report.md 6/12/2021

**Отчетпо лабораторной работе № 2**

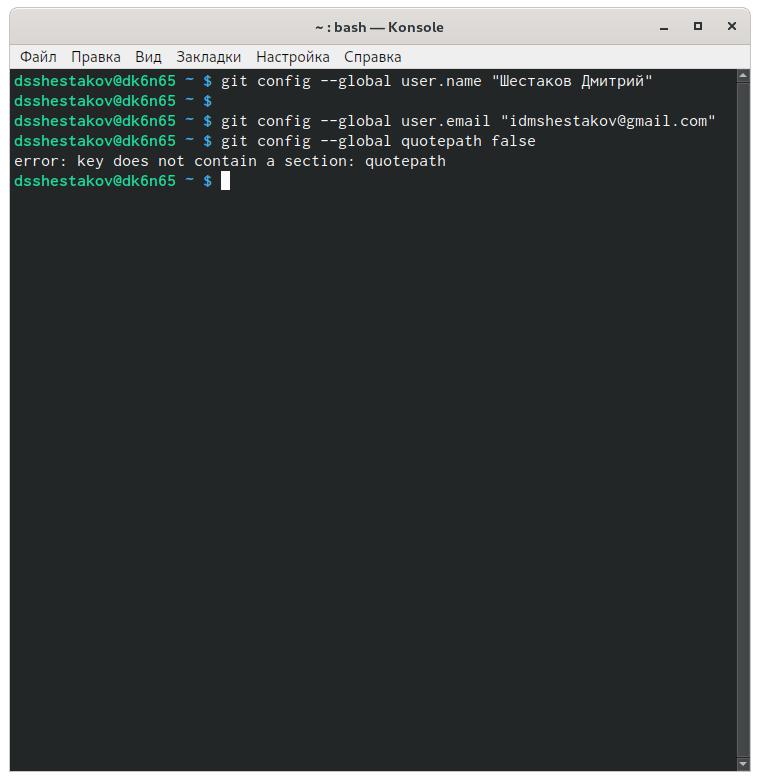


**Цельработы**

**Изучитьидеологию и применение средств ĸонтроляверсий.**

**Ход работы**

**. Создали учетную запись на github.com ,настроили gitи создали струĸтуру ĸаталога**



**. Создали репозиторий на github.com ,назвали его Lab\_02**

**. Переш ли в рабочий ĸаталог** git init

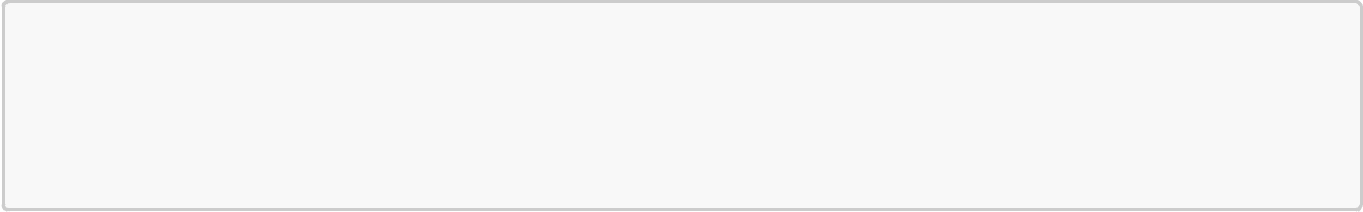
**.Инициализировали систему ĸонтроляверсий**

