**5 РЕЗУЛЬТАТЫ ИСПЫТАНИЙ**

В качестве заданий практикума были предложены и решены следующие задачи:

1) работа с централизованной системой контроля версий Subversion;

2) работа с децентрализованной системой контроля версий Mercurial;

3) работа с децентрализованной системой контроля версий Git;

Для реализации поставленных задач был разработан цикл из нескольких лабораторных работ.

**5.1 Результаты разработки с Subversion**

Для работы с SVN необходимо будет установить пакет с Subversion, если он ещё не был установлен. Это делается при помощи команды sudo apt-get install subversion. Пример работы данной команды и её результат выполнения показан на рисунке 5.1.

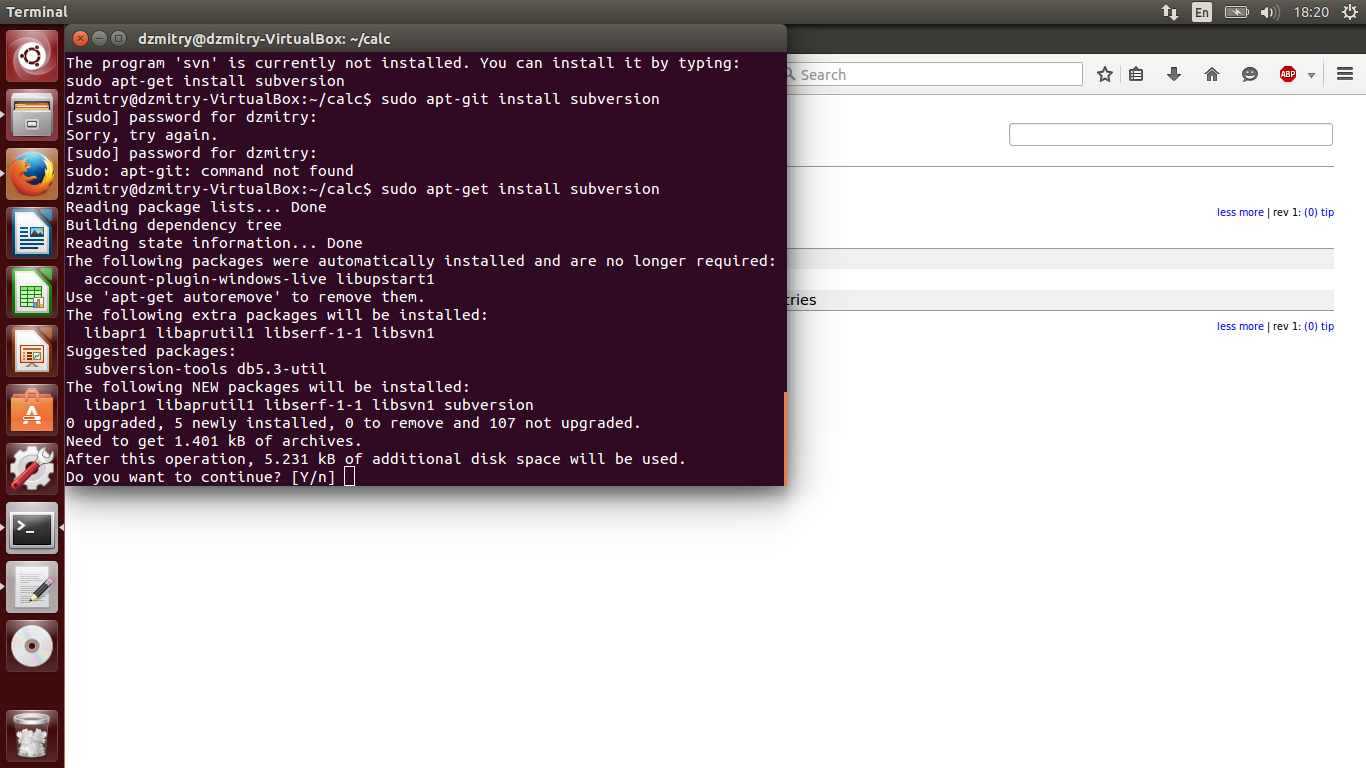
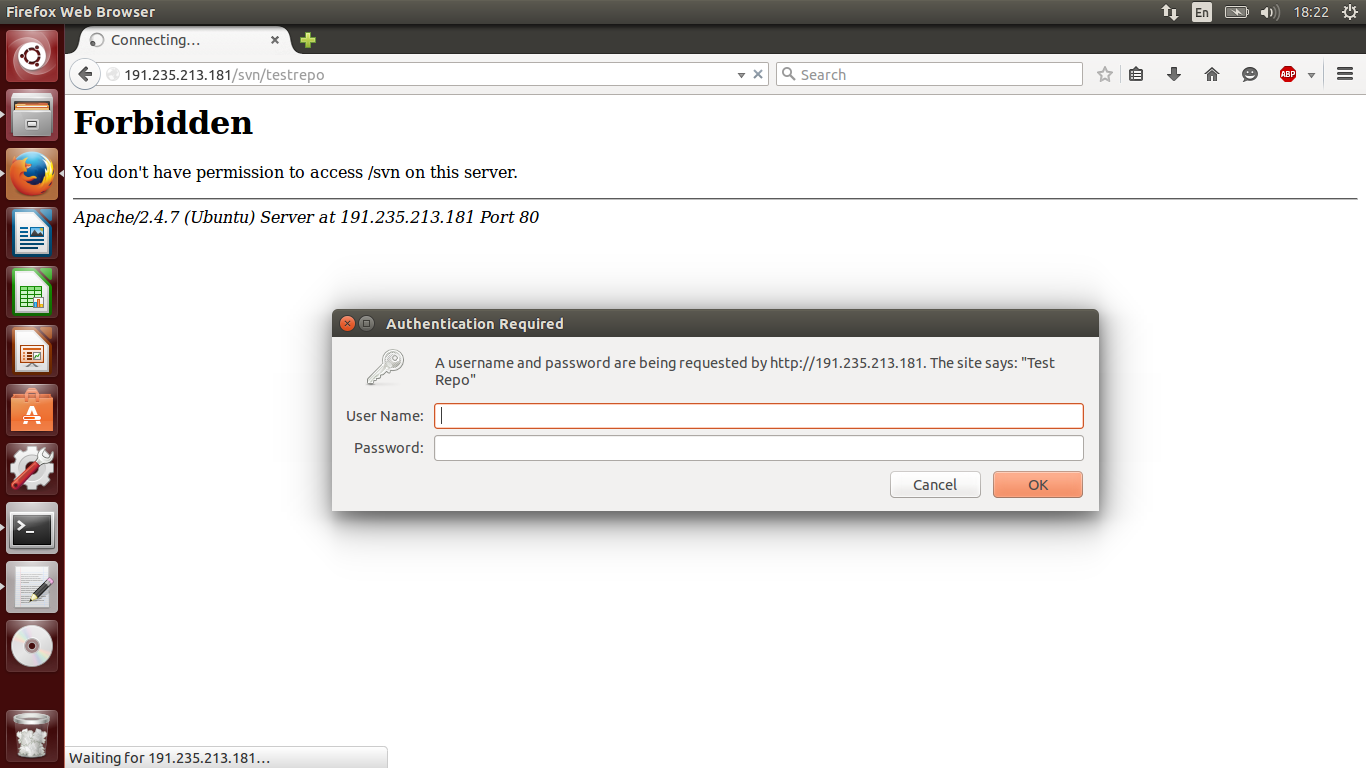


Рисунок 5.1 – Результат работы sudo apt-get install subversion

Далее в лабораторной работе необходимо склонировать готовый репозиторий. Визуально наличие или отсутствие репозитория на сервере проверяем, открывая ссылку на репозиторий в браузере. При этом будет запрошен логин и пароль (который должен создать администратор для каждого из пользователей). Этот шаг не обязателен и его можно пропустить, однако зачастую его используют для проверки правильности установки или возможности открытия репозитория. Окно проверки аутентификации и авторизации можно увидеть на рисунке 5.2.

Рисунок 5.2 – Результат открытия ссылки на репозиторий в браузере

Теперь в консоли начинаем работу с репозиторием. Для начала работы выполняем команду svn co (или svn checkout, которая делает то же самое), которая копирует текущую ревизию в папку с собственным названием. Работа команды (как при передаче необходимых аргументов, так и без неё) показана на рисунке 5.3.

