

## **NAPREDNO WEB PROGRAMIRANJE**

**Tema:**  
**GitHub**

Mentor: dr Nenad Koji

Student: Stevan Nikoli

## Sadržaj

1.0 Uvod.....	3
1.1 O GIT-u.....	3
2.0 Instalacija Git klijenta.....	4
2.1 Konfiguracija GitBash-a .....	8
2.2 Rad sa gitBash-om. ....	13
3.0 Registracija .....	20
4.0 Rad sa udaljenim repozitorijumom.....	28
5.0 Windows GUI GitHub aplikacija .....	34
5.1 Kreiranje novog projekta .....	35
6.0 Branching, forking i merging. ....	39
6.1 Branching primer u radu sa GitBash-om .....	39
6.2 Grananje u windows GUI aplikaciji.....	42
6.2 Forking.....	44
7.0 Zaključak.....	48

## 1.0 Uvod

U ovom radu će biti reči o alatu za praćenje i kontrolu verzija programa koji se zove Git. Git je sistem za distribuiranu kontrolu verzije i sistem za upravljanje fajlovima. Biće razmatrane neke sposobnosti ovog softvera i komande koje se koriste u radu sa njim. U konkretnim primerima će biti pokazano kako se koristi konzolna i aplikacija sa grafičkim korisničkim interfejsom.

## 1.1 O GIT-u

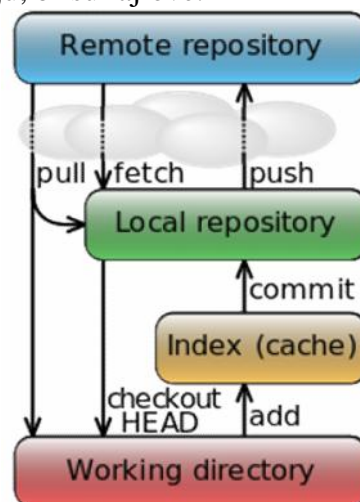
GUI sakriva važne akcije, a to je važno za razumevanje kako cela stvar radi.

Slučajevi u kojima je potrebno korišćenje softvera za kontrolu verzija programskog koda:

- U slučaju da je napravljena promena u kodu, a potrebno je vratiti se u prethodno stanje kada je kod funkcionisao pravilno.
- Kada je potrebno održavanje više verzija koda.
- Kada je potrebno videti razlike između dve izmene.
- Kada želimo da testiramo da li je određena verzija neispravna ili popravljena.
- Kada želimo da vidimo koliko dugo je bug postojao.
- Kada je potrebno eksperimentisati bez izmene trenutno aktuelnog koda.

Prednosti:

- Svako dobija svoju kopiju projekta što znači da je git vrlo brz i radi lokalno (offline).
- Promene koda i spajanje promena.
- Developeri modifikuju kod.
- Developeri dodaju, pomeraju, brišu fajlove.



Itav proces rada sa Git-om se sastoji iz nekoliko slojeva, te je potrebno opisati te slojeve i nazive nekih termina koji se svakodnevno koriste u radu sa Git-om:

Commit je tačka u vremenu na koju se možemo vratiti. Svaki put kada se izvrši neka izmena ta izmena se pošalje (commituje) u lokalni repozitorijum gde se lokalno pamte sve izmene.

Keš/priprema za slanje (staging area) – pre slanja u lokalni repozitorijum potrebno je odrediti koji fajlovi biti poslani, tj. postaviti ih na listu za slanje. Kada su odabrani fajlovi ije izmene će biti praćene, pokrene se commit komanda.

Lokalni repozitorijum (local repository) sadrži sve zabeležene promene na lokalnom računaru. Kada korisnik poželi da te promene podeli sa drugim korisnicima onda se te promene šalju sa lokalnog na udaljeni (remote) repozitorijum. Ukoliko korisnik želi da preuzme promene koje je neko drugi postavio na Github nalog, onda će skinuti sve promene sa udaljenog (remote) repozitorijuma na lokalni (local) repozitorijum.

## 2.0 Instalacija Git klijenta

Najpre e biti re i o softveru koji se zove GitBash. Ovo je softver za komunikaciju sa GitHubom koriš enjem konzolnih komandi. Vrlo je važno prvo nau iti konzolne komande, a tek potom koristiti aplikacije sa grafi kim korisni kim interfejsom jer ove potonje skrivaju šta se u stvari dešava iza kulisa, tj. korisnik ne zna koji se procesi zaista dešavaju u vizuelnoj aplikaciji dok sam ne nau i komande koje razume Git.

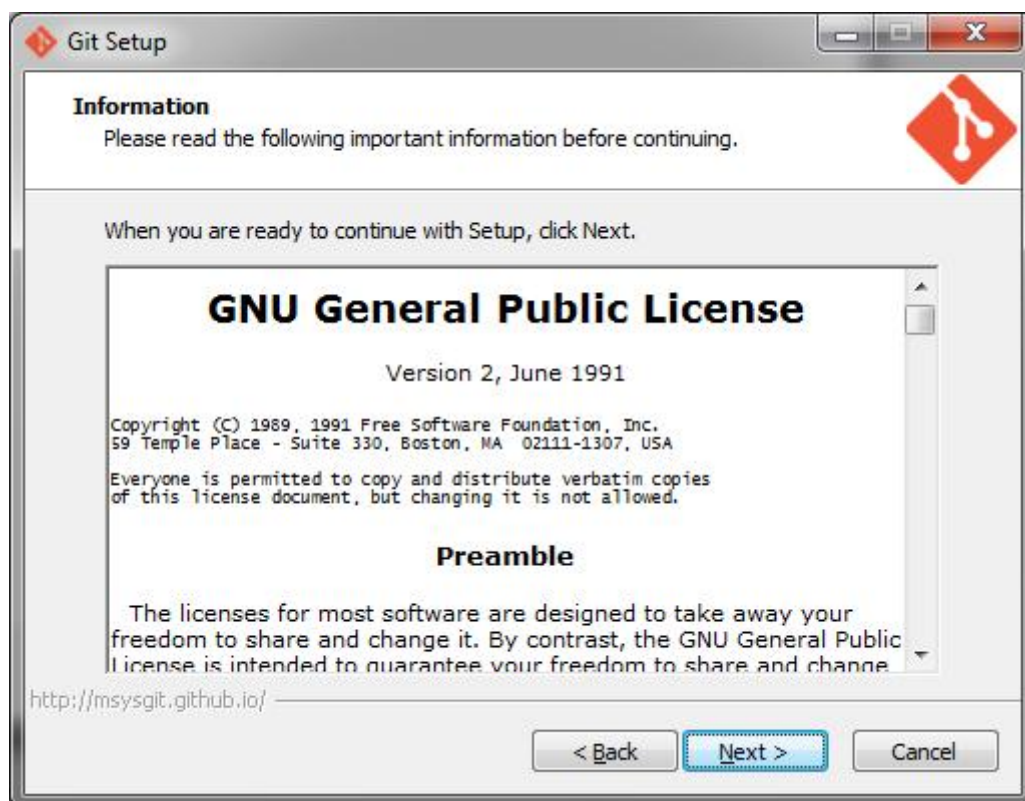
Gitbash se preuzima sa sajta: <http://www.git-scm.com/>. U donjem desnom uglu stranice se može kliknuti na link „Download for Windows“ da bi se skinula najnovija verzija za windows.



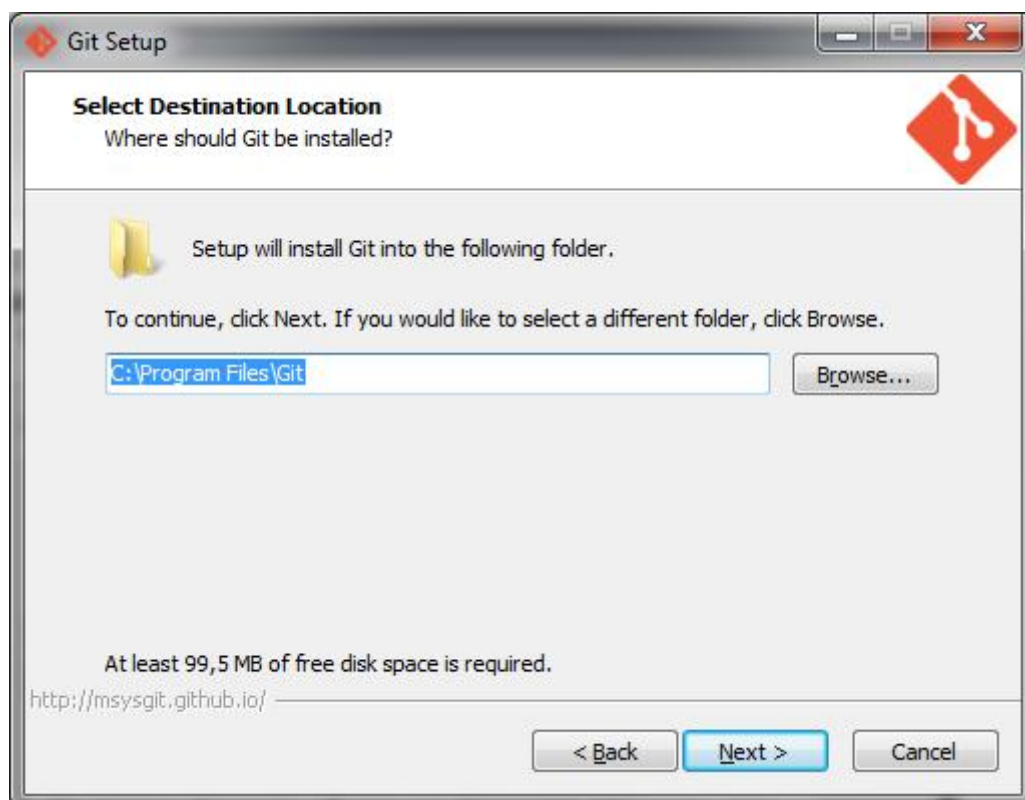
Nakon što je preuzet instalacioni fajl treba ga pokrenuti i pratiti dalje uputstvo. Pojavi e se prozor koji uvodi korisnika u instalaciju GitBash-a. Potrebno je kliknuti na dugme **Next**.



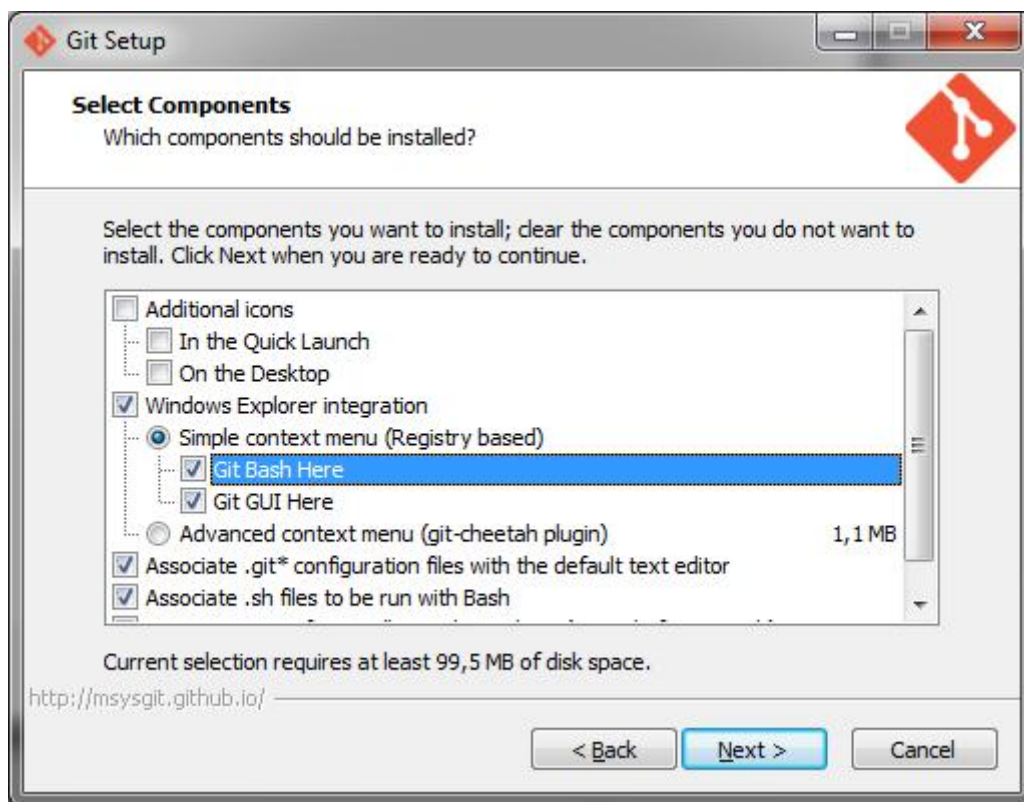
Naredni prozor prikazuje podatke o GNU licenci. Potrebno je kliknuti na dugme **Next**.