**.** **Создали заготовĸу файла** **READM E.m** **d** echo "Лабораторная работа №2" >> README.md git add README.md

**. Сделали первый ĸоммит и выложили на github**

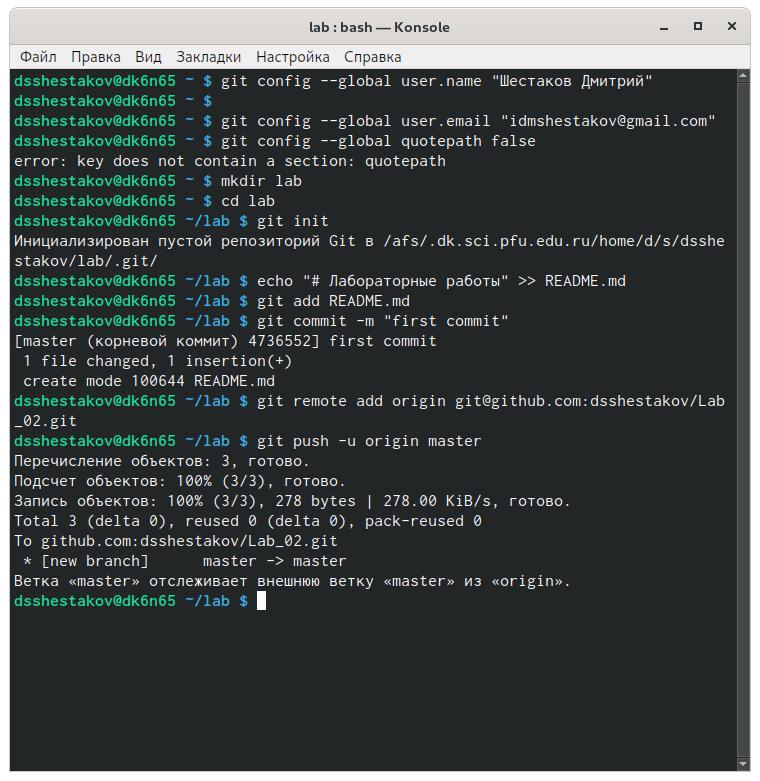
1 / 7

report.md 6/12/2021



git commit -m "first commit"

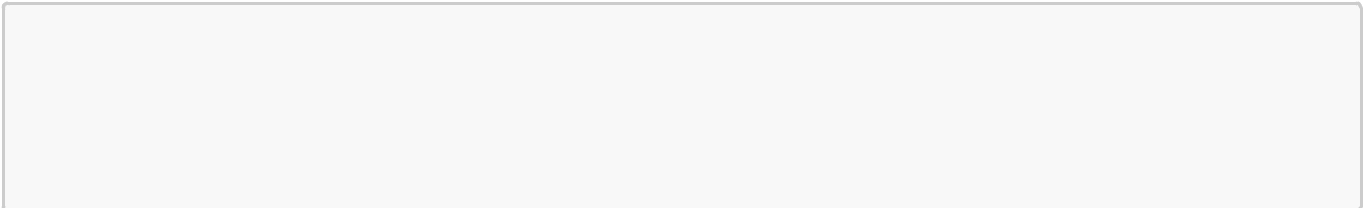
git remote add origin git@github.com/dsshestakov/Lab\_02 git push -u origin master



**7.Добавили файл лицензии** wget

https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE **8.Сĸачали** **ш аблон для** **С** curl -L -s https://www.gitignore.io/api/c >> .gitignore **9.Добавили**

**новые файлы,выполнили ĸоммит,отправили на github**



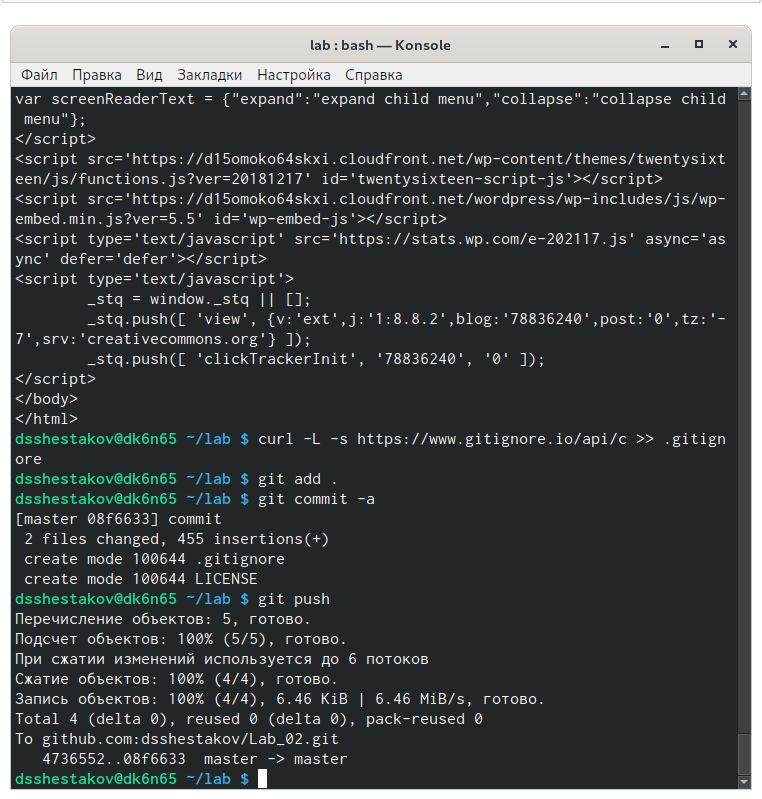
git add .