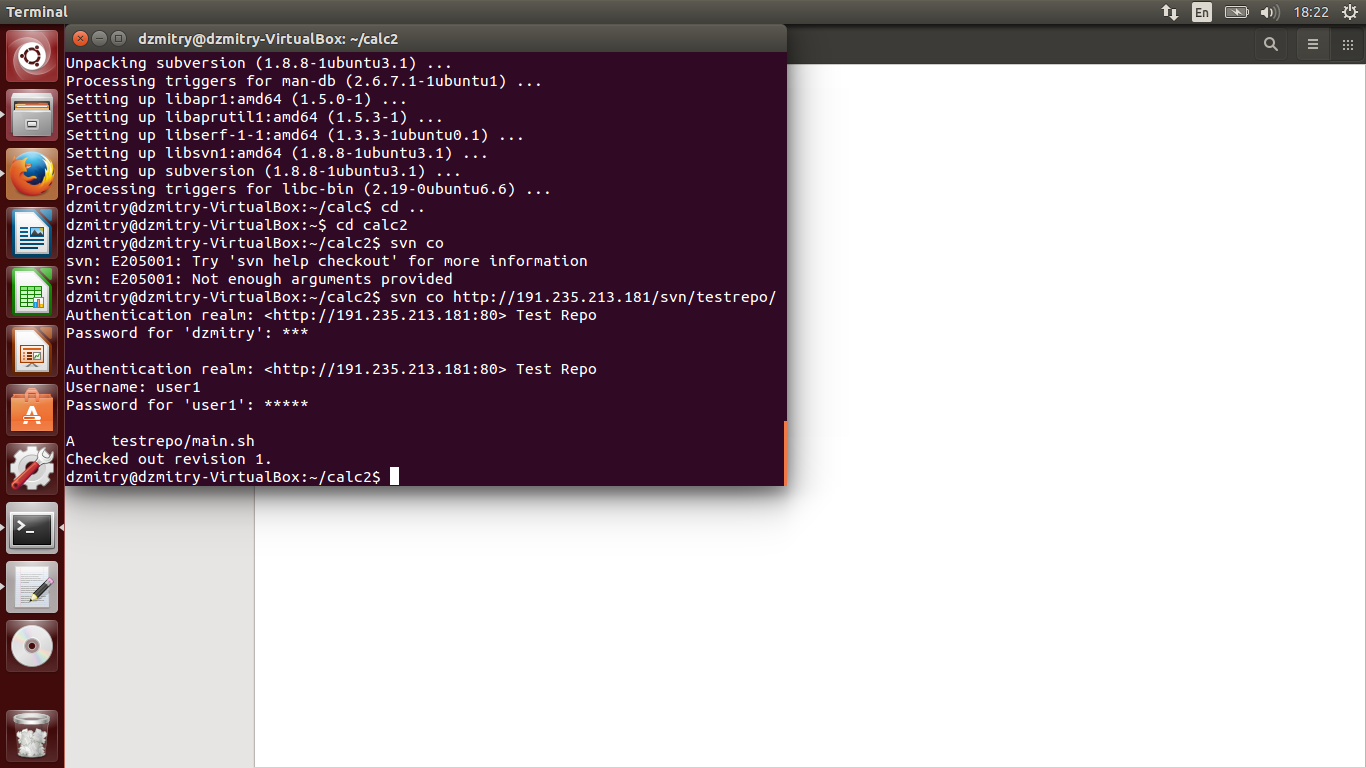


Рисунок 5.3 – Результат работы команды svn checkout

Если всё сделано правильно, то в файловой системе можно найти файлы, которые были созданы в первой ревизии, и, в случае с лабораторной работой №1, их можно будет запустить при помощи команды sh. В случае правильного выполнения скрипта, результат будет таким, как на рисунке 5.4.

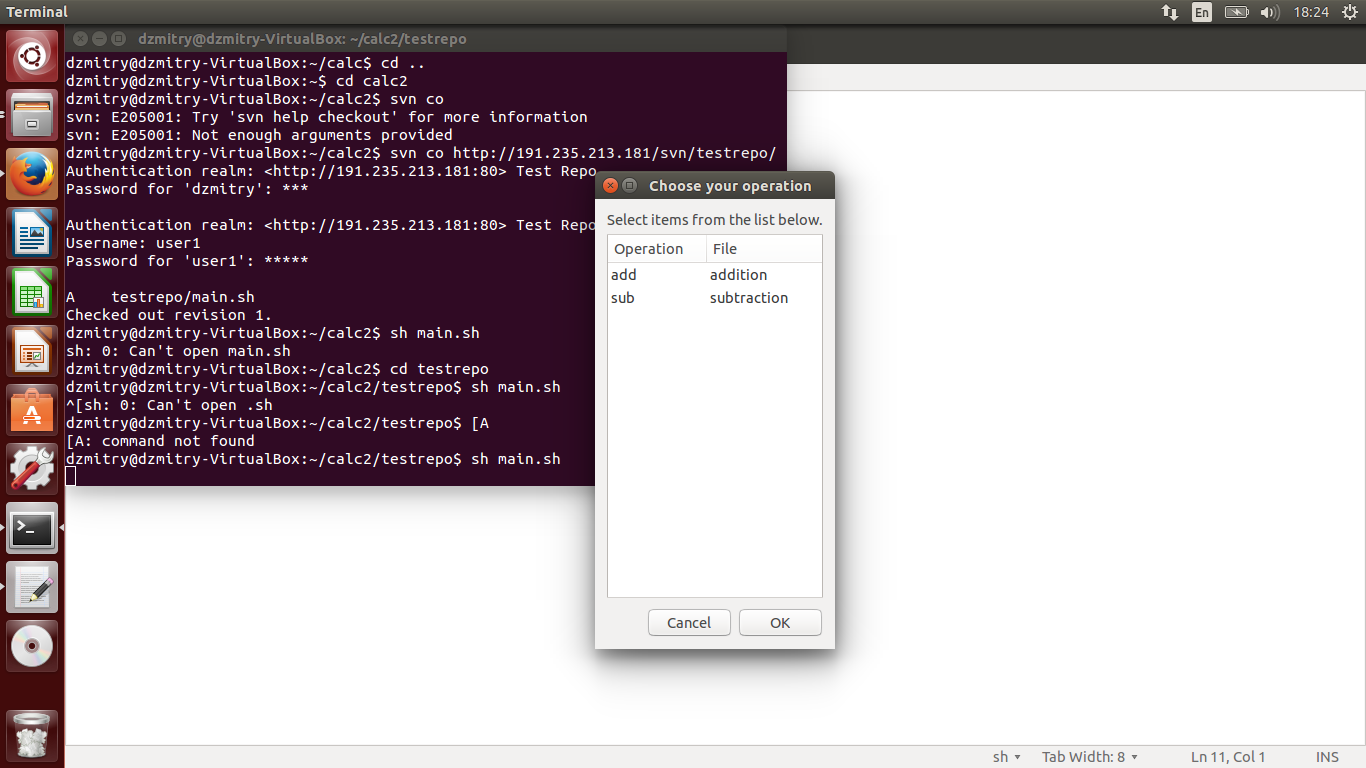


Рисунок 5.4 – Результат работы команды sh

Таким образом, у нас появился рабочий репозиторий со стартовым функционалом. Теперь изменим файл main.sh так, чтобы он выглядел как на рисунке 5.5, и добавим данный файл во вторую ревизию при помощи команды svn commit main.sh, как показано на рисунке 5.6.