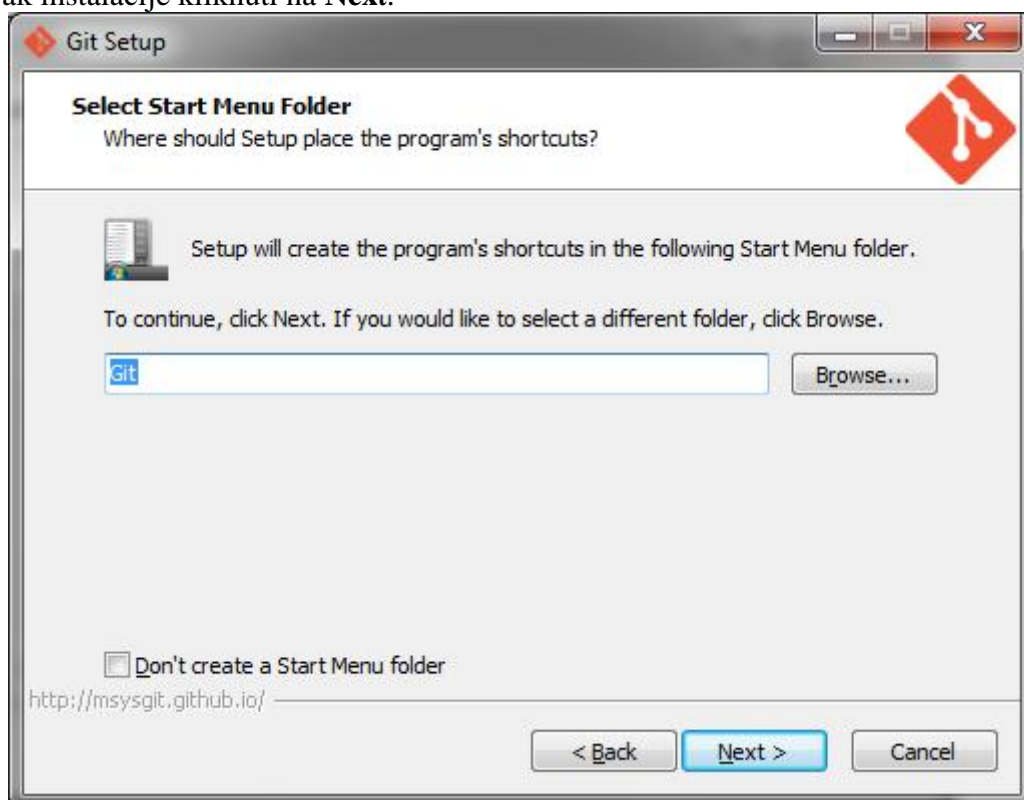
Naredni prozor traži od korisnika da odabere u kom direktorijumu će biti instaliran GitBash. Moguće je ukucati putanju ili pomoću dugmeta **Browse** pronaći odgovarajući direktorijum. Nakon odabira direktorijuma treba kliknuti na dugme **Next**.



Naredni prozor zahteva od korisnika da odabere komponente koje želi da budu instalirane. Da bi se koristio GitBash treba ekirati ku icu GitBash Here, a ukoliko je korisniku potrebna GUI (Graphical user interface) verzija potrebno je da ekira i GUI Here. Nakon odabranih komponenti kliknuti na dugme **Next**.



Prozor koji se pojavljuje e tražiti od korisnika da definiše pre ica u Start meniju. ekiranjem ku ice Don't create a Start Menu folder mogu e je odbiti ponudu Git Setup-a da se kreira pre ica. Za nastavak instalacije kliknuti na **Next**.

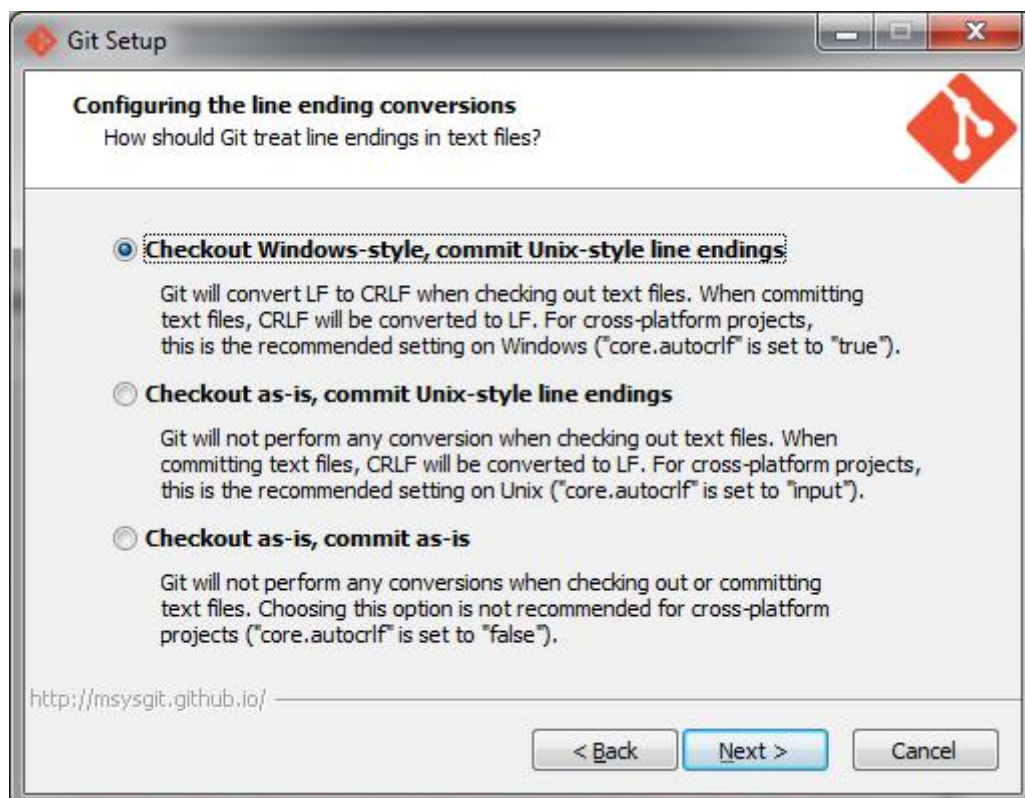


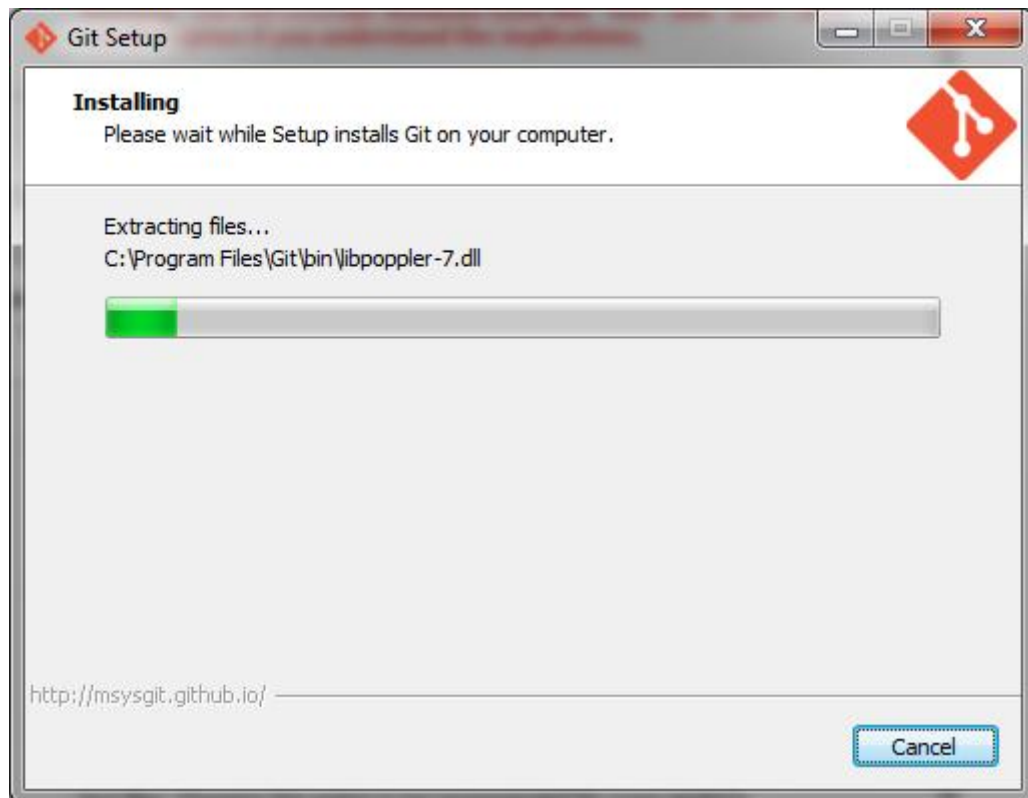


Naredni prozor pita korisnika kako će koristiti Git. Moгуće je koristiti git samo iz GitBash programa, iz Windows Command Prompt-a ili Obe verzije. Pri instalaciji GitBasha za svrhu ovog seminarskog rada odabrana je prva opcija. Za nastavak kliknuti **Next**.



Naredni prozor zahteva od korisnika da odabere kako će Git tretirati kraj linije u tekstualnim fajlovima. Za potrebe ovog seminarskog rada odabrana je prva opcija. Kliknuti na dugme **Next**.



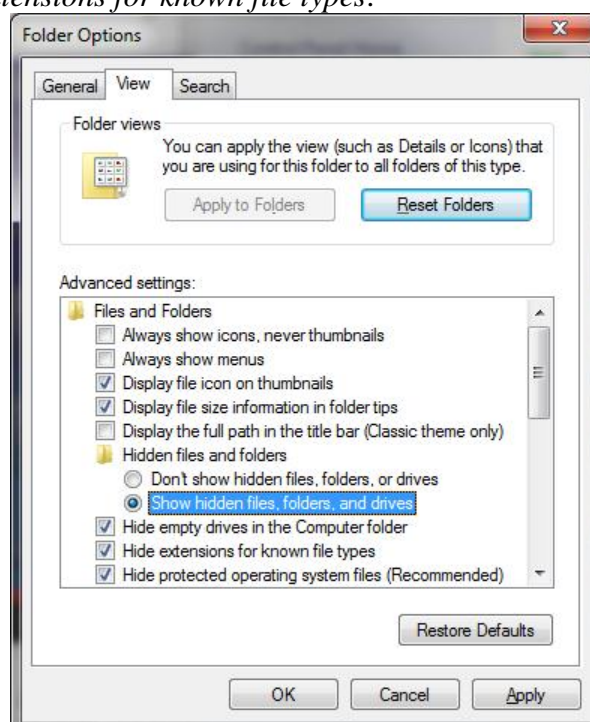


Nedugo zatim kopira se fajlovi u odabrani direktorijum i instalacija je bita gotova. Instalacija GIT klijenta je neophodna jer se ina posla obavlja lokalno na računaru.

## 2.1 Konfiguracija GitBash-a

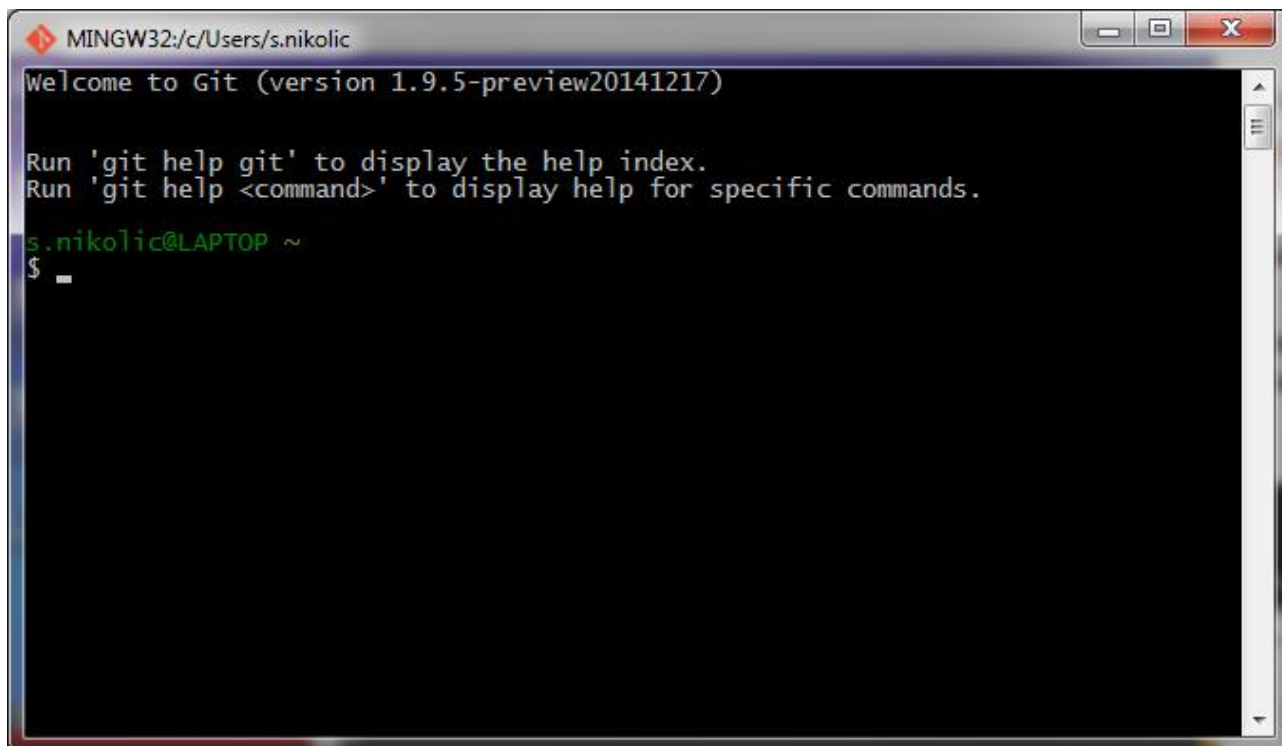
Da bi korisnik mogao da vidi skrivene direktorijume i fajlove koje kreira GitBash potrebno je podesiti windows da ih prikazuje. Ovi direktorijumi kasnije bitu potrebni pri konfiguraciji GitBash-a. Da bi se omogućio prikaz ovih direktorijuma potrebno je kliknuti na: Control panel → appearance and personalisation → folder options → view tab

- Odabrati *Show hidden files, folders, and drives*
- Odškrinuti *Hide extensions for known file types*.