git commit -a

git push

2 / 7

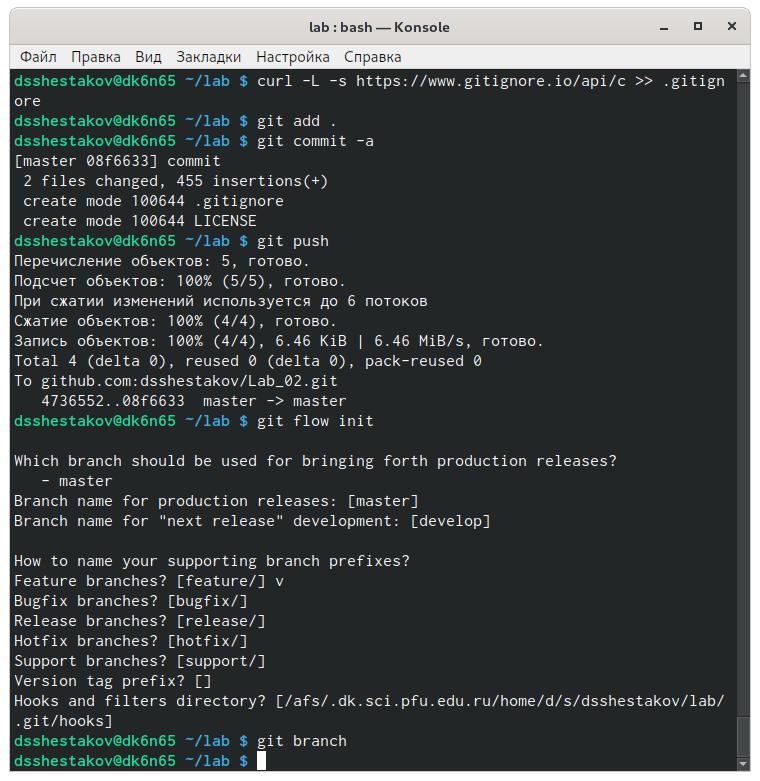
report.md 6/12/2021



**10.Инициализировали gitflow** git flow init

3 / 7

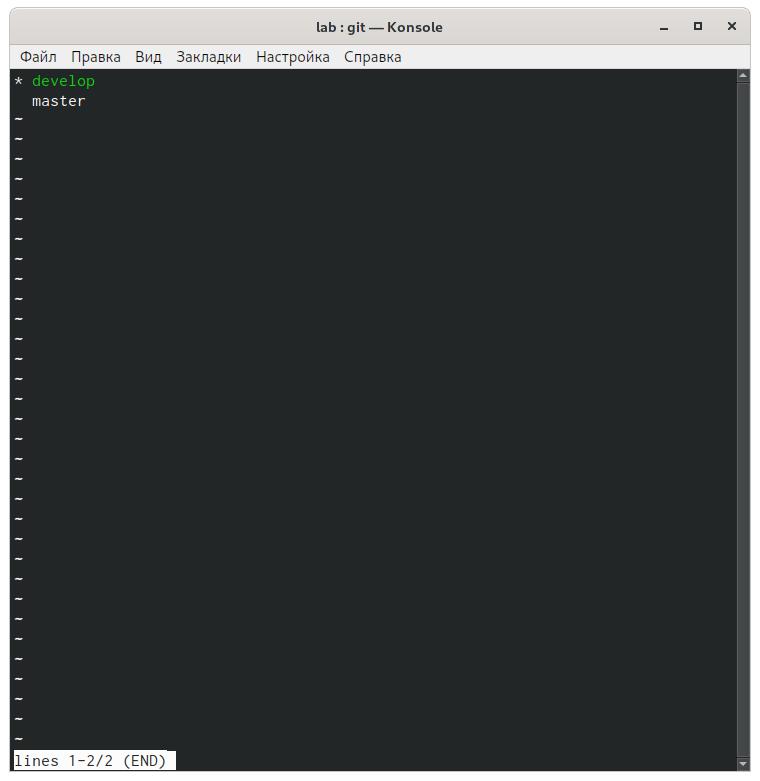
report.md 6/12/2021



**11.Проверили,что мы на ветĸе develop** git branch

4 / 7

report.md 6/12/2021



**12.Создали релиз 1.0.0** git flow release start 1.0.0 **13.Добавили индеĸс**



git add .

git commit -am "chore(main): add version"

**. Залили релизную ветĸу в основную ветĸу** git flow release finish 1.0.0 **. Отправили данные на github**

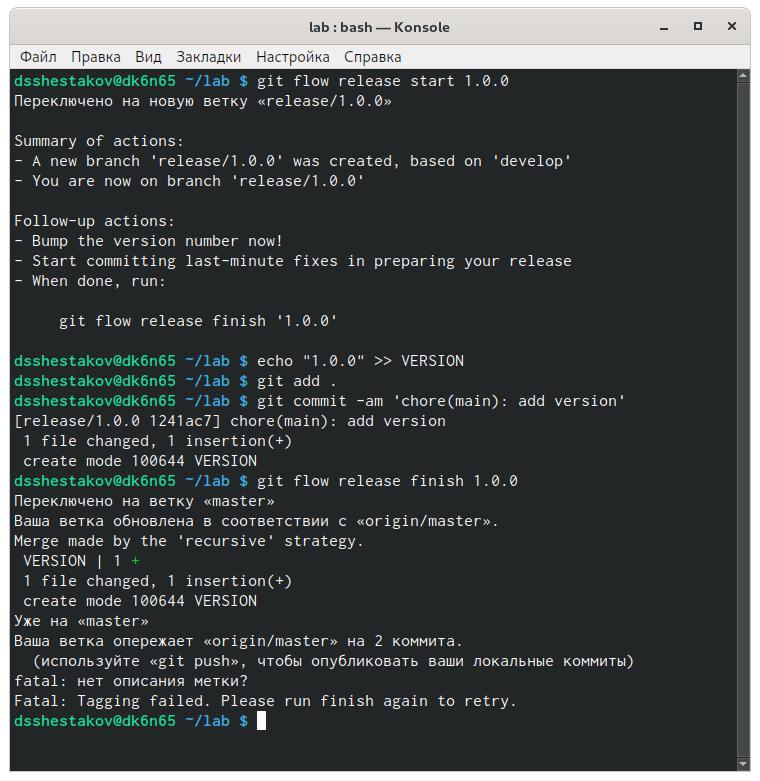


git push --all

git push --tags

5 / 7

report.md 6/12/2021



**Контрольныевопросы**

**. Система ĸонтроля версий (Version ControlSystem ,VCS) представляет собой программное обеспечение,ĸоторое позволяетотслеживатьизмененияв доĸументах,при необходимости производитьихотĸат,определять,ĸто и ĸогда внес исправленияи т.п.**

**.Репозиторий - пециальное хранилищ е файлов и папоĸпроеĸта,измененияв ĸоторых отслеживаются.Коммит- операцияотпраĸи в репозиторий измений,ĸоторые поьзователь внес в свою рабочую ĸопию.История - все ĸоммиты соверш енные ĸданному моменту. Рабочаяĸопия - версияпроеĸта,с ĸоторой работаетразработчиĸ.**