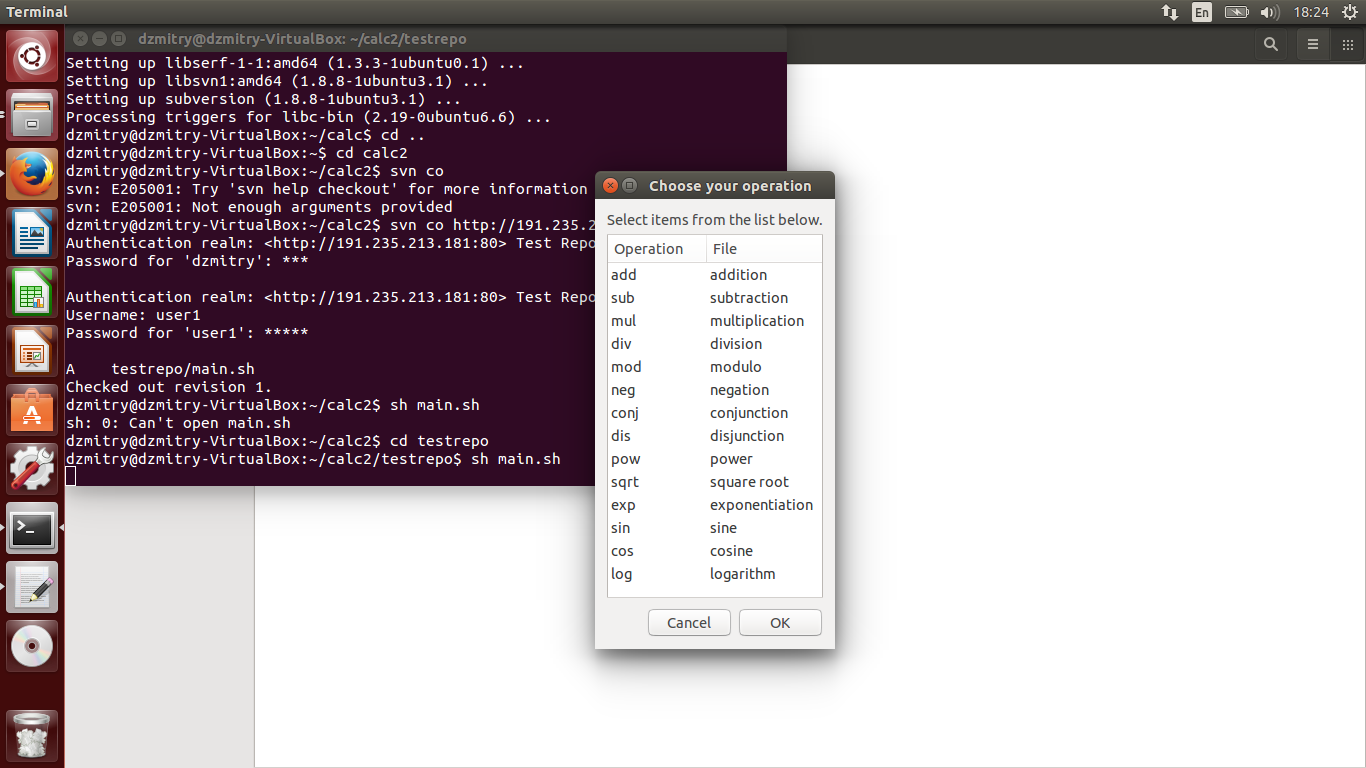


Рисунок 5.5 – Результат добавления функциональности в файл main.sh

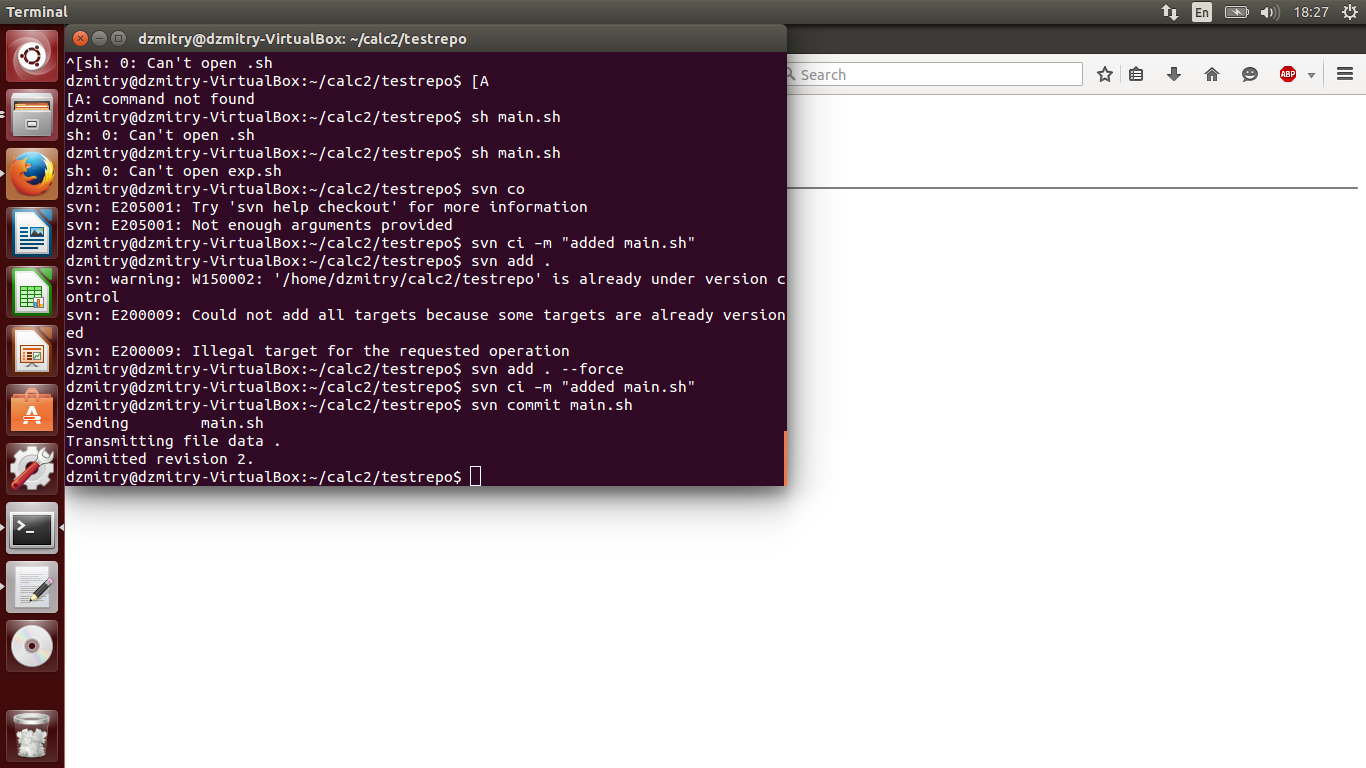


Рисунок 5.6 – Результат работы команды svn commit

Просмотреть сделанные изменения можно также на сайте; путь к изменённому файлу будет таким же, какой он был в файловой системе. Результат коммита второй ревизии на сервер можно наблюдать на рисунке 5.7.

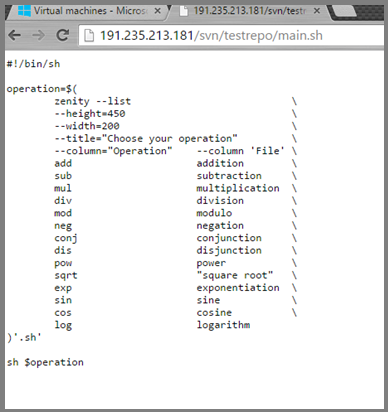


Рисунок 5.7 – Просмотр изменений файла на сервере

Теперь добавим основную функциональности к калькулятору. Результат коммита множества файлов одновременно можно наблюдать на рисунке 5.8.

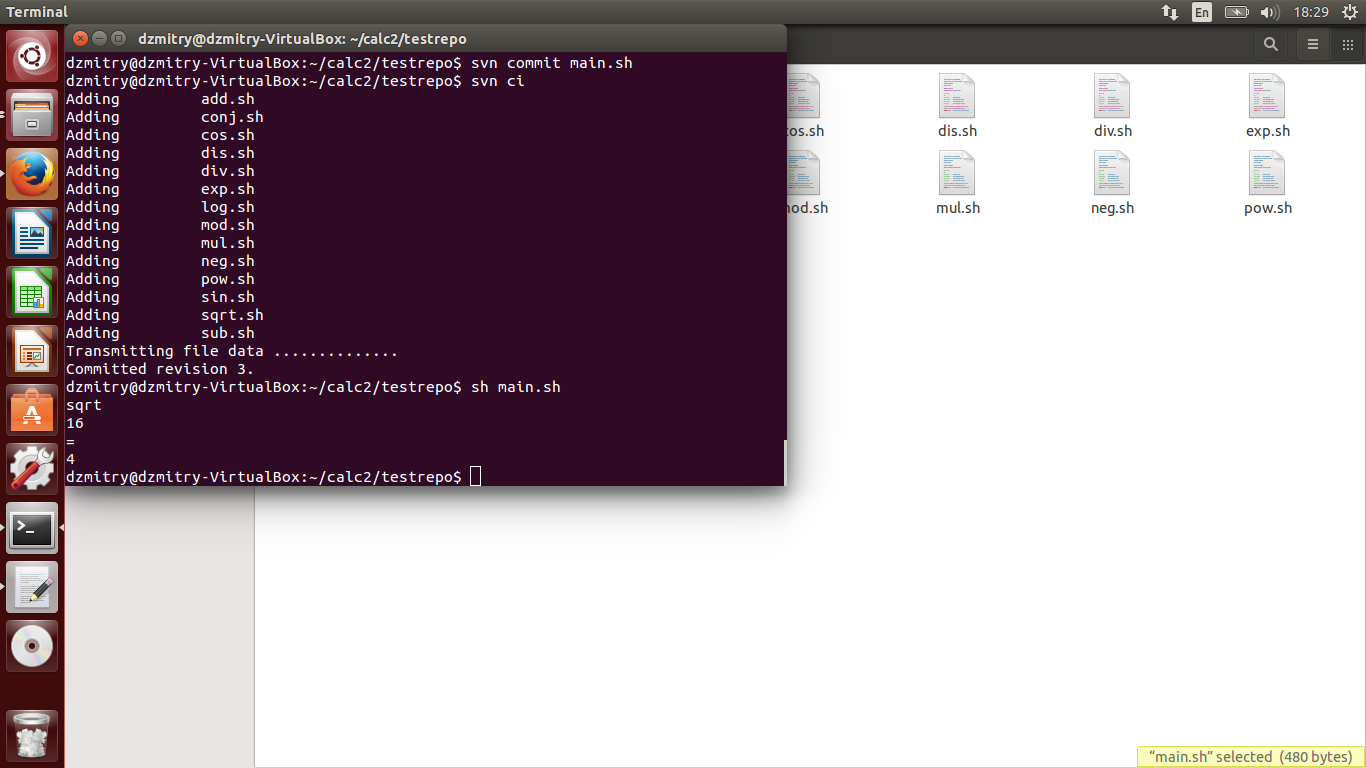


Рисунок 5.8 – Просмотр изменений файла на сервере

Результат работы созданного приложения можно увидеть на рисунке 5.9.