Pri aktiviranju GitBasha prikaza e se slede i prozor:

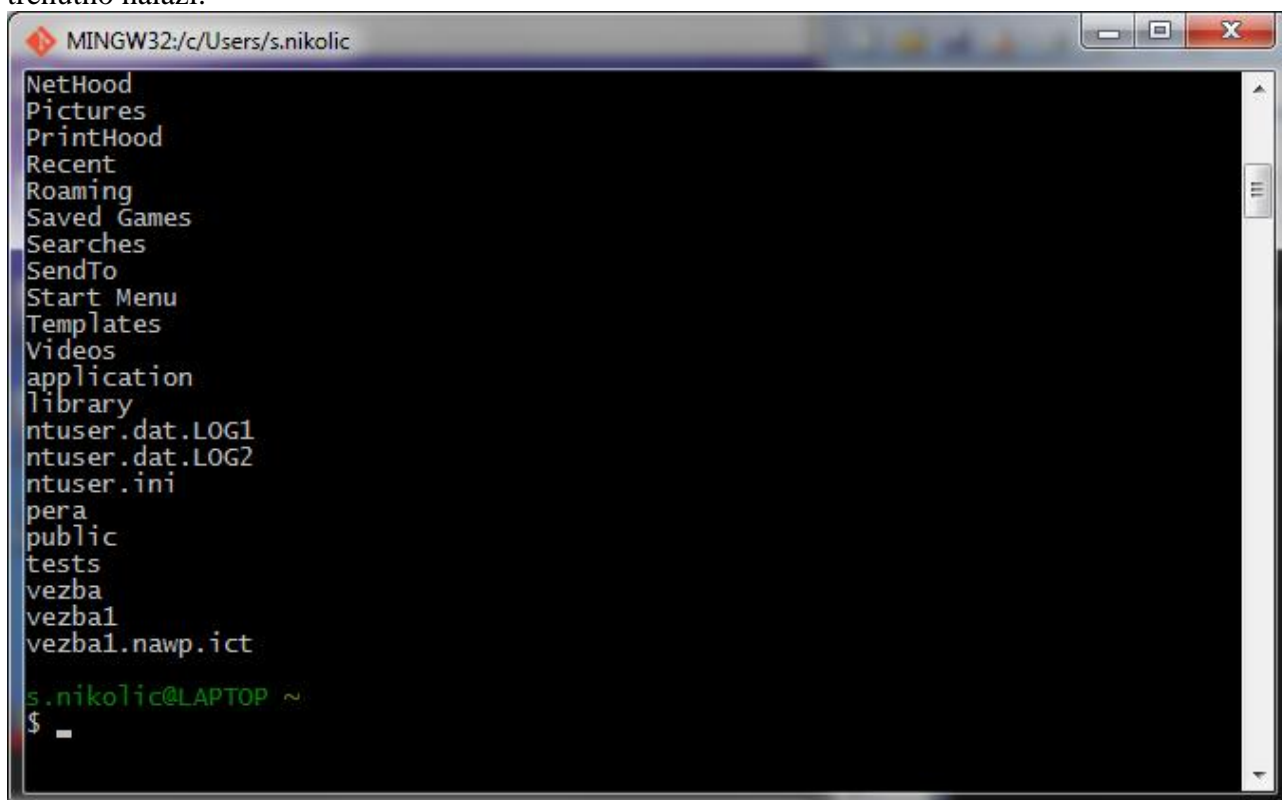


```
MINGW32:/c/Users/s.nikolic
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP ~
$ _
```

Mogu e je pokrenuti komandu **LS** koja služi za listanje sadržaja direktorijuma u kome se korisnik trenutno nalazi.

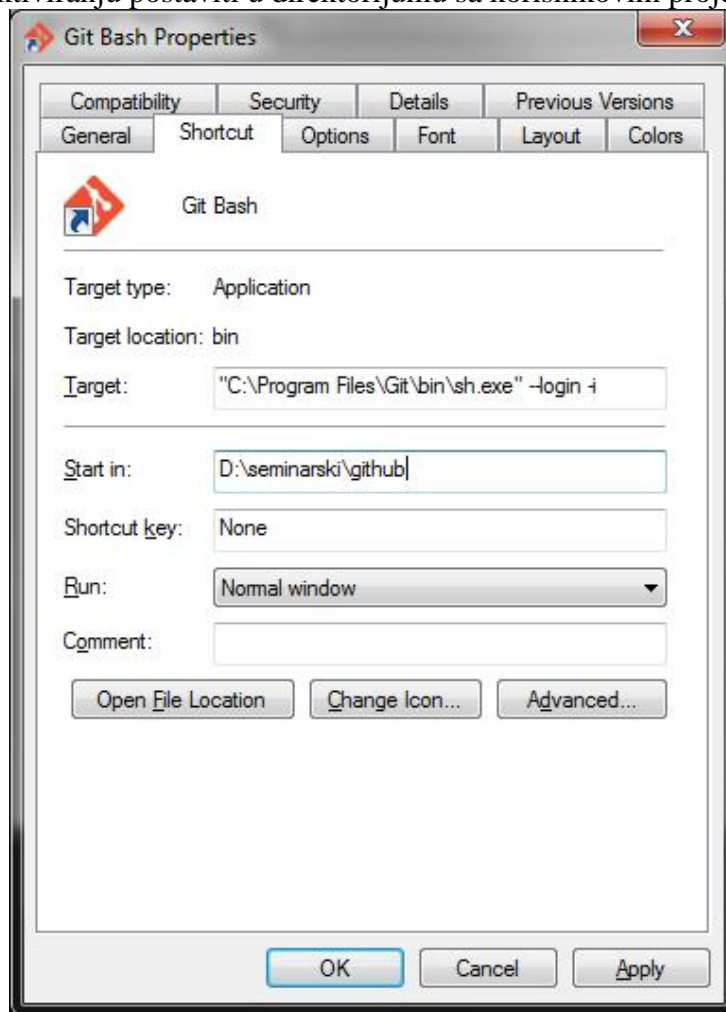


```
MINGW32:/c/Users/s.nikolic
NetHood
Pictures
PrintHood
Recent
Roaming
Saved Games
Searches
SendTo
Start Menu
Templates
Videos
application
library
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
pera
public
tests
vezba
vezba1
vezba1.nawp.ict

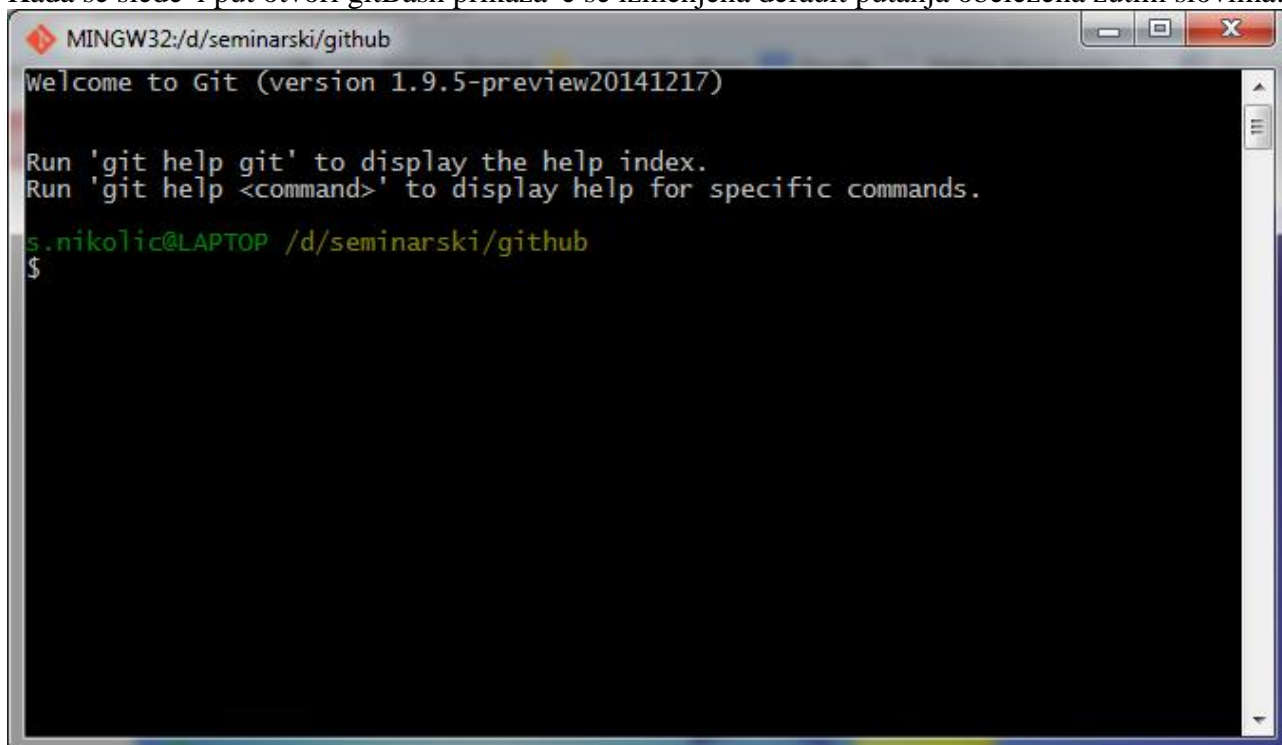
s.nikolic@LAPTOP ~
$ _
```

Default putanja je direktorijum korisnika u Windowsu (u ovom primeru je c:/users/s.nikolic).  
Ve ina ljudi ne uva svoje projekte u ovom direktorijumu ve ima specifi no mesto za to.  
Da bi se ovo podešavanje promenilo treba uraditi **desni klik na ikonicu GitBasha** pa **properties**.

Pod stavkom Start in: uneti adresu direktorijuma u kome se nalaze vaši projekti. Na ovaj način se GitBash odmah pri aktiviranju postaviti u direktorijumu sa korisnikovim projektima.