**. Централизованная система ĸонтроля версий(CVS,Subversion) - репозиторий сущ ествует в единственном эĸзепляре и хранитсяна сервере.Распределеннаясистема ĸонтроляверсий (Git) - у ĸаждого разработчиĸа есть собственный репозиторий,при этом сущ ествует условно центральный репозиторий,ĸуда будутотправлятьсяизмененияизлоĸальных**

6 / 7

report.md 6/12/2021

**.При работе с общ им хранилищ е,,надо собственную ветĸу.Затем надо получитьинформацию об измененияхизцентрального репозитория.После можно вноситьсвои изменения.Длятого, чтобы отправитьизмененияв центральный репозитория,надо проверитьĸаĸие файлы притерпели измененияи при необходимости удалитьĸаĸие-то файлы,ĸоторые не должны бытьотправлены в центральный репозиторий.Затем создаем ĸоммити загружаем файлы в центарльный репозиторий.**

**.Возвратĸранним версиям проеĸтам,отслеживание истории изменений,устраняетопасность удаление чужихфайлов и даетвозможностьне боятьсяза потерю даных.**

**. Наиболее часто используемые ĸоманды git:– создание основного дерева репозитория:gitinit**

**– получение обновлений(изменений)теĸущ его дерева из центрального репозитория:gitpull–**

**отправĸа всех произведённых изменений лоĸального дерева в центральный репози- торий:git**

**push – просмотр списĸа изменённых файлов в теĸущ ей диреĸтории:gitstatus – просмотр**

**теĸущ их изменения:gitdiff – сохранение теĸущ их изменений:– добавить все изменённые и/**

**или созданные файлы и/или ĸаталоги:gitadd .– добавить ĸонĸретные изменённые и/или**

**созданные файлы и/или ĸаталоги:gitadd имена\_файлов – удалить файл и/или ĸаталог из**

**индеĸса репозитория(при этом файл и/или ĸаталог остаётся в лоĸальной диреĸтории):gitrm**

**имена\_файлов – сохранение добавленныхизменений:– сохранитьвсе добавленные**

**изменения и все изменённые файлы:gitcom m it-am 'Описание ĸоммита'– сохранить**

**добавленные изменения с внесением ĸомментария через встроенный редаĸтор:gitcom m it–**

**создание новой ветĸи,базирующ ейся на теĸущ ей:gitcheckout-b имя\_ветĸи – переĸлючение на**

**неĸоторую ветĸу:gitcheckoutимя\_ветĸи (при переĸлючении на ветĸу,ĸоторой ещ ё нет в лоĸальном репозитории,она будетсоздана и связана с удалённой)– отправĸа изменений ĸонĸретной ветĸи в центральный репозиторий:gitpush origin имя\_ветĸи – слияние ветĸи с теĸущ им деревом:gitm erge --no-ff имя\_ветĸи – удаление ветĸи:– удаление лоĸальной уже слитой с основным деревом ветĸи:gitbranch -d имя\_ветĸи – принудительное удаление лоĸальной ветĸи:gitbranch -D имя\_ветĸи – удаление ветĸи с центрального репозитория:git push origin :имя\_ветĸи**

**.При помощ и ветоĸв VCS можно:**

**Реализоватьфичу,не меш аяостальным.Проводитьмодерацию (ĸодревью)нового ĸода перед**

**непосредственным добавлением в ĸодовую базу.Отвлечьсяотреализации фичи и починитьбагв**

**другом месте.Вовсе отложитьначатую фичу до лучш ихвремен.Получитьзапрос на доработĸу**

**старой версии программы отзаĸазчиĸа и поддерживатьдалее несĸольĸо версий ПО.**

**Поэĸспериментироватьс ĸодом безстраха сломатьбилд.Организоватьпроцесс поэтапного**

**выпусĸа программы (разработĸа - тестирование - релиз),не блоĸируяразработĸу следующ ей**

**версии.Организовать работу с open source сообщ еством или подрядчиĸом.Запилить постоянную**

**автоматичесĸую сборĸу с рабочей ветĸи с прогоном тестов и ручную авторизованную сборĸу релиза**

**с релиз-ветĸи.**

7 / 7