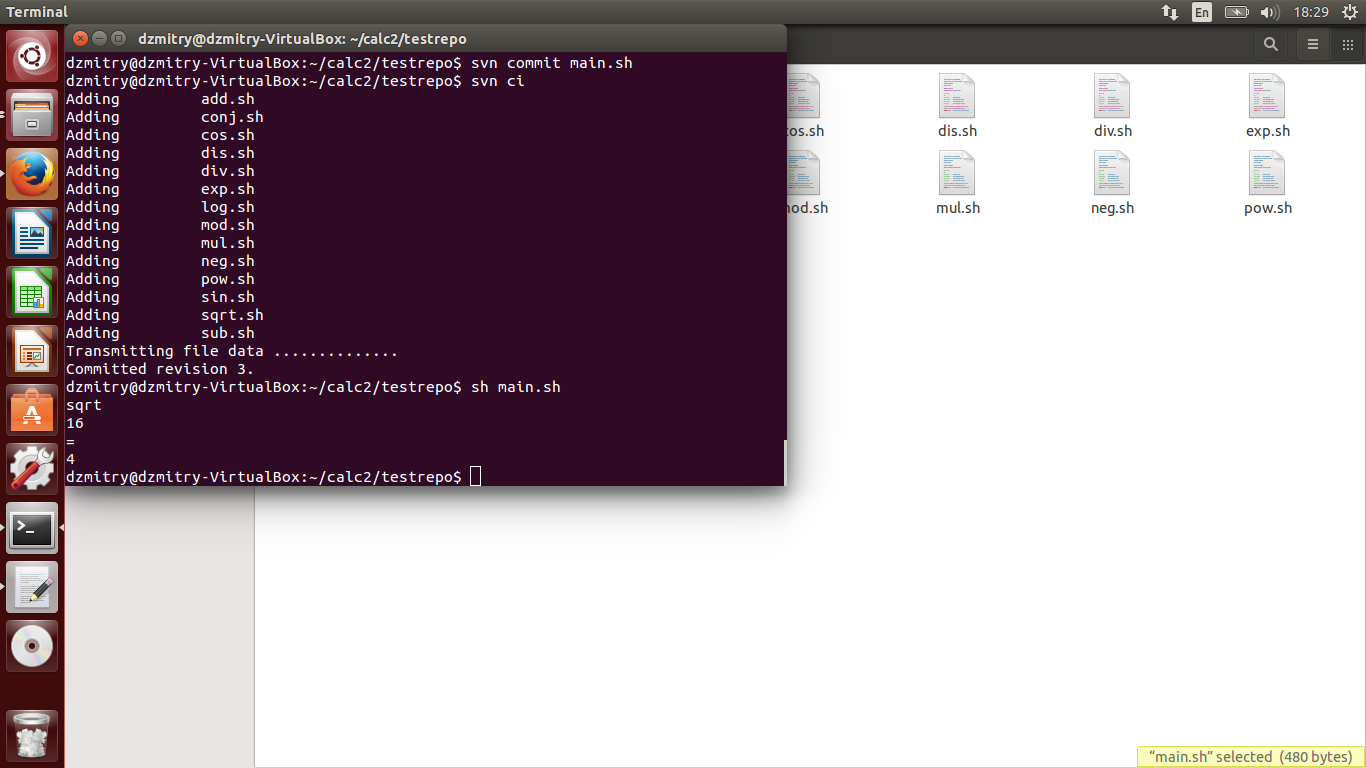


Рисунок 5.9 – Результат работы программы

Из рисунка 5.8 видно, что программа работает успешно, а из рисунка 5.7 можно сделать вывод, что в репозитории находятся 3 ревизии, каждая со своим набором изменений. Таким образом, в репозитории содержится полный набор необходимых изменений.

**5.2 Результаты разработки с Mercurial**

Пакет hg уже установлен в системе Linux Ubuntu по умолчанию, поэтому его устанавливать не нужно (если он отсутствует, можно воспользоваться командой sudo apt-get install hg). Начало работы с Mercurial-репозиторием такое же, как и в случае с Subversion, только здесь для клонирования используется команда hg clone, результат выполнения которой можно увидеть на рисунке 5.10.

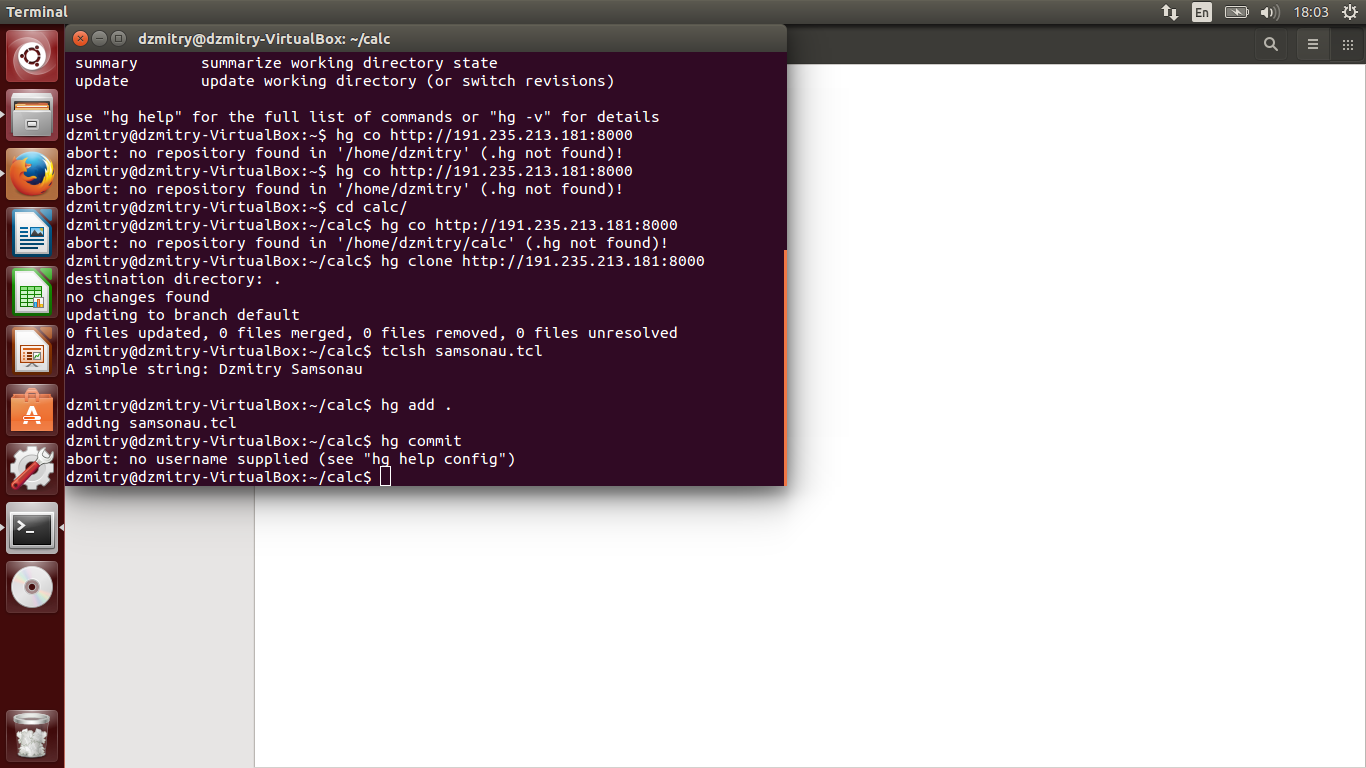


Рисунок 5.10 – Результат работы функции hg clone

Теперь напишем программу, которая выводила бы строку с именем человека, который её написал; для имени файла используем фамилию этого человека. Результат работы такой программы можно увидеть на рисунке 5.11.

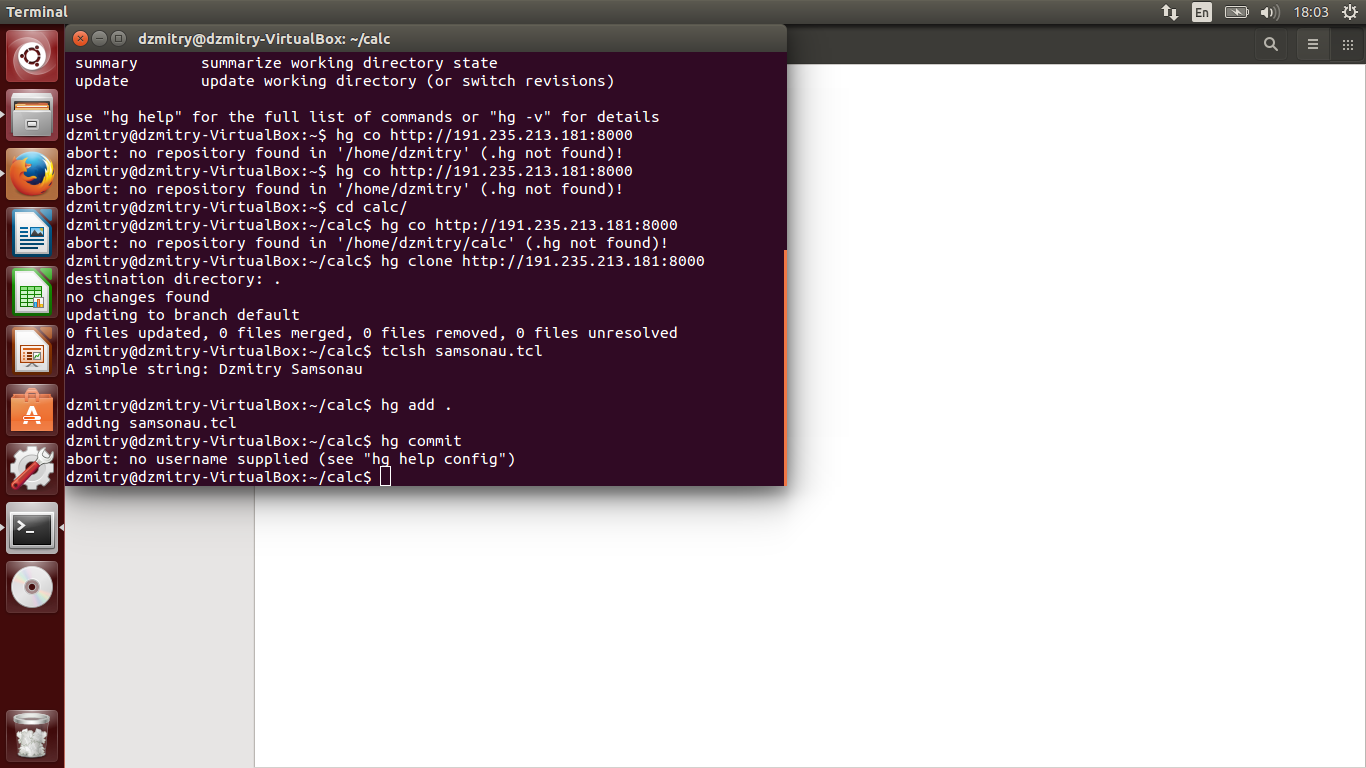


Рисунок 5.11 – Корректный результат работы для имени Dzmitry Samsonau

Добавим файл в репозиторий и выполним функцию hg commit, которая создаст набор изменений для отправки на сервер. Если на рабочей машине такая операция выполняется первый раз, то может появиться ошибка, как на рисунке 5.12.

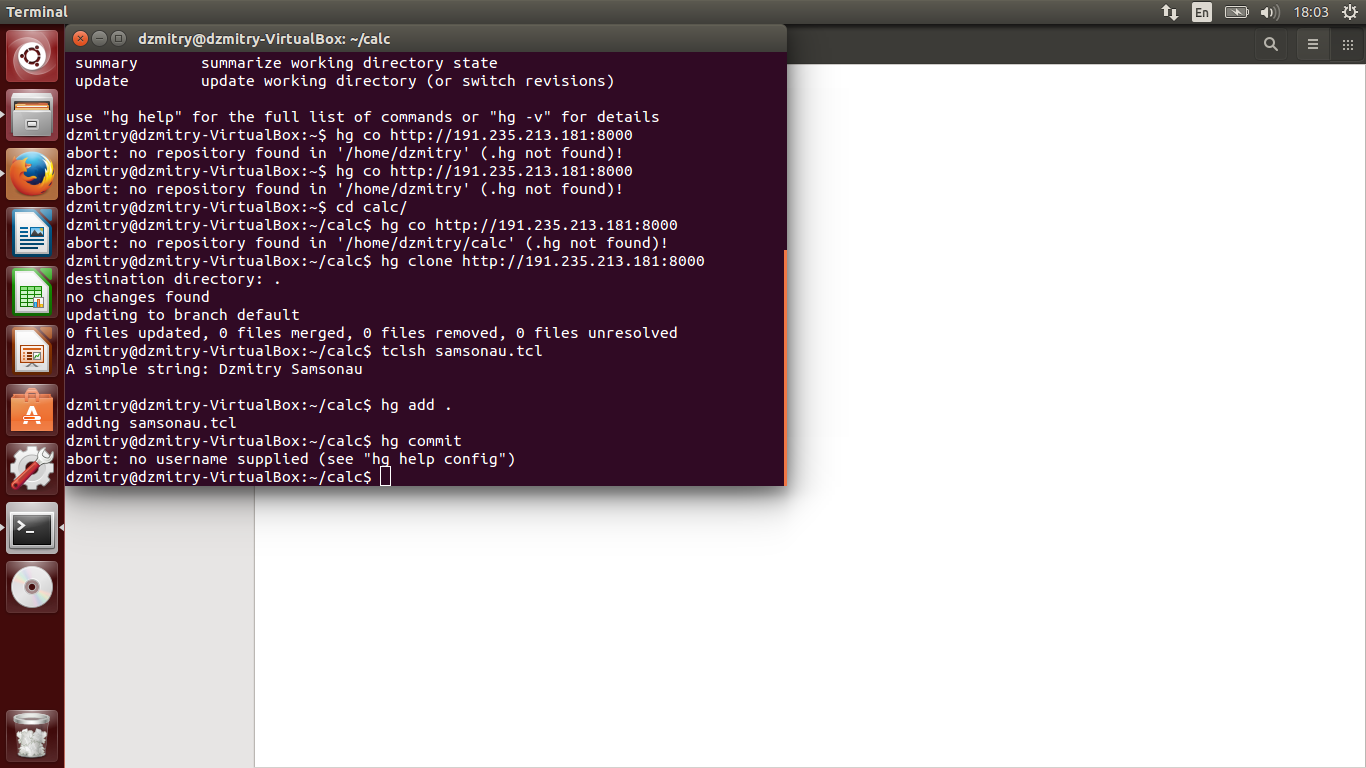


Рисунок 5.12 – Ошибка “no username supplied”

Это можно починить, добавив в файл hgrc (в папке .hg) строку с именем и email. Правильный файл hgrc представлен на рисунке 5.13.

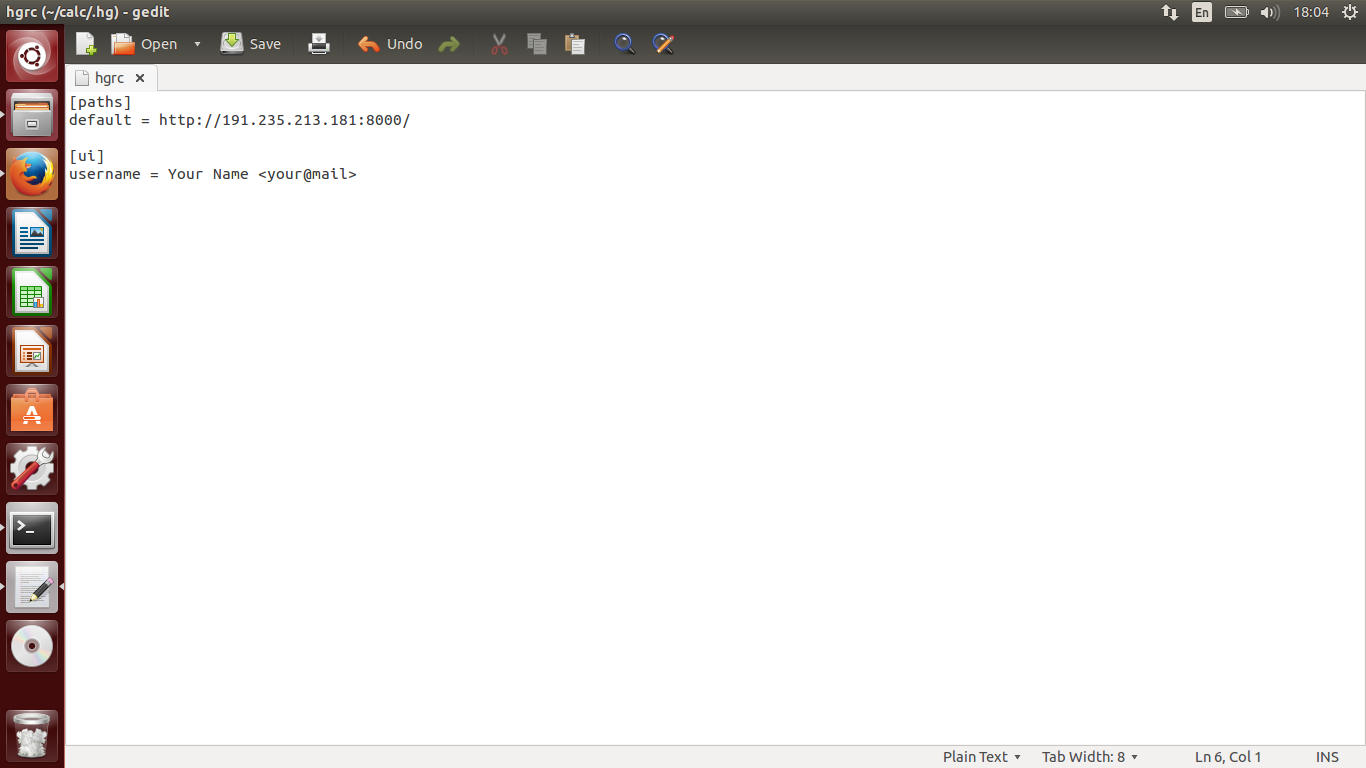


Рисунок 5.13 – Корректное содержимое файла hgrc

После этого выполняем push. Корректный результат работы этой функции представлен на рисунке 5.14.