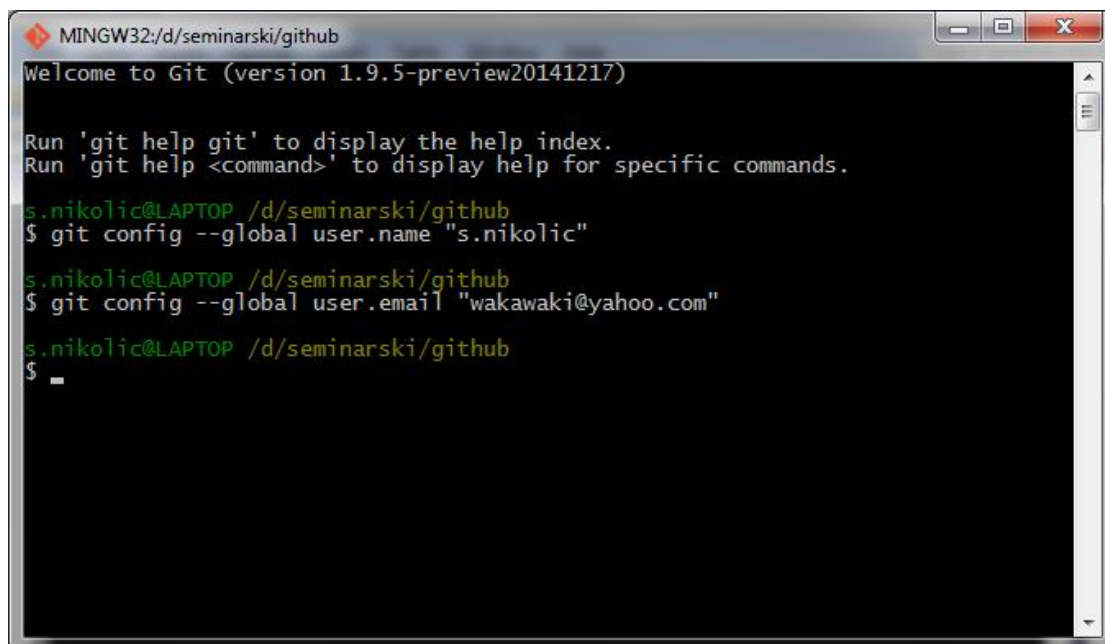
Kada se sledeći put otvori gitBash prikazaće se izmenjena default putanja obeležena žutim slovima:



Da bi se zabeležili podaci o lokalnom korisniku potrebno je uneti naredne dve komande:

Git config --global user.name "<vaše ime>"

Git config --global user.email "<vaš email>"



```
MINGW32:/d/seminarski/github
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.name "s.nikolic"

s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.email "wakawaki@yahoo.com"

s.nikolic@LAPTOP /d/seminarski/github
$ _
```

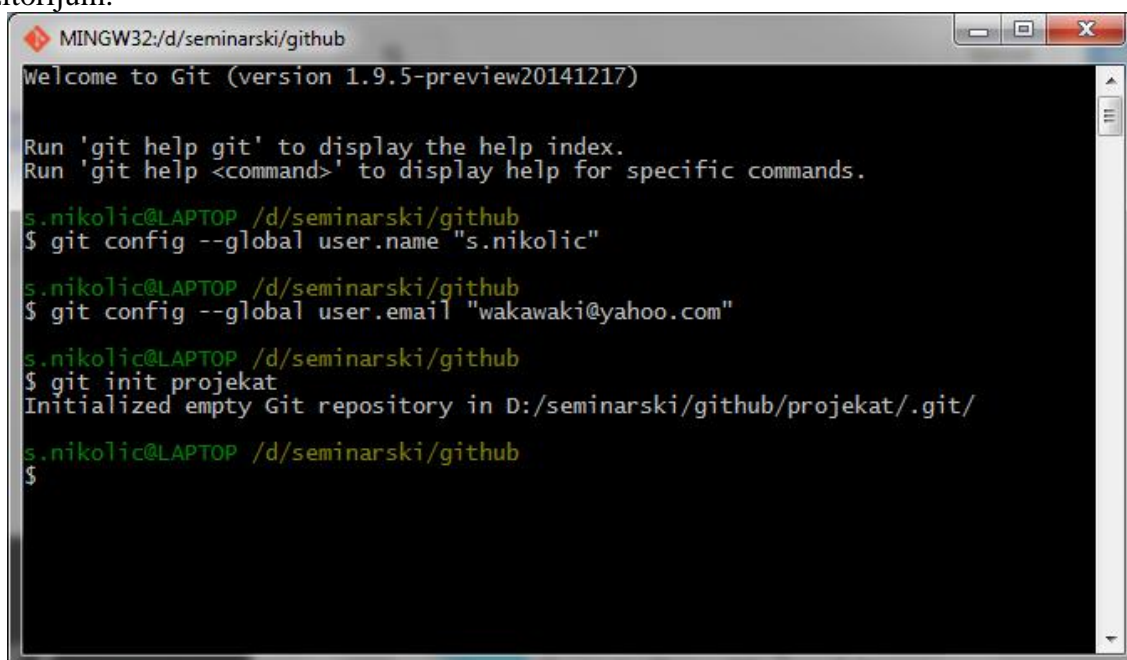
Ovo su podaci koji će se pojavljivati pri svakom commitovanju podataka. Ovi podaci opisuju ko je aktivirao commiti odnose se na korisničko ime i email korisnika.

Sada je potrebno kreirati direktorijum unutar direktorijuma sa projektima. To će biti direktorijum u kome će se konkretno nalaziti projekat sa kojim se radi. U tom direktorijumu bi trebalo postaviti fajlove od projekta i jedan readme.txt fajl.

GitBashu je potrebno reći da neki direktorijum predstavlja mesto na kome se nalazi projekat i je se verzije kontrolisati i pratiti. Zbog toga se mora inicijalizovati direktorijum projekata i to se radi komandom:

Git init <naziv foldera projekta>

Na ovaj način se podešava da je folder git repozitorijum, tj. inicijalizirati prazan lokalni git repozitorijum.



```
MINGW32:/d/seminarski/github
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

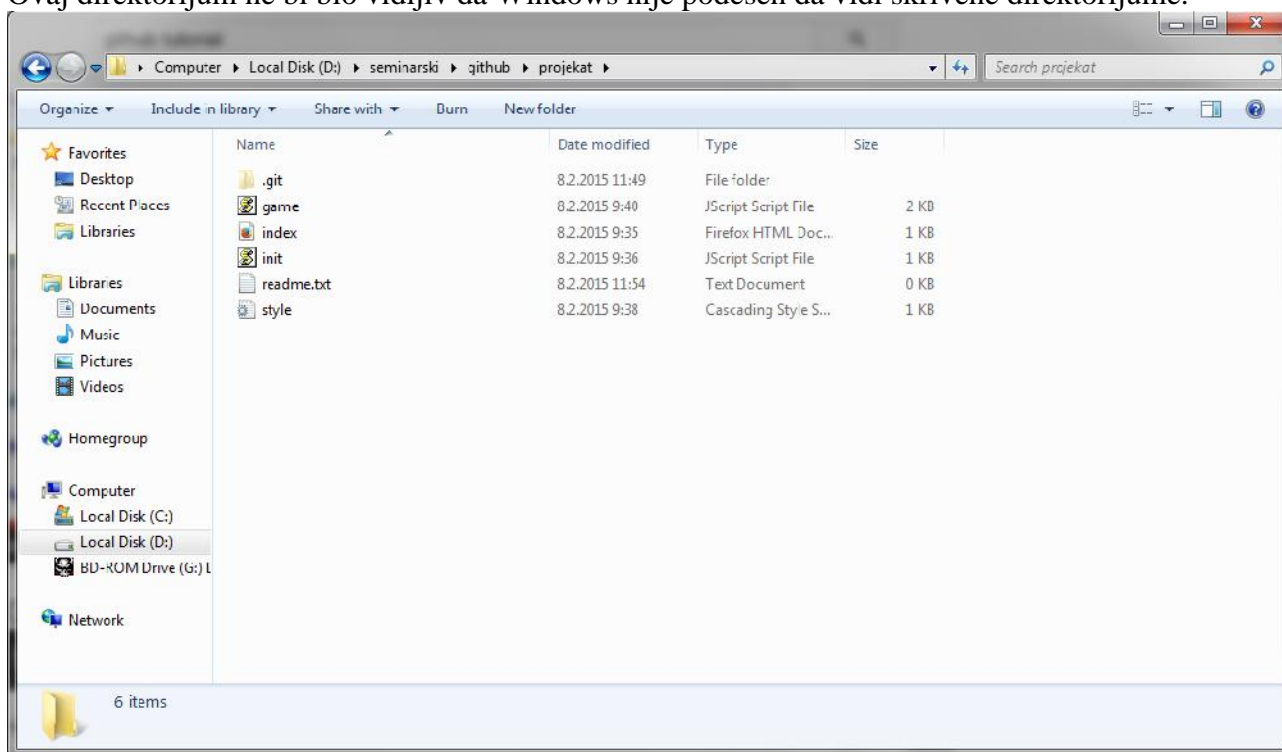
s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.name "s.nikolic"

s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.email "wakawaki@yahoo.com"

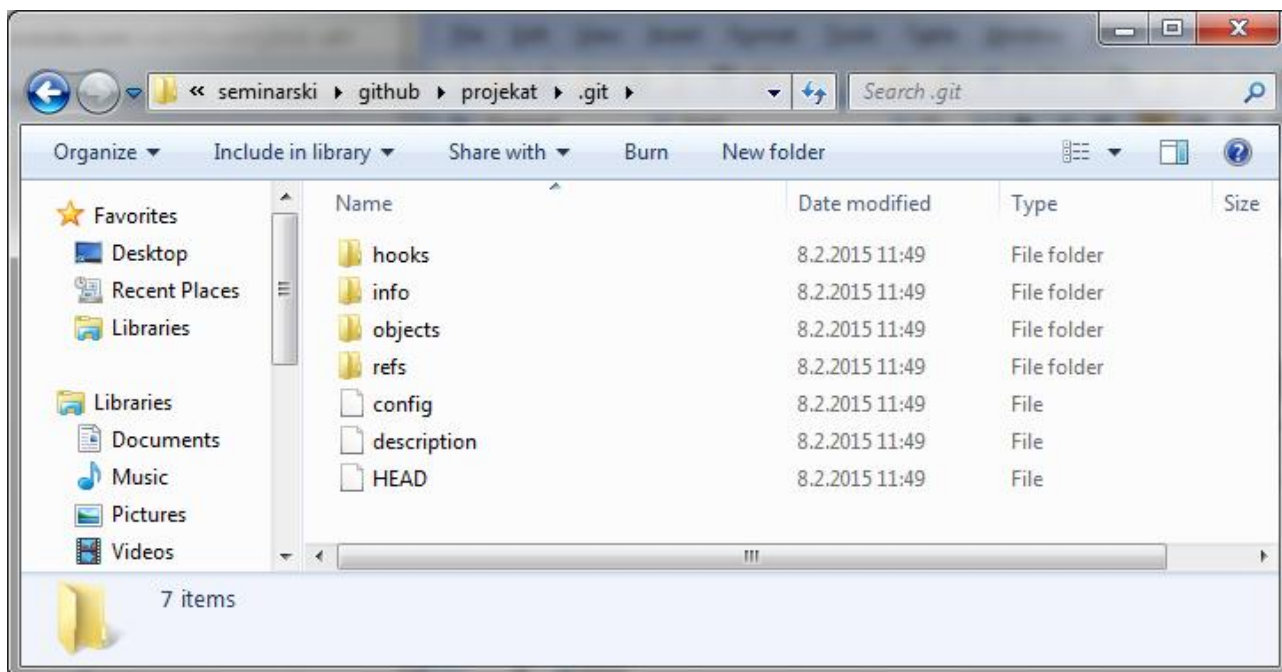
s.nikolic@LAPTOP /d/seminarski/github
$ git init projekat
Initialized empty Git repository in D:/seminarski/github/projekat/.git/

s.nikolic@LAPTOP /d/seminarski/github
$
```

Nakon inicijalizacije može se primetiti da se u direktorijumu projekta pojavio novi direktorijum .git. Ovaj direktorijum ne bi bio vidljiv da Windows nije podešen da vidi skrivene direktorijume.



Ovaj direktorijum sadrži fajlove i reference koje git koristi da kontrolise verzije projekta. Sadržaj direktorijuma se može videti na sledećoj slici:



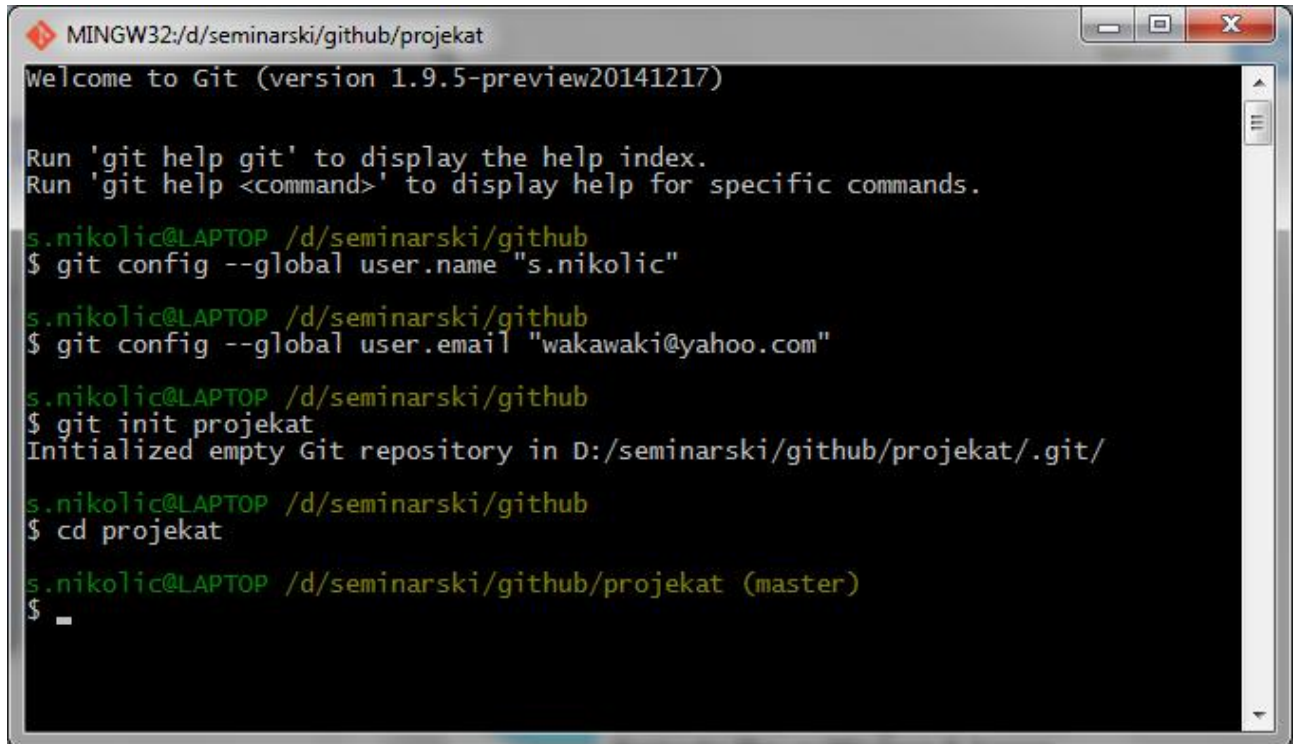


## 2.2 Rad sa gitBash-om.

Za promenu direktorijuma koristi se klasi na komanda iz DOS-a: `CD <NAZIV PROJEKTA>`.

`CD` je skra enica od **Change directory**. U ovom primeru svoj projekat sam nazvao PROJEKAT, te e komanda za ulazak u direktorijum sa projektom glasiti:

CD PROJEKAT



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.name "s.nikolic"

s.nikolic@LAPTOP /d/seminarski/github
$ git config --global user.email "wakawaki@yahoo.com"

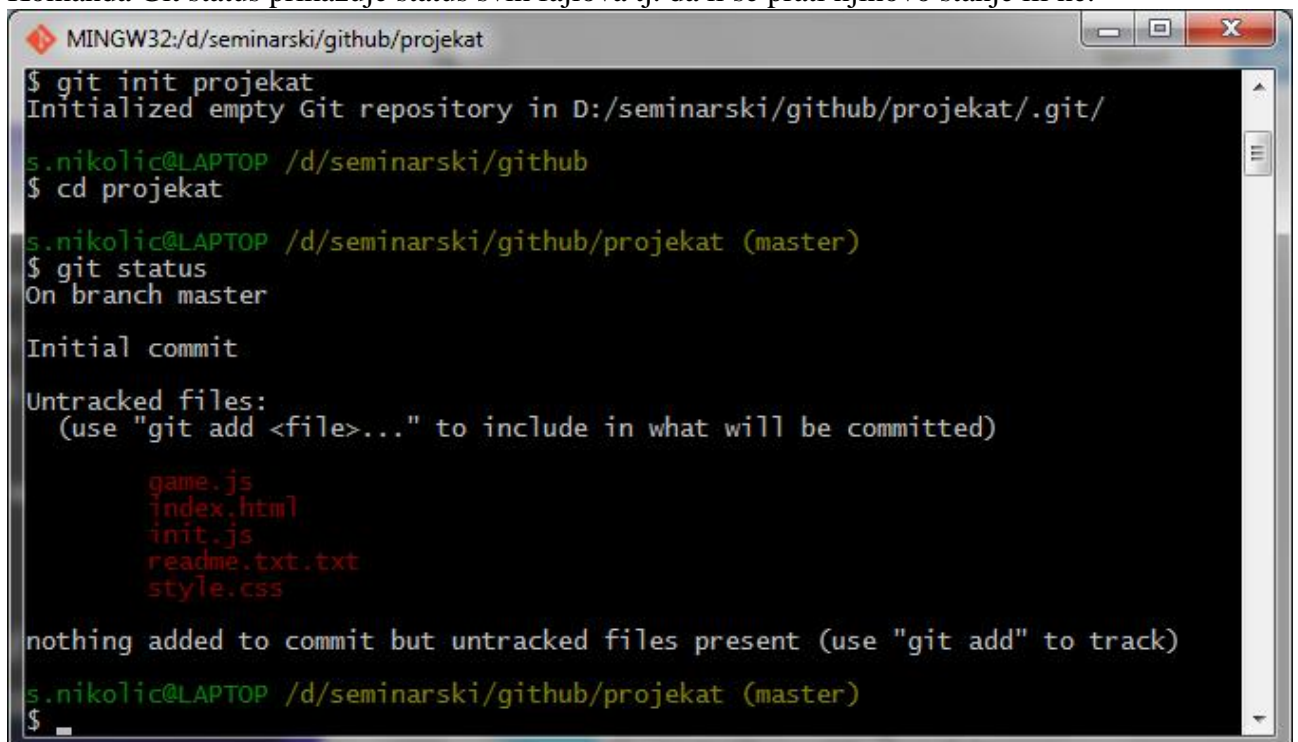
s.nikolic@LAPTOP /d/seminarski/github
$ git init projekat
Initialized empty Git repository in D:/seminarski/github/projekat/.git/

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ _
```

Može se primetiti da se iza putanje direktorijuma nalazi oznaka (**master**). Kada se inicijalizuje git projekat taj direktorijum dobija oznaku *master*. Ona ozna čava da je ovaj direktorijum default grana projekta.

Komanda `Git status` prikazuje status svih fajlova tj. da li se prati njihovo stanje ili ne.



```
MINGW32:/d/seminarski/github/projekat
$ git init projekat
Initialized empty Git repository in D:/seminarski/github/projekat/.git/

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    game.js
    index.html
    init.js
    readme.txt.txt
    style.css

nothing added to commit but untracked files present (use "git add" to track)

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ _
```

Na po etku e se prikazati da su svi fajlovi crvene boje (untracked - ne prate se njihove promene).



**Git status** komanda daje sledeće podatke:

- Na kojoj grani se korisnik nalazi
- Koji fajlovi su promenjeni
- Koji fajlovi se ne prate
- Šta činiti dalje.

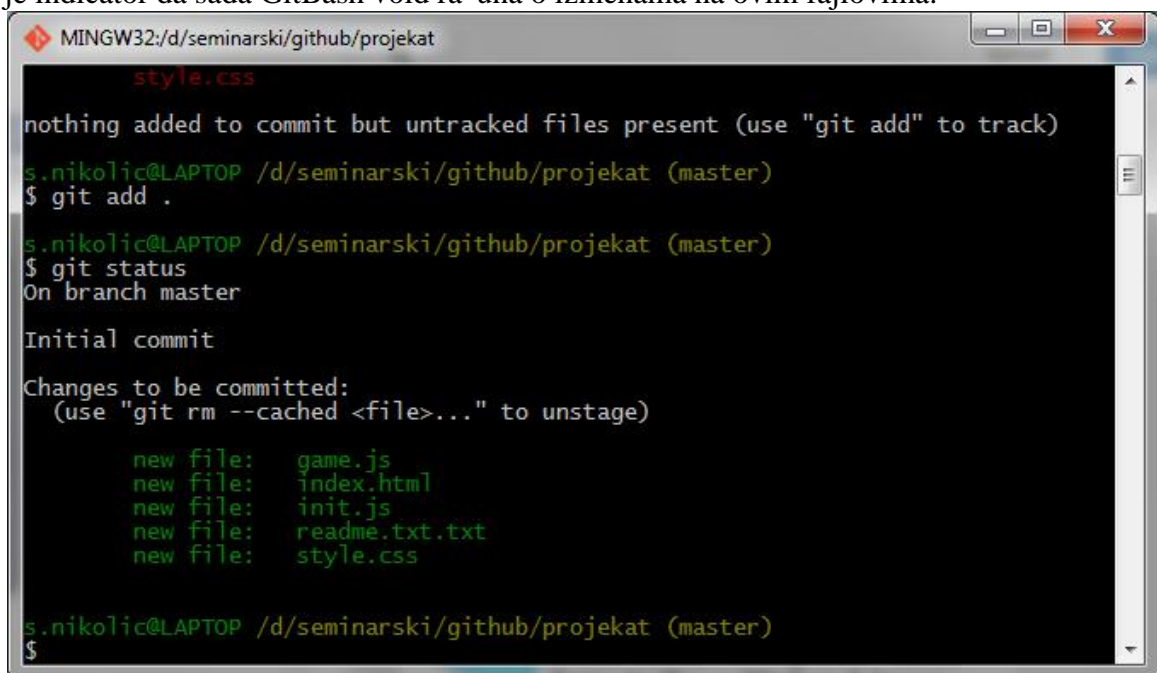
Da bi GitBash pratio promene na fajlovima potrebno je dodati ga u listu fajlova za praćenje statusa. To se postiže komandom:

**GIT ADD .**

Ova komanda će dodati sve fajlove u direktorijumu na listu praćenja.

Alternativa je **GIT ADD <ime fajla>** gde će se podesiti praćenje određenog fajla.

Kada se ponovo pozove **GIT STATUS** može se primetiti da su svi fajlovi u listi zelene boje. Zelena boja je indikator da sada GitBash vodi računa o izmenama na ovim fajlovima.



```
MINGW32:/d/seminarski/github/projekat
style.css
nothing added to commit but untracked files present (use "git add" to track)
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git add .
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git status
On branch master

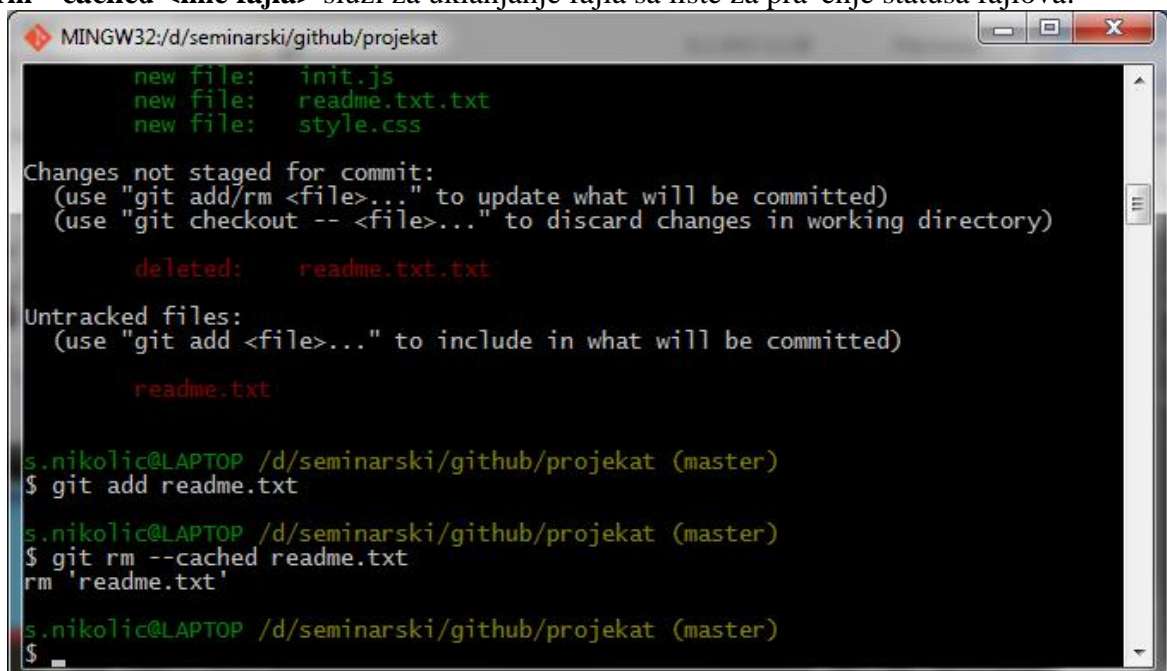
Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   game.js
        new file:   index.html
        new file:   init.js
        new file:   readme.txt.txt
        new file:   style.css

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

**Git rm --cached <ime fajla>** služi za uklanjanje fajla sa liste za praćenje statusa fajlova.



```
MINGW32:/d/seminarski/github/projekat
        new file:   init.js
        new file:   readme.txt.txt
        new file:   style.css

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    readme.txt.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        readme.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git add readme.txt
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git rm --cached readme.txt
rm 'readme.txt'
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Kada bi se ponovo ukucala komanda **git status** videlo bi se da je readme.txt skinut sa liste pra enja (untracked).

```
MINGW32:/d/seminarski/github/projekat
Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   game.js
    new file:   index.html
    new file:   init.js
    new file:   readme.txt.txt
    new file:   style.css

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    deleted:    readme.txt.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    readme.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Sada je mogu e uraditi prvi commit. Kao primer uneta je slede a komanda:

**GIT COMMIT –M “PRVI COMMIT”**

```
MINGW32:/d/seminarski/github/projekat

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    deleted:    readme.txt.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    readme.txt

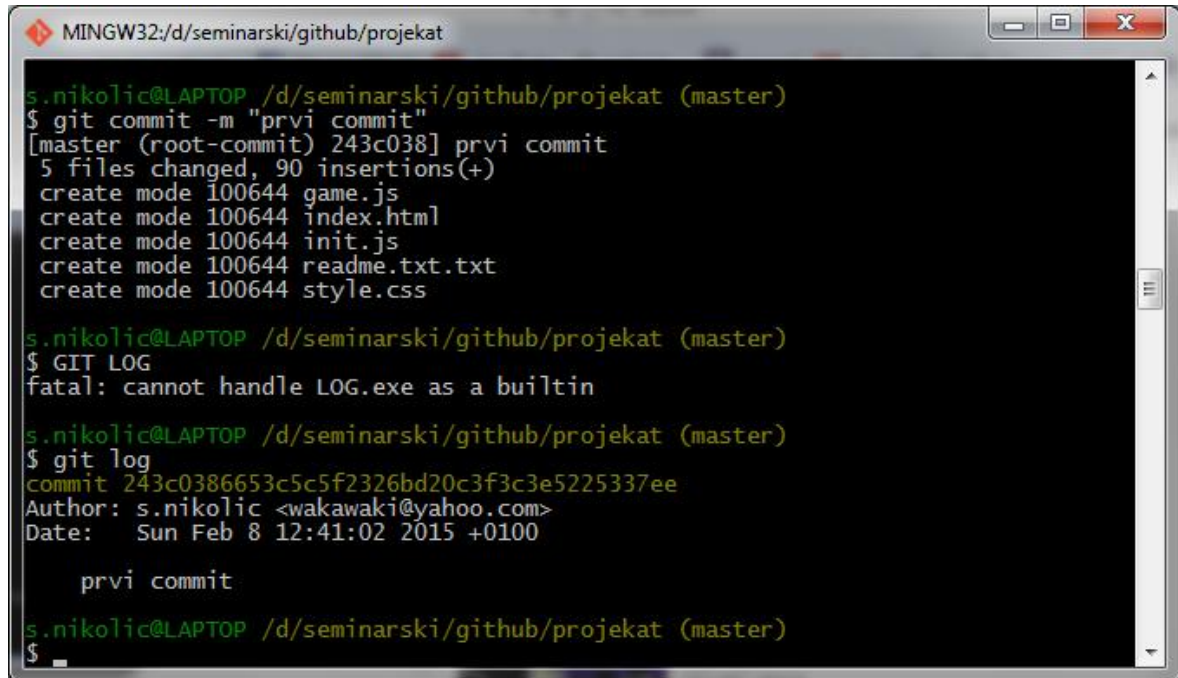
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -m "prvi commit"
[master (root-commit) 243c038] prvi commit
5 files changed, 90 insertions(+)
 create mode 100644 game.js
 create mode 100644 index.html
 create mode 100644 init.js
 create mode 100644 readme.txt.txt
 create mode 100644 style.css

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Parametar –m zna i da e uz commit biti prika ena poruka(message) iji e sadržaj biti naveden pod znacima navoda nakon parametra –m. Na ovaj na in se sadržaj liste za pra enje statusa fajlova šalje(commituje) lokalnom repozitorijumu, a opis ovog commita e biti ono što je dodato parametrom –m.

Komandom **GIT LOG** se dobija lista svih commitova. U ovom slučaju se vide sledeći podaci:

- Commit – id commita.
- Author: autor koji je poslao commit. Ovde se vide podaci o korisniku i email.
- Date: datum kada je commit poslat.
- Komentar koji prikazuje uz commit.



```
MINGW32:/d/seminarski/github/projekat
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -m "prvi commit"
[master (root-commit) 243c038] prvi commit
5 files changed, 90 insertions(+)
create mode 100644 game.js
create mode 100644 index.html
create mode 100644 init.js
create mode 100644 readme.txt.txt
create mode 100644 style.css

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ GIT LOG
fatal: cannot handle LOG.exe as a builtin

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git log
commit 243c0386653c5c5f2326bd20c3f3c3e5225337ee
Author: s.nikolic <wakawaki@yahoo.com>
Date: Sun Feb 8 12:41:02 2015 +0100

    prvi commit

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

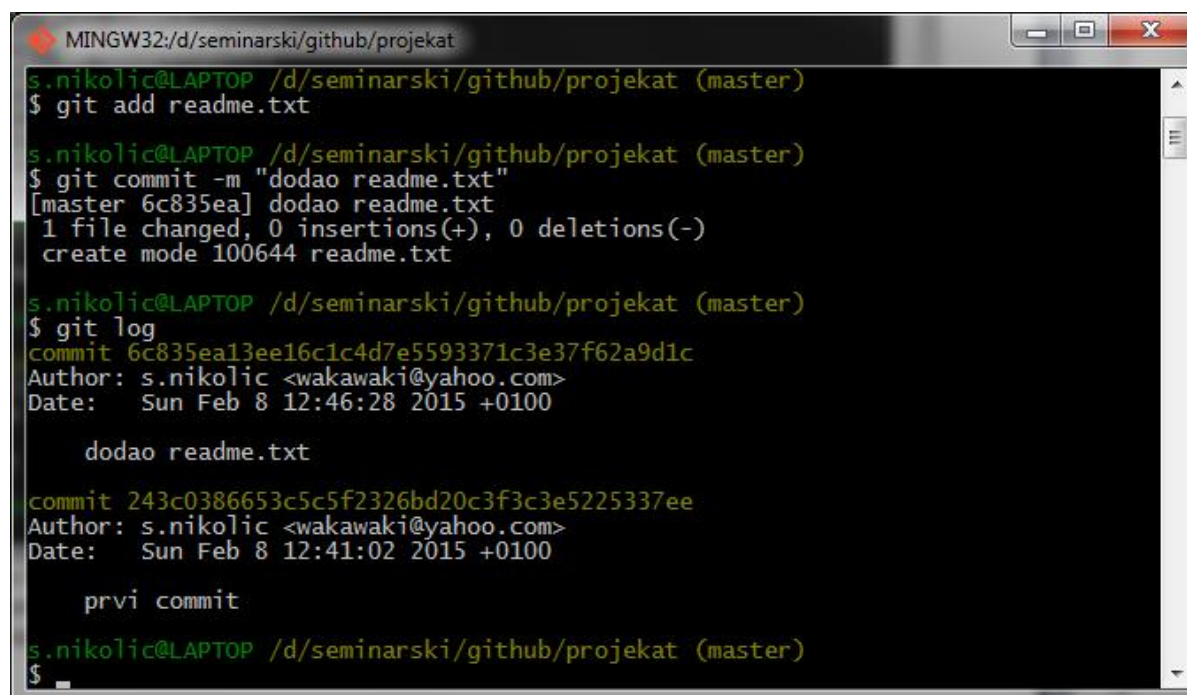
Da bi demonstrirao kako se kreira log doda u readme.txt listi pravi komentar:

**GIT ADD README.TXT**

i commitovati sve komandom:

**GIT COMMIT – M “DODAO README.TXT”**

Nakon toga komanda **GIT LOG** prikazuje sve promene.



```
MINGW32:/d/seminarski/github/projekat
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git add readme.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -m "dodao readme.txt"
[master 6c835ea] dodao readme.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 readme.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git log
commit 6c835ea13ee16c1c4d7e5593371c3e37f62a9d1c
Author: s.nikolic <wakawaki@yahoo.com>
Date: Sun Feb 8 12:46:28 2015 +0100

    dodao readme.txt

commit 243c0386653c5c5f2326bd20c3f3c3e5225337ee
Author: s.nikolic <wakawaki@yahoo.com>
Date: Sun Feb 8 12:41:02 2015 +0100

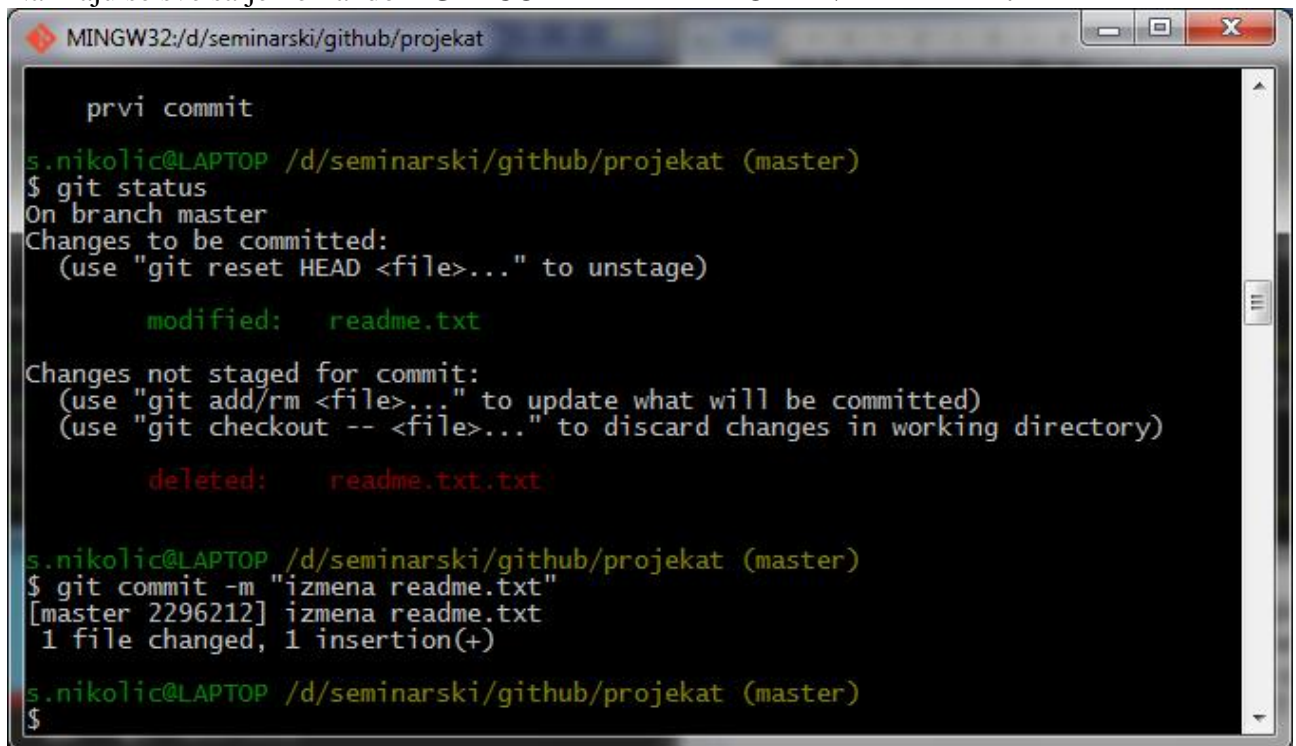
    prvi commit

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Savet: kada je log lista prevelika prikazuje se stranu po stranu. Da bi se izašlo iz ovog spiska treba sitisnuti **SHIFT+Z** dva puta.



U radu sa Git-om esto postoji više commitova i izmena. Kao primer sada je potrebno izmeniti sadržaj *readme.txt* fajla. Nakon toga se ponovo poziva komanda “**GIT ADD .**” da bi se svi fajlovi postavili na listu za pripremu za slanje. Ova procedura se ponavlja svaki put pre commitovanja. Na kraju se sve šalje komandom **GIT COMMIT -M “PROMENA README.TXT”**



```
MINGW32:/d/seminarski/github/projekat

prvi commit

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   readme.txt

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    readme.txt.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -m "izmena readme.txt"
[master 2296212] izmena readme.txt
1 file changed, 1 insertion(+)

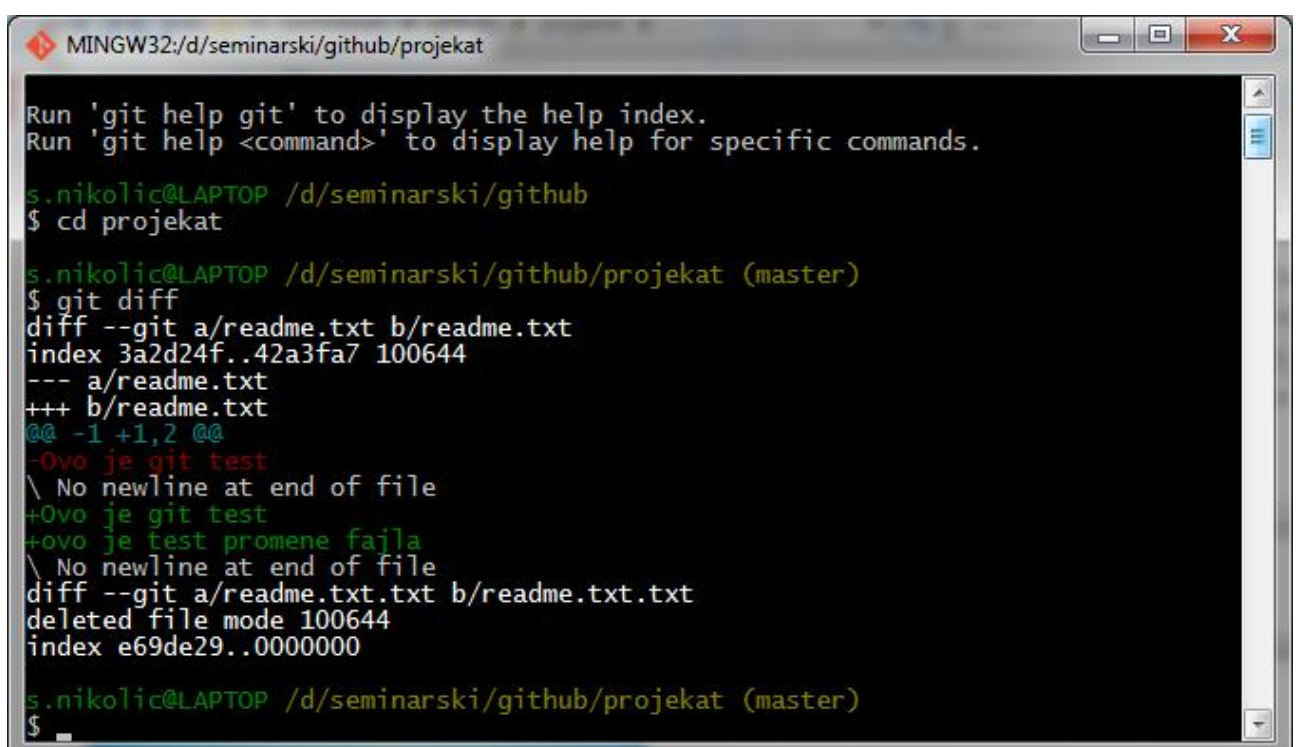
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Da bise demonstriralo kako se vide izmene, u *readme.txt* treba dodati redove, npr. :

*Ovo je git test*  
*ovo je test promene fajla*

Prethodna izmene su napravljena da bi se demonstrirao efekat komande:

Git diff



```
MINGW32:/d/seminarski/github/projekat

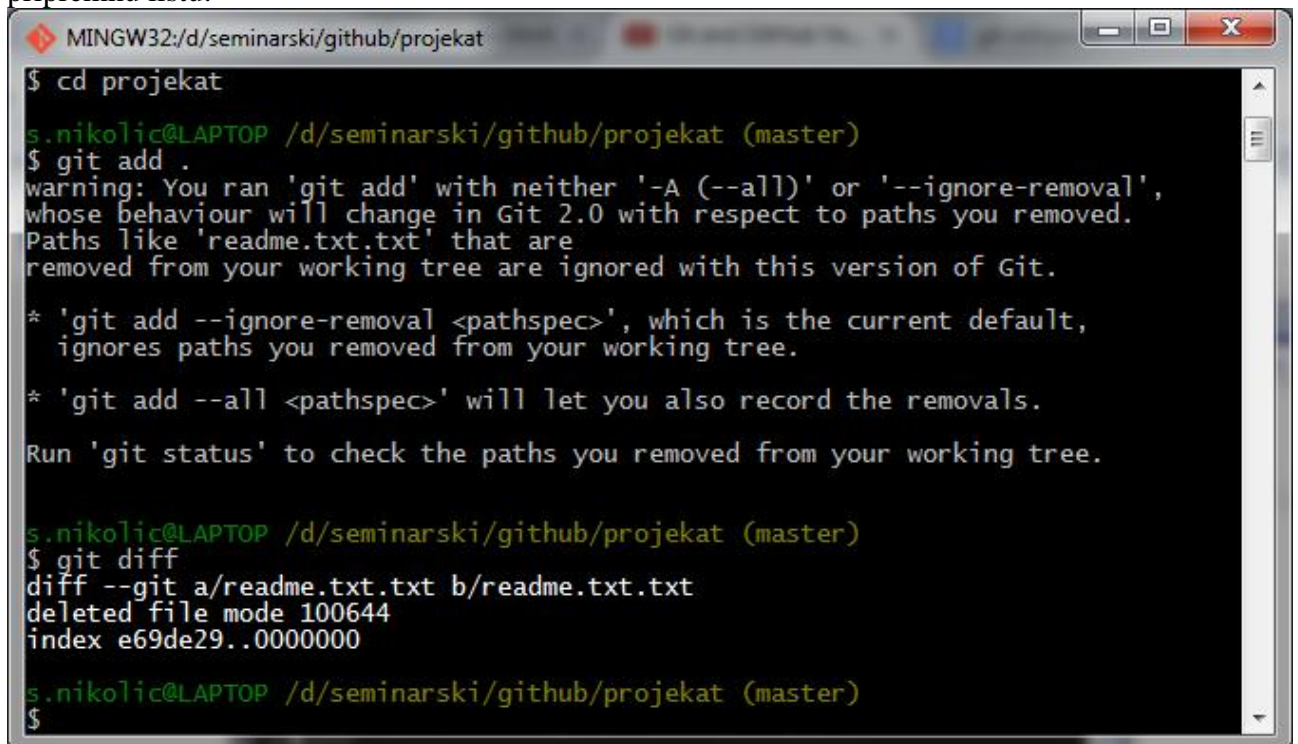
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git diff
diff --git a/readme.txt b/readme.txt
index 3a2d24f..42a3fa7 100644
--- a/readme.txt
+++ b/readme.txt
@@ -1,2 @@
-Ovo je git test
\ No newline at end of file
+Ovo je git test
+ovo je test promene fajla
\ No newline at end of file
diff --git a/readme.txt.txt b/readme.txt.txt
deleted file mode 100644
index e69de29..0000000

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Zelenom bojom je obeleženo da ova promena nije dodata na listu za pripremu za slanje. Kada se otkuca **GIT ADD** i ponovo se aktivira **GIT DIFF** može se videti da je sve dodato na pripremnu listu.



```
MINGW32:/d/seminarski/github/projekat
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git add .
warning: You ran 'git add' with neither '-A (--all)' or '--ignore-removal',
whose behaviour will change in Git 2.0 with respect to paths you removed.
Paths like 'readme.txt.txt' that are
removed from your working tree are ignored with this version of Git.

* 'git add --ignore-removal <paths>', which is the current default,
  ignores paths you removed from your working tree.

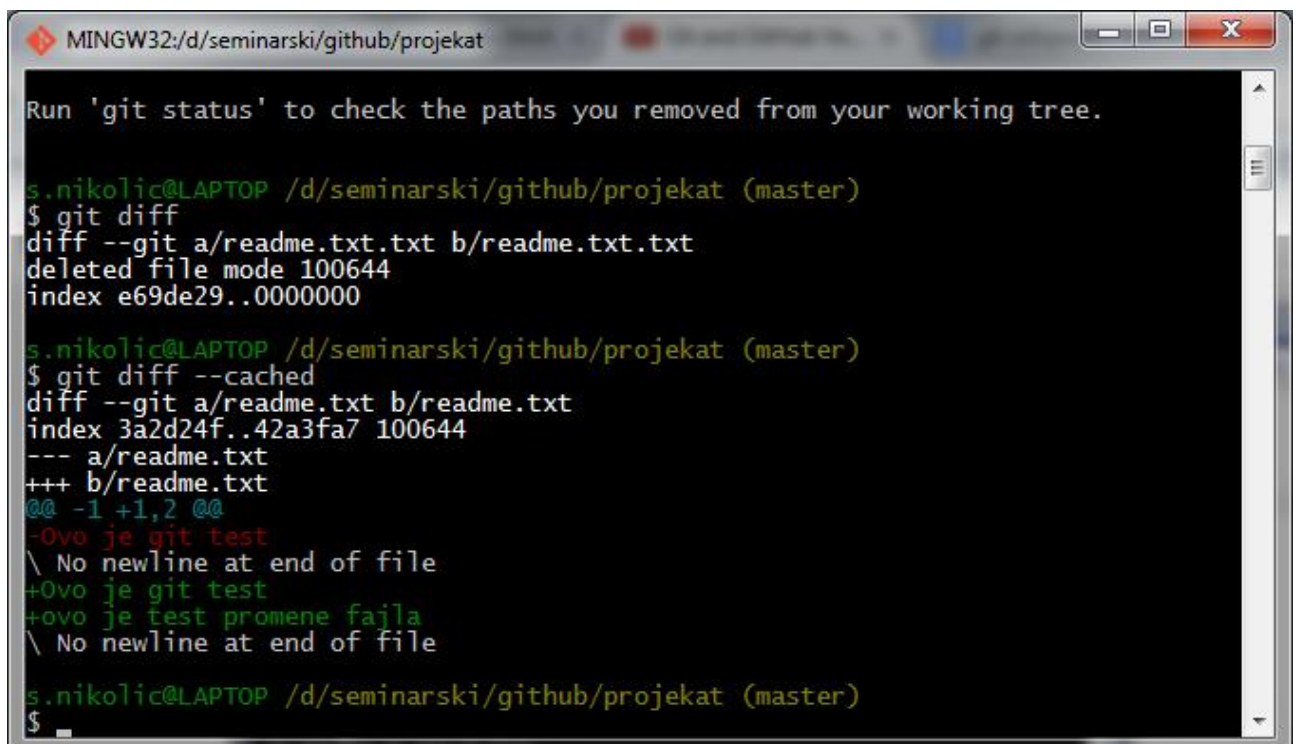
* 'git add --all <paths>' will let you also record the removals.

Run 'git status' to check the paths you removed from your working tree.

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git diff
diff --git a/readme.txt.txt b/readme.txt.txt
deleted file mode 100644
index e69de29..0000000

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Sada se ne mogu videti promene, jer još uvek nisu commitovane, ali ako se unese **GIT DIFF - CACHED** vide se razlike dok je fajl u pripremljenoj listi za commitovanje. Ovo je prikazano na sledećoj slici:



```
MINGW32:/d/seminarski/github/projekat

Run 'git status' to check the paths you removed from your working tree.

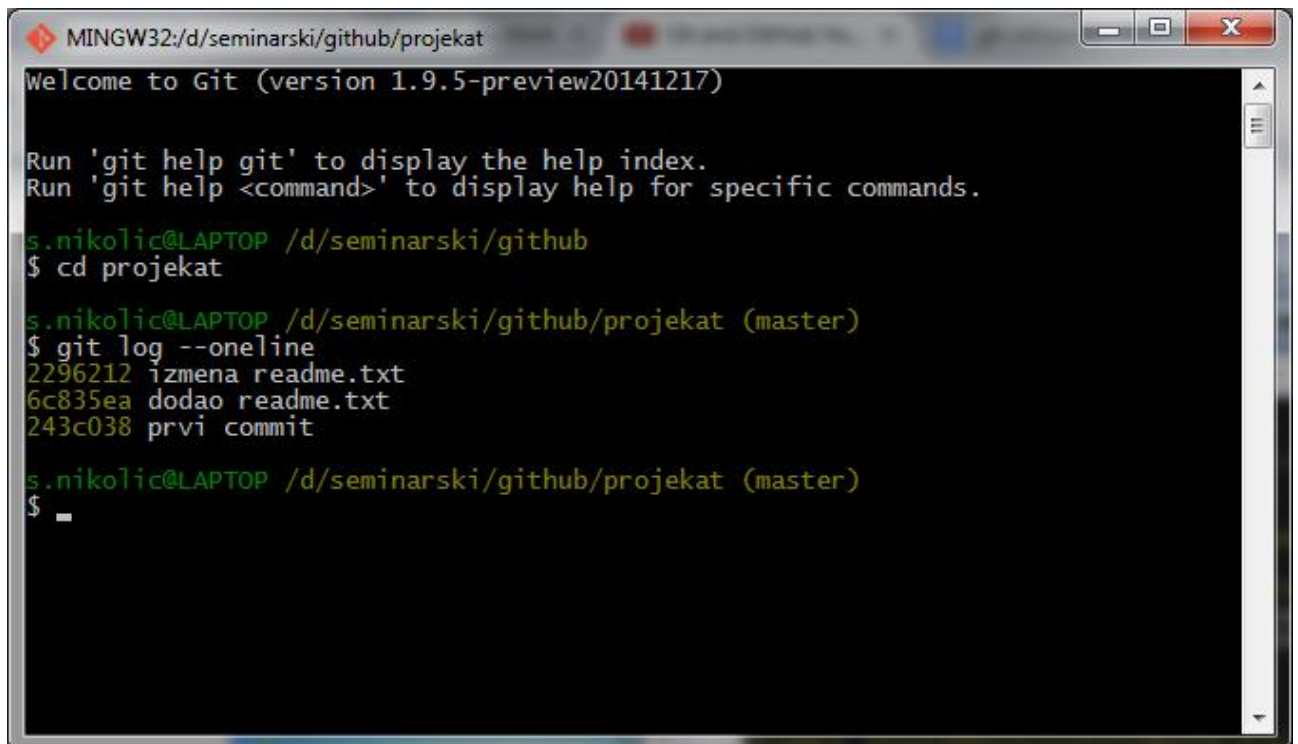
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git diff
diff --git a/readme.txt.txt b/readme.txt.txt
deleted file mode 100644
index e69de29..0000000

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git diff --cached
diff --git a/readme.txt b/readme.txt
index 3a2d24f..42a3fa7 100644
--- a/readme.txt
+++ b/readme.txt
@@ -1,2 @@
-Ovo je git test
\ No newline at end of file
+Ovo je git test
+Ovo je test promene fajla
\ No newline at end of file

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```



Komanda **GIT LOG - ONELINE** prikaza e samo commitove, njihove ID-ove i komentare.



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

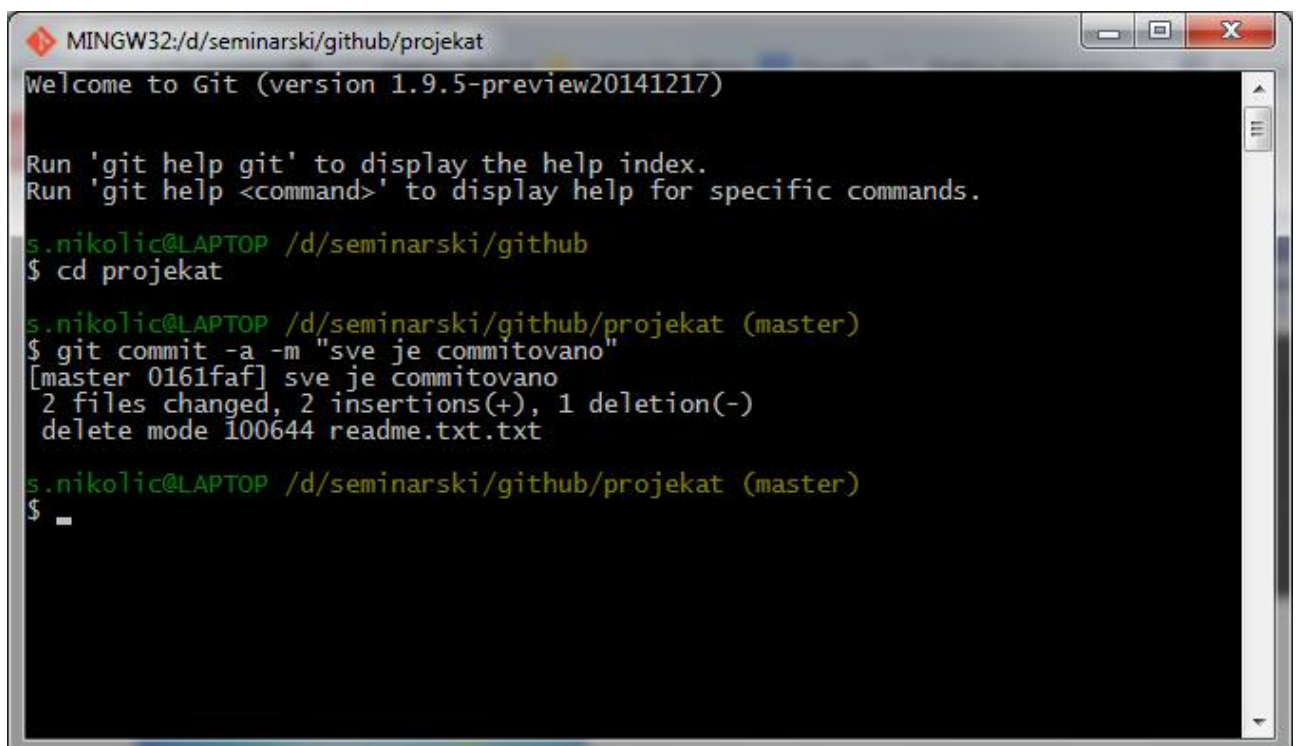
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git log --oneline
2296212 izmena readme.txt
6c835ea dodao readme.txt
243c038 prvi commit

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ _
```

Ovo je korisno ako je potreban samo kratak pregled commitova.

Da bi se commitovalo sve što je u folderu koristi se komanda commit sa dodatnim parametrima:

**GIT COMMIT -A -M "Sve je commitovano"**



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

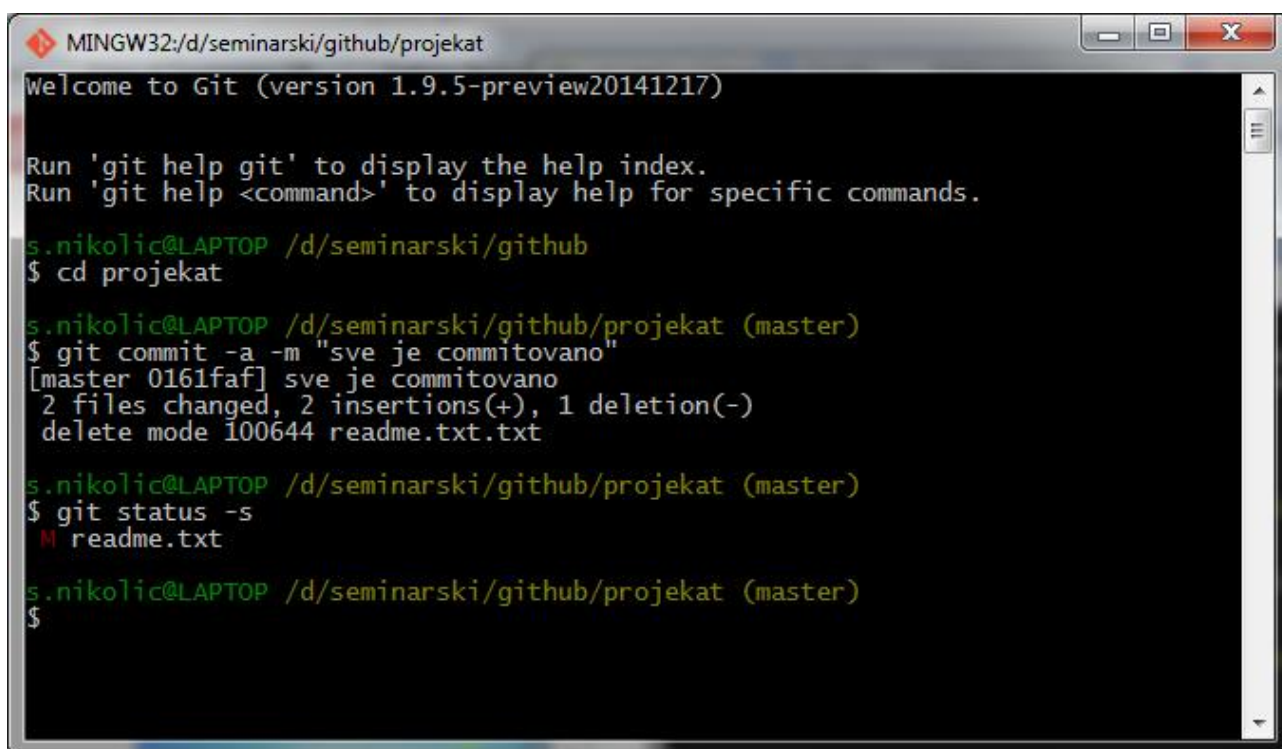
s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -a -m "sve je commitovano"
[master 0161faf] sve je commitovano
2 files changed, 2 insertions(+), 1 deletion(-)
delete mode 100644 readme.txt.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ _
```

Ukoliko je potrebno prikazati skraćenu listu statusa potrebno je komandi status dodati parameter **-s**:  
**GIT STATUS -S.**

Pretpostavimo da je *readme.txt* još jedanput izmenjen. Kada se unese ova komanda prikaza e se šta je još modifikovano i bi e obeleženo crvenim slovom M.



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git commit -a -m "sve je commitovano"
[master 0161faf] sve je commitovano
2 files changed, 2 insertions(+), 1 deletion(-)
delete mode 100644 readme.txt.txt

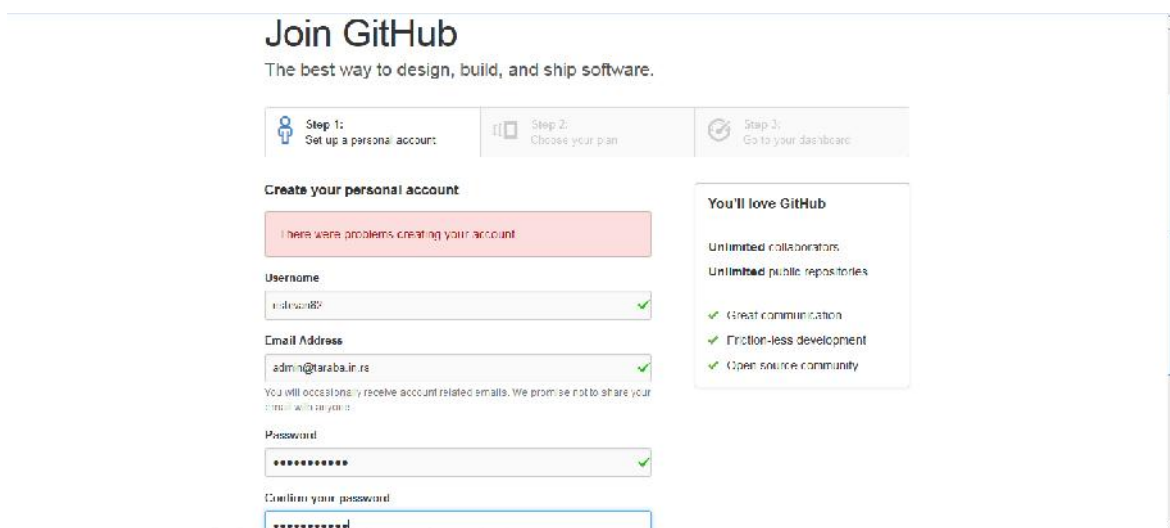
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git status -s
M readme.txt

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Git koristi keširanje da bi verifikovao da se repozitorijum koji se povezuje (lokalni ili udaljeni) imaju istu strukturu fajlova, tako da štiti od malicioznih radnji ili ošte enja fajlova. Git se ne koristi za bezbednost, ve zbog integriteta fajlova.

### 3.0 Registracija

Da bi koristio udaljeni repozitorijum (remote repository), svaki korisnik mora da otvori nalog na <https://github.com/>. Može se primetiti da Github koristi https zašti enu konekciju. Za otvaranje novog naloga kliknuti na *Sign up for Github*. Najpre e korisnik popuniti podatke o svom nalogu.



**Join GitHub**  
The best way to design, build, and ship software.

Step 1: Set up a personal account | Step 2: Choose your plan | Step 3: Go to your dashboard

**Create your personal account**

There were problems creating your account.

Username:  ✓

Email Address:  ✓  
You will occasionally receive account related emails. We promise not to share your email with anyone.

Password:  ✓

Confirm your password:

**You'll love GitHub**

- ✓ Unlimited collaborators
- ✓ Unlimited public repositories
- ✓ Great communication
- ✓ Friction-less development
- ✓ Open source community

Ukoliko korisnik ne želi da plati, odabra e open source nalog. On je otvoren za javnost. Ukoliko je potrebno da fajlovi ne budu vidljivi svima, tj. zašti eni, pla anje je obavezno.

# Welcome to GitHub

You've taken your first step into a larger world, @nstevan82.

 **Completed**  
Set up a personal account

 **Step 2:**  
Choose your plan

 **Step 3:**  
Go to your dashboard

## Choose your personal plan


Plan	Cost <a href="#">(view in RSD)</a>	Private repos	
Large	\$50/month	50	<a href="#">Choose</a>
Medium	\$22/month	20	<a href="#">Choose</a>
Small	\$12/month	10	<a href="#">Choose</a>
Micro	\$7/month	5	<a href="#">Choose</a>
Free	\$0/month	0	<a href="#">Chosen</a>

## Each plan includes:

**Unlimited** collaborators  
**Unlimited** public repositories

- ✓ Free setup
- ✓ SSL Protection
- ✓ Email support
- ✓ Wikis, Issues, Pages, & more

Kada se klikne na dugme *finish sign up* pojavi e se naredna strana:


 Search GitHub


Explore Gist Blog Help


nstevan82 + -


News Feed Pull Requests Issues


**GitHub Bootcamp** ×

**1**  
  
**Set up Git**  
A quick guide to help you get started with Git.


**2**  
  
**Create repositories**  
Repositories are where you'll work and collaborate on projects.


**3**  
  
**Fork repositories**  
Forking creates a new, unique project from an existing one.

**4**  
  
**Work together**  
Send pull requests, follow friends. Star and watch projects.

 **Welcome to GitHub! What's next?** (4 minutes ago)  
Create a repository  
Tell us about yourself  
Browse interesting repositories  
Follow @github on Twitter

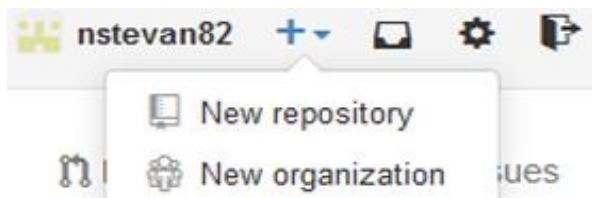
**Your repositories** 0 [+ New repository](#)  
You don't have any repositories yet!  
[Create your first repository](#) or [learn more about Git and GitHub](#).

 **ProTip!** Edit your feed by updating the users you [follow](#) and repositories you [watch](#).

 **ProTip!** You can download past receipts on the [receipts page](#).

U gornjem desnom uglu se mogu primetiti nekoliko ikonica:

