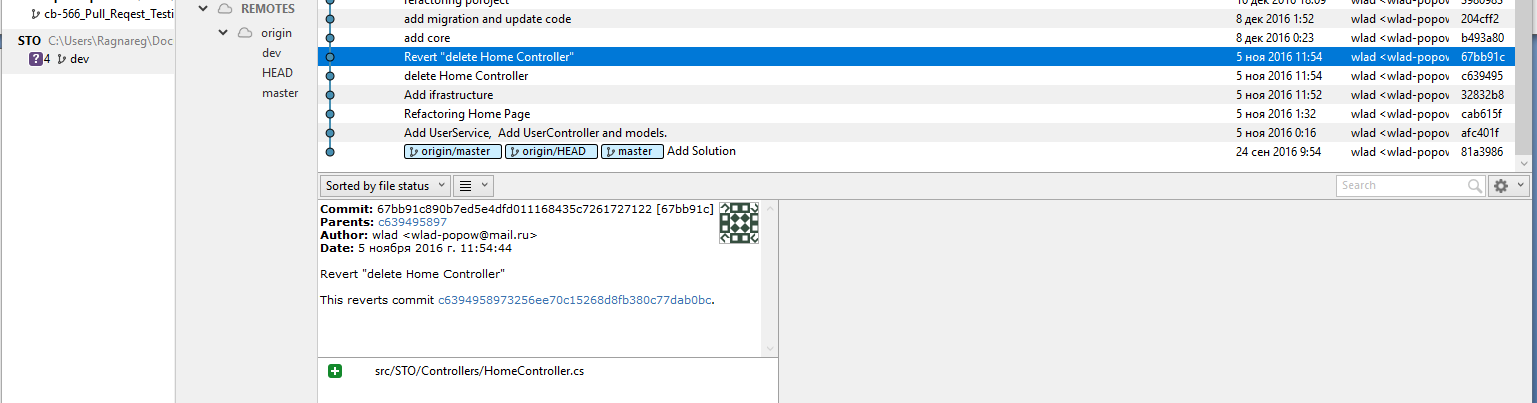


Рисунок 7 – Проект в программе SouceTree

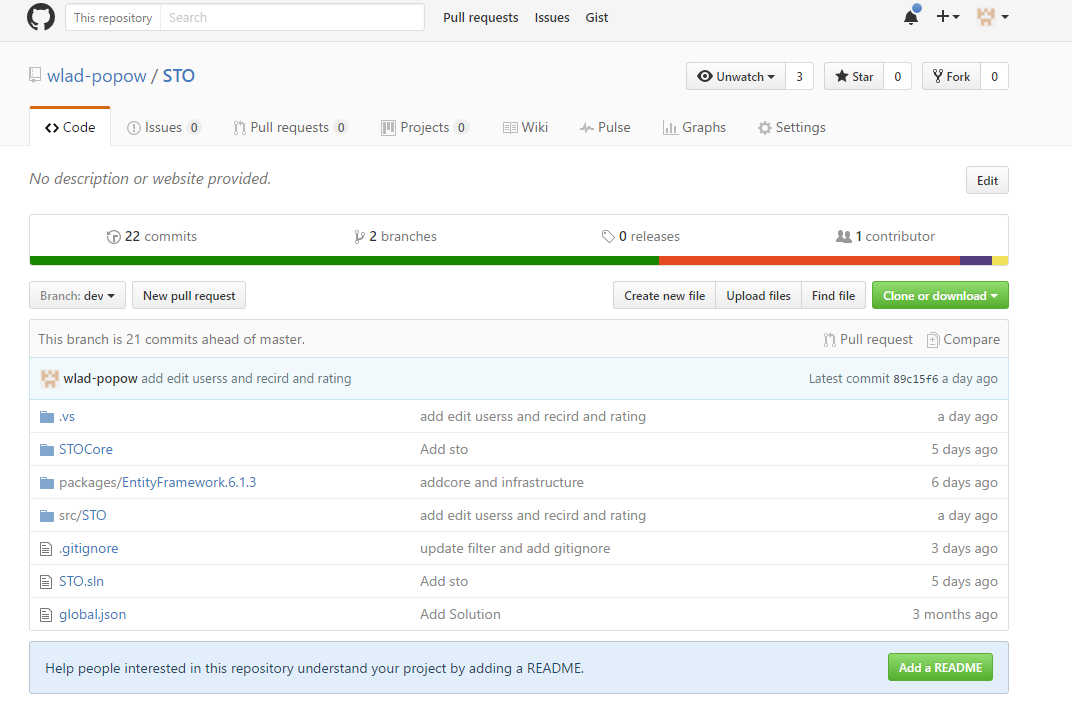


Рисунок 8 – Удаленный хостинг проекта GitHub

# Использование средств автоматизации тестирования программного обеспечения

Для проверки работоспособности программы и правильности выполнение использовалось автоматическое тестирование. Для написания тестов использовался Фреймворк xUnit. В тестах реализована отправка запросов к сервису и сравнивание ответов с ожидаемыми результатами. В первом тесте оправляется запрос на получение СТО которое существует в азе данных и сравнивается возвращаемый статус код, если равен ОК то тест пройден успешно. А во втором тесте отправляется запрос на получение СТО которого в базе не существует и сравнивается возвращаемый статус код, если не равен ОК то тест пройден успешно.

Так как приложение написано на asp .net core с использование шаблона mvc, весь функционал приложения находится в контроллерах. Что крайне неудобно, и большую часть функционала программы не удалось протестировать. Это и является слабым местом программы, потому что нельзя наверняка сказать о надежности выполнения.

Можно разделить функционал программы, убрав основной из контроллеров. Это даст возможность протестировать больше модулей программы и повысит надежность.

На рисунке 9 представлены результаты выполненных тестов.



Рисунок 9 – Результаты тестов

**Код тестов**

using System;   
using System.Linq;   
using [System.Net](https://vk.com/away.php?to=http%3A%2F%2FSystem.Net);   
using System.Net.Http;   
using Xunit;   
  
namespace Test   
{   
public class STOFacts   
{   
[Fact]   
public void GetSto()   
{   
using (var client = new HttpClient())   
{   
string url = "http://localhost:57907/STO/Index/43f905dbb760490495578669e32e88a8";   
var task = client.GetAsync(url);   
var responce = task.Result;   
if (responce.StatusCode == HttpStatusCode.OK)   
{   
url = "OK";   
}   
Assert.Equal(url,"OK");   
}   
}   
  
[Fact]   
public void GetNoValidSto()   
{   
using (var client = new HttpClient())   
{   
string url = "http://localhost:57907/STO/Index/43f905dbb32e88a8";   
var task = client.GetAsync(url);   
var responce = task.Result;   
if (responce.StatusCode == HttpStatusCode.OK)   
{   
url = "OK";   
}   
Assert.NotEqual(url, "OK");   
}   
}   
}   
}

# Сервис записи на СТО.

Веб-сервис предназначен для записи автовладельца на сервис станции технического обслуживания.

Цель веб-сервиса: облегчить автолюбителю выбор станции технического обслуживания для ремонта (обслуживания) автомобиля. Сервис должен предложить автолюбителю возможные станции технического обслуживания и их услуги. После выбора станции технического обслуживание, предоставляется выбор услуги, которая необходима автолюбителю.

Функциональным назначением веб-сервиса является предоставление пользователю информации о наличии услуг и возможность записи на станции технического обслуживания.

Использовать веб-сервис будут пользователи, желающие ознакомится с услугами для обслуживания на станции технического обслуживания.

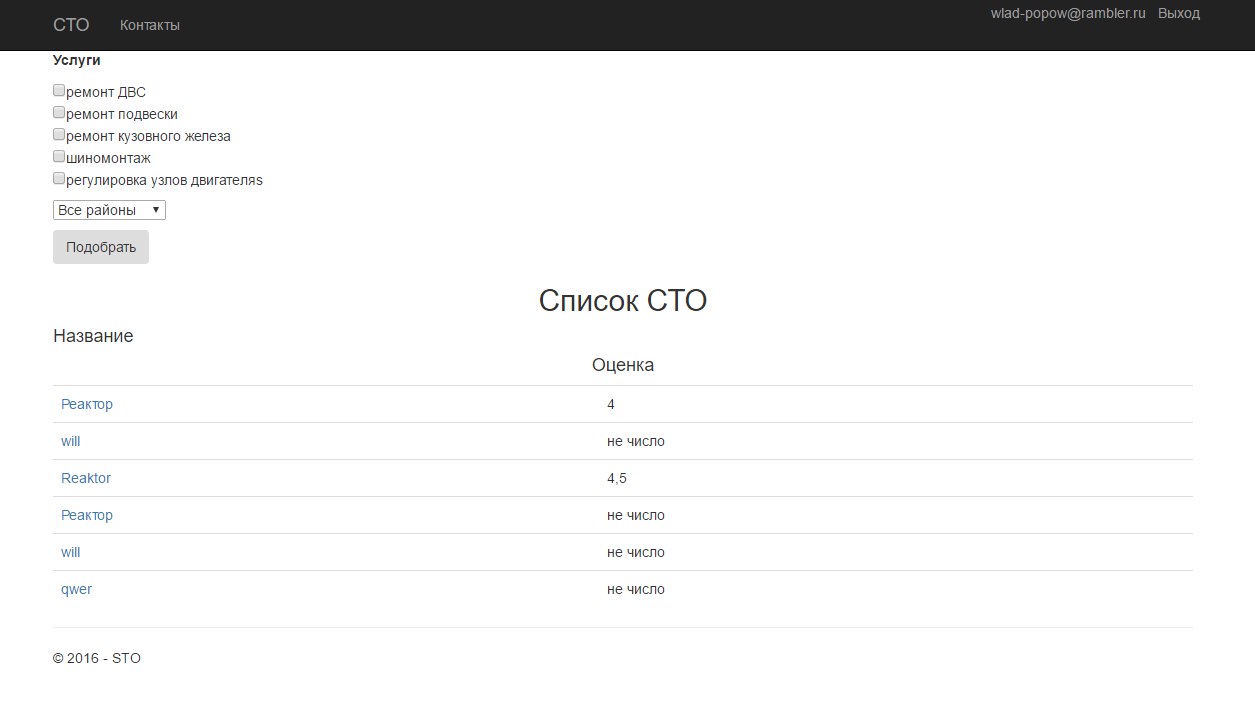


Рисунок 10 – сервис записи на СТО

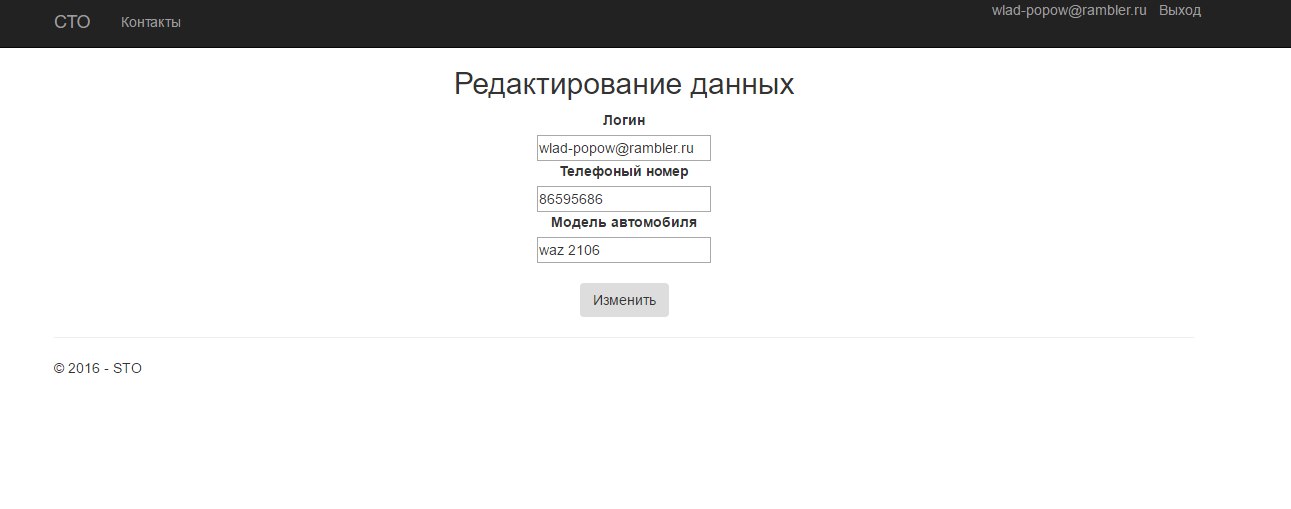


Рисунок 11 – Редактирование данных

# Список используемых источников

1. **Основы управления проектами : практическое руководсво / Дэннис Локк.;** HIPPO, 2004. – 242с.
2. **Управление проектами. Корпоративная система шаг за шагом/ Вадим Богданов;** Манн, Иванов и Фербер, **2012 – 241с.**