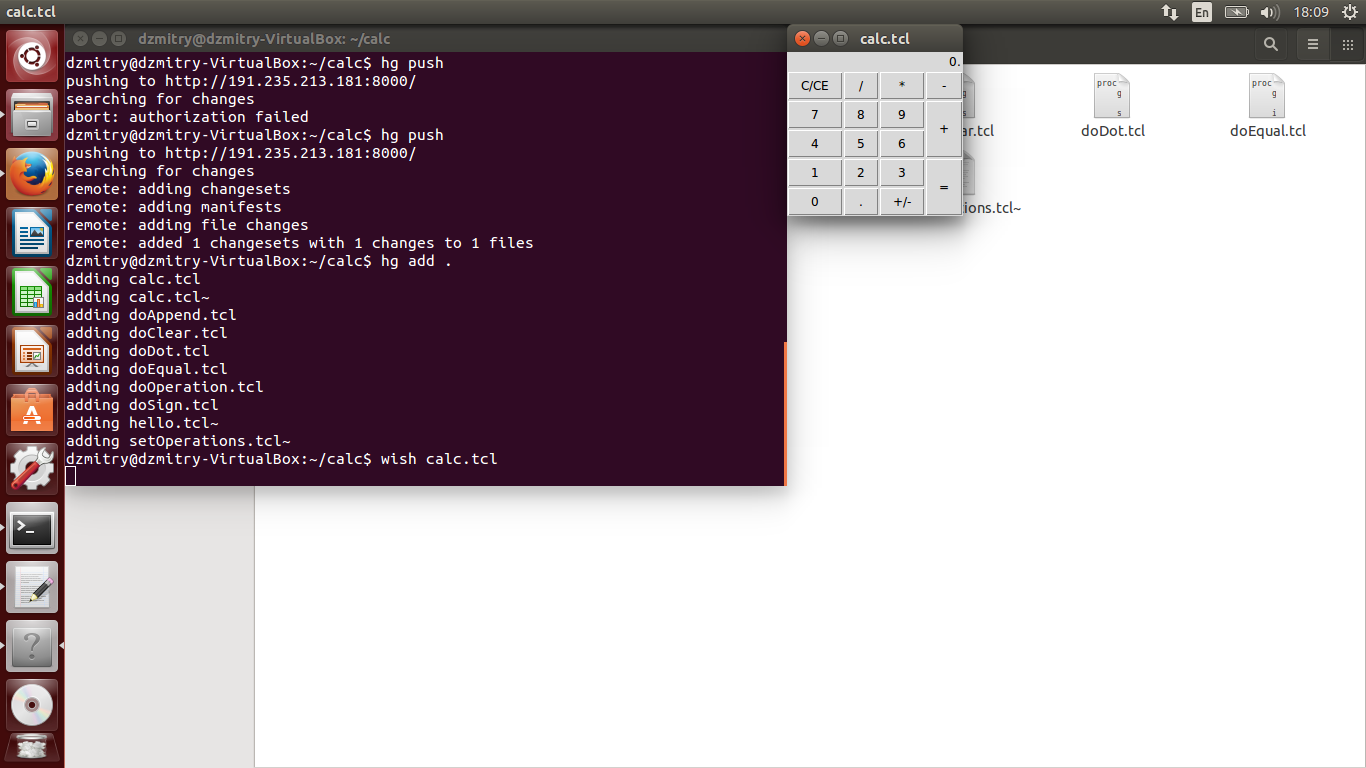


Рисунок 5.14 – Результат работы hg push

Далее необходимо по такому же принципу добавить функциональность для всех вариантов; после этого команда wish calc.tcl должна выдать на экран такое же окно, как на рисунке 5.15.

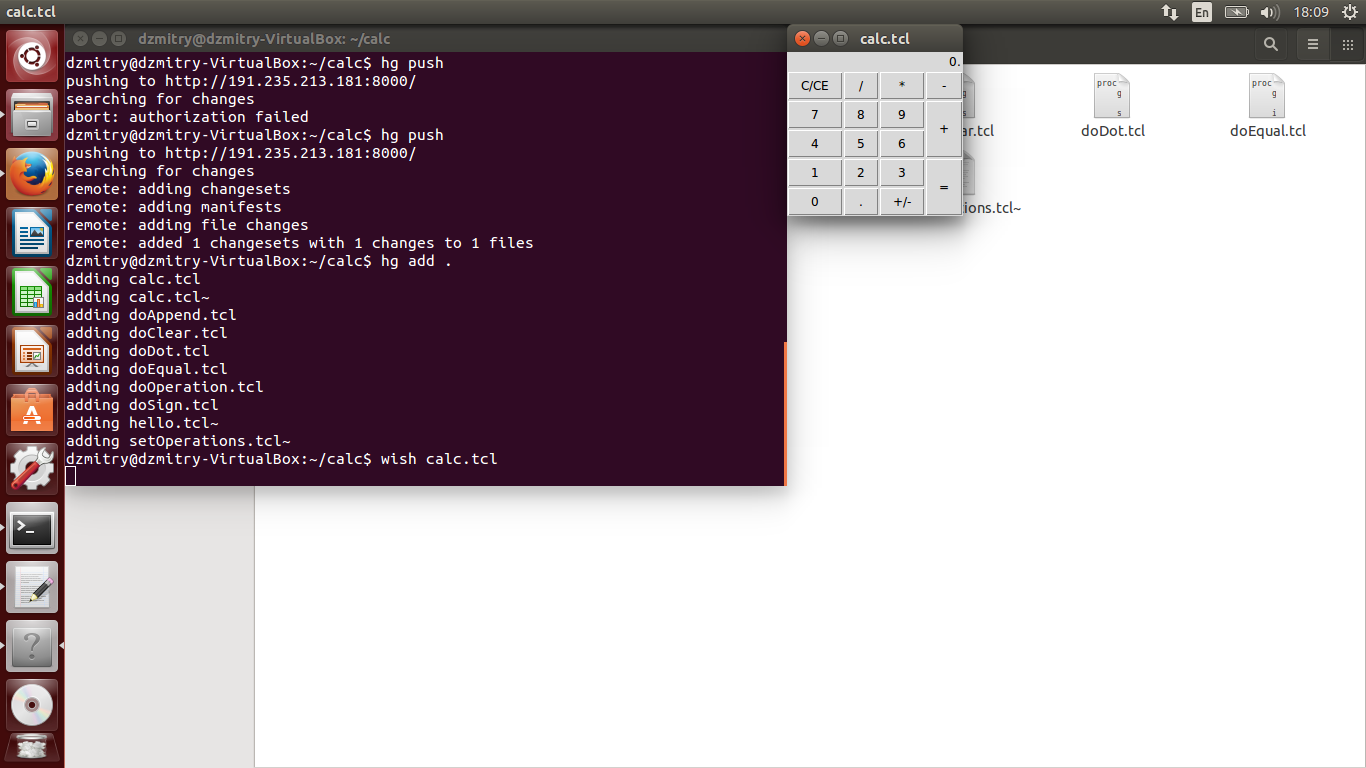


Рисунок 5.15 – Корректная работа команды wish

При правильном выполнении работы, сайт с репозиториев должен будет выглядеть, как на рисунке 5.16, показывая результаты работы и все коммиты, к которым была применена операция hg push.

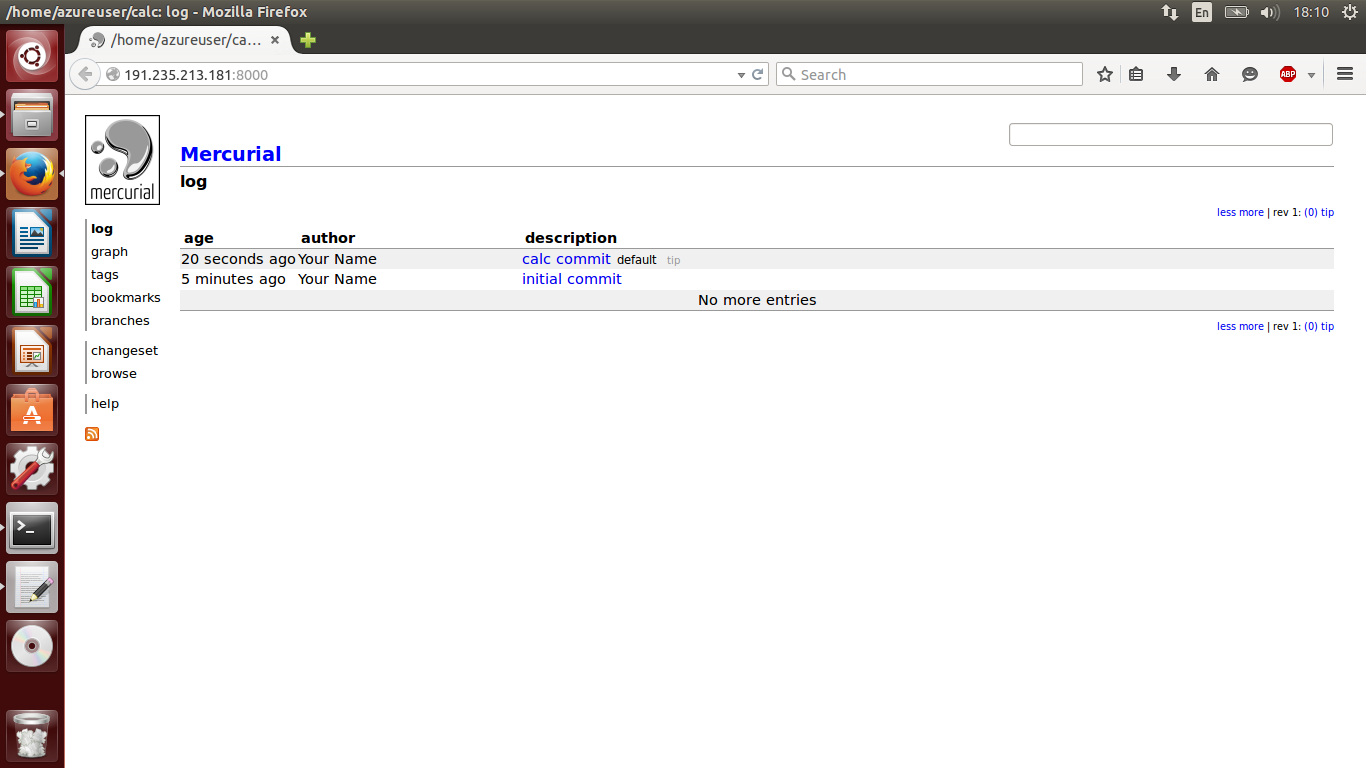


Рисунок 5.16 – Сайт Mercurial-репозитория

**5.3 Результаты разработки с Git**

Для работы с заданием по git сначала нужно добавить все необходимые страницы. Это можно сделать в самом первом коммите, результат правильного создания которого содержится на рисунке 5.17.

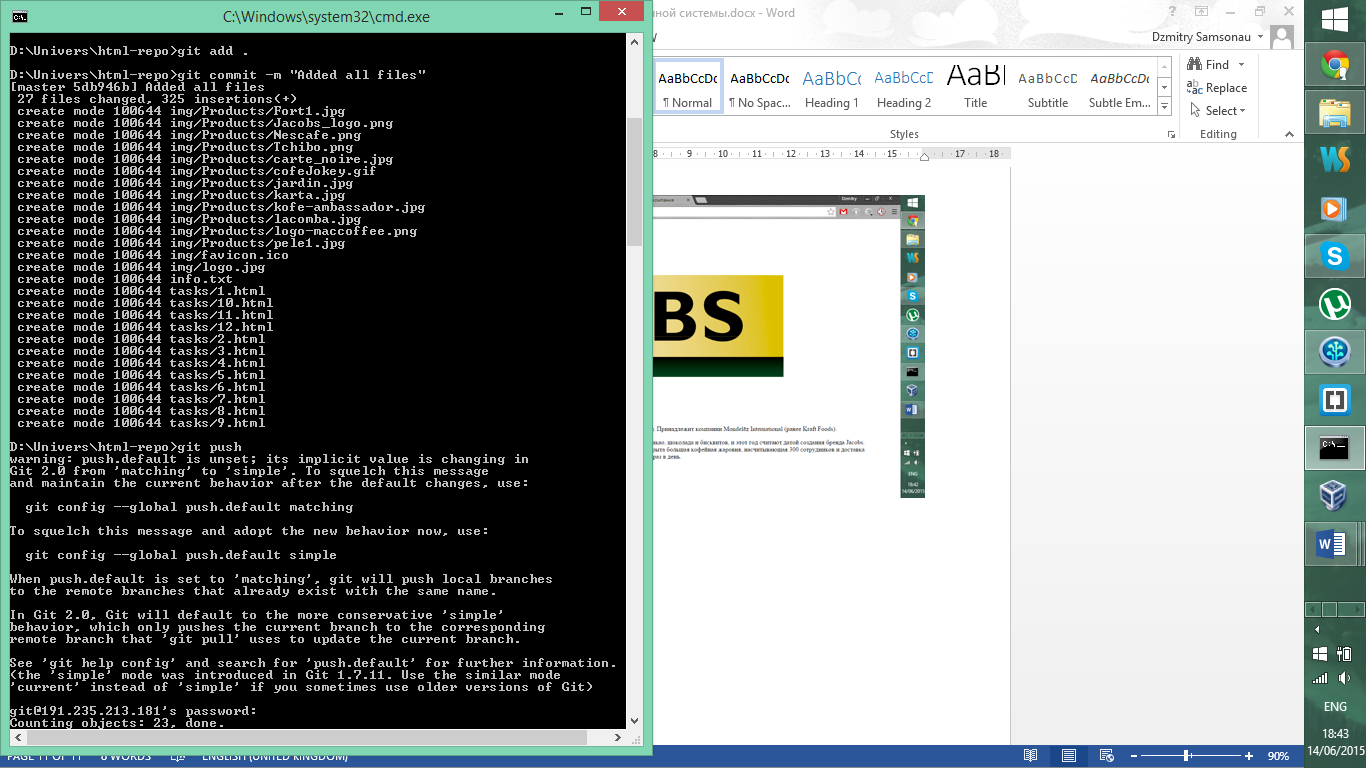


Рисунок 5.17 – Результат добавления всех необходимых страниц в локальный git-репозиторий

Результат отправки этих сообщений на сервер в консоли выглядит следующим образом (рисунок 5.18).

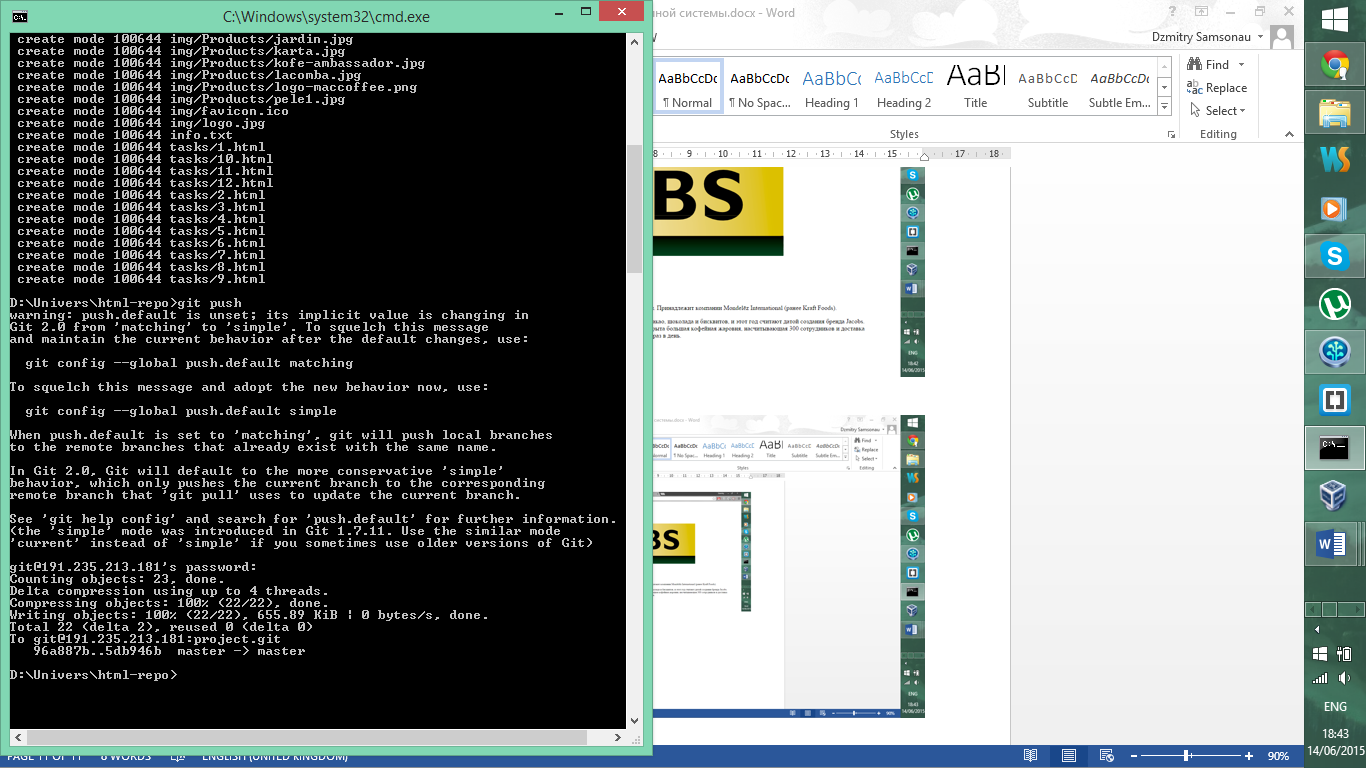


Рисунок 5.18 – Результат добавления всех необходимых страниц в git-репозиторий на сервере

Правильность выполнения первого коммита визуально проверяем на локальном сайте, который должен выглядеть, как на рисунке 5.19.

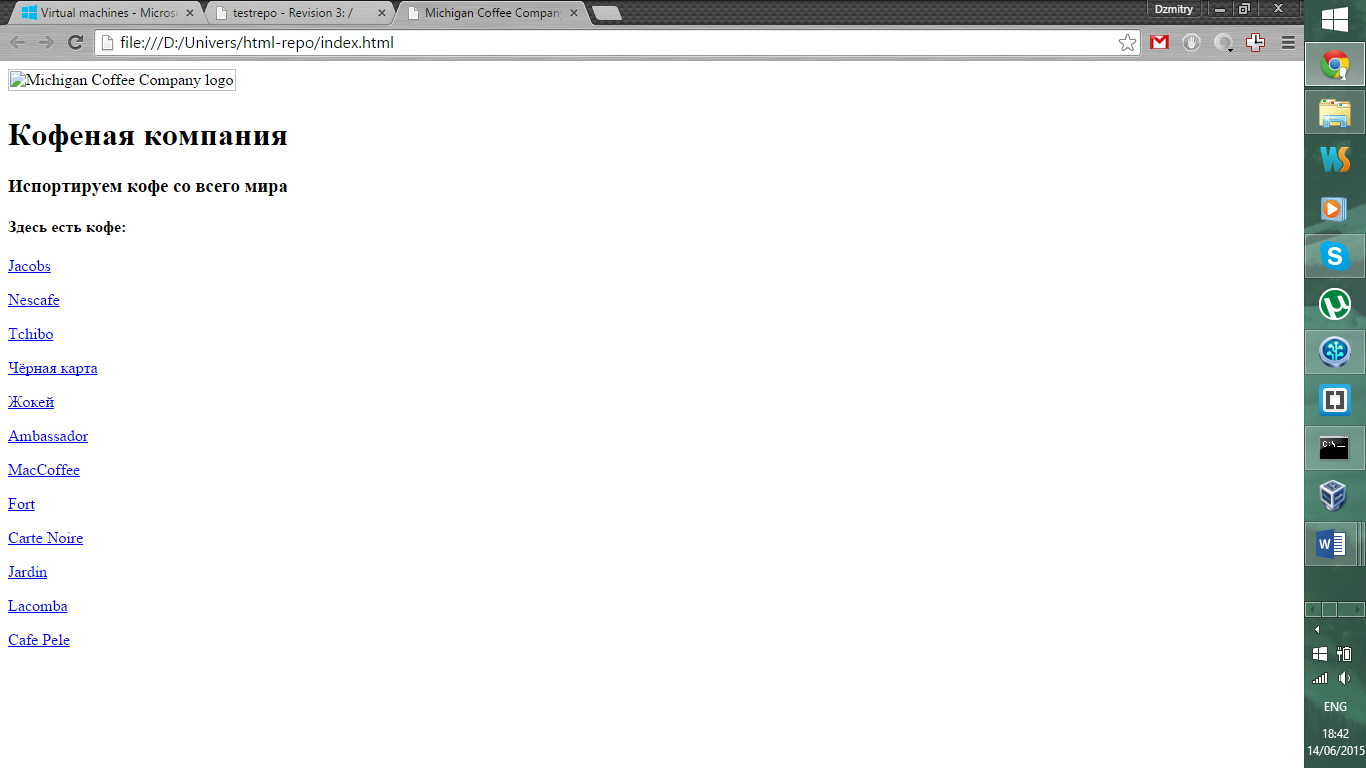


Рисунок 5.19 – Результат правильного выполнения первого коммита

Далее полный цикл работы выполнен при помощи команд git add, git commit, git remote и git push, правильная работа который показана на рисунке 5.20.

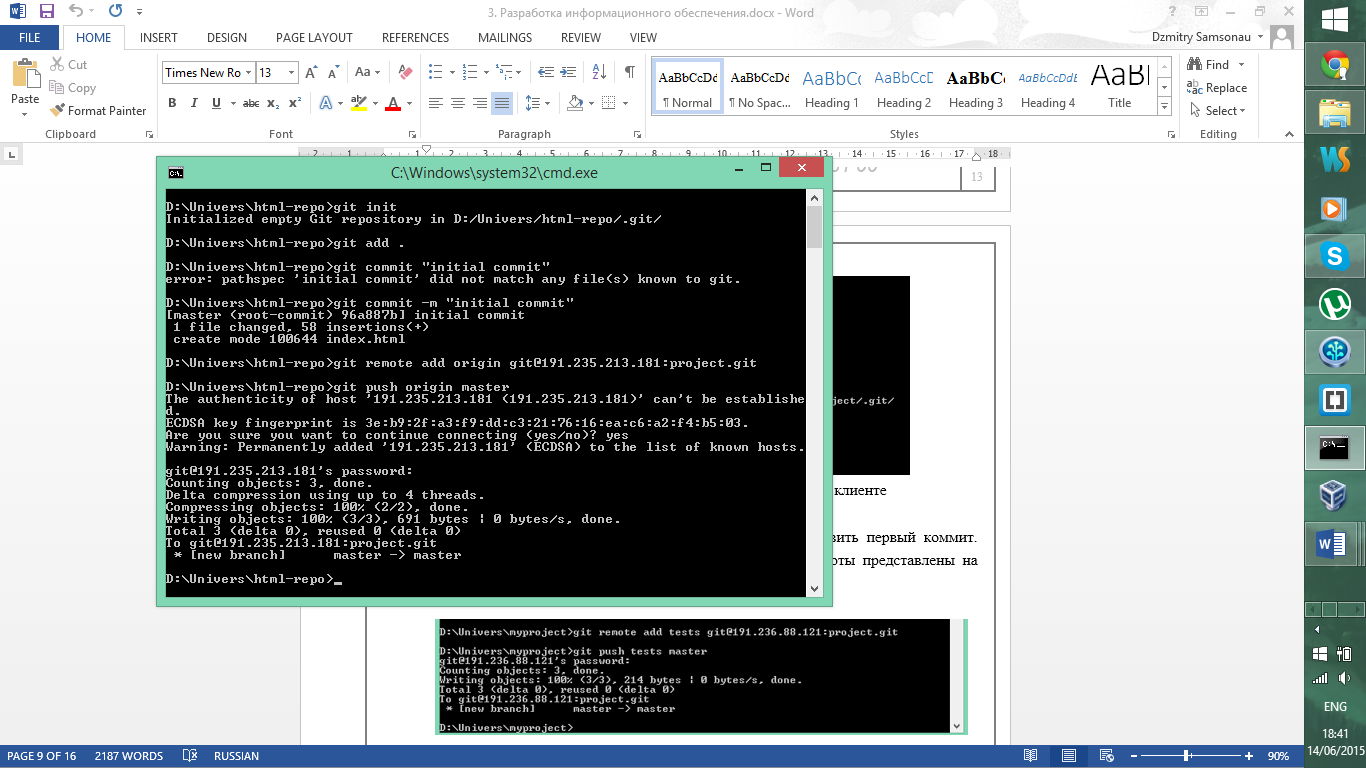


Рисунок 5.20 – Результат полного цикла работы с git-репозиторием

Результат создания корректно работающего сайта (а именно одной из его подстраниц) изображён на рисунке 5.21.



Рисунок 5.21 – Корректный вид подстраницы сайта

В отличии от SVN и Mercurial, Git не имеет функциональности добавления сайта; вместо этого для проверки результата выполнения можно попробовать воспользоваться встроенной командой git graph, а можно специальными графическими инструментами, одним из которых является SourceTree. Результат корректного выполнения части задания для варианта 1 показан на рисунке 5.22.

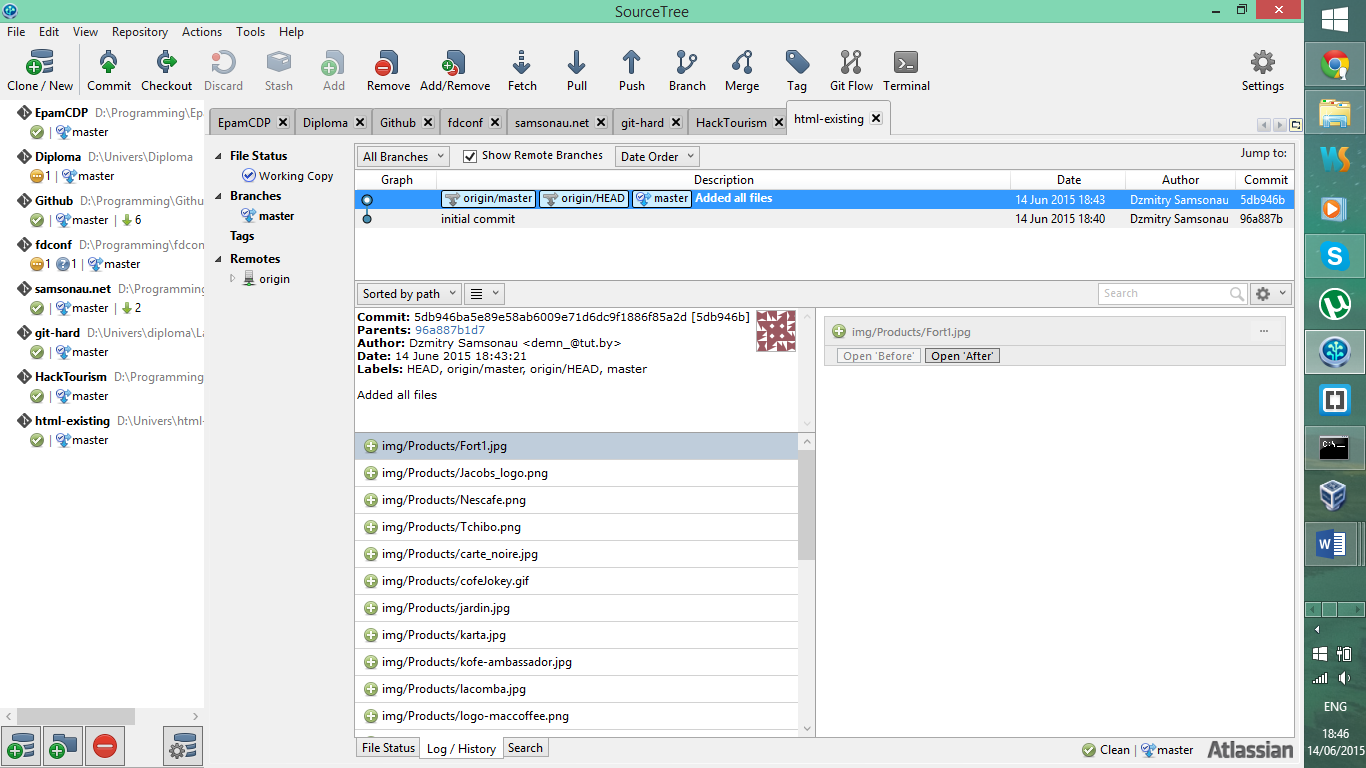


Рисунок 5.22 – Просмотр изменений в git-репозитории при корректной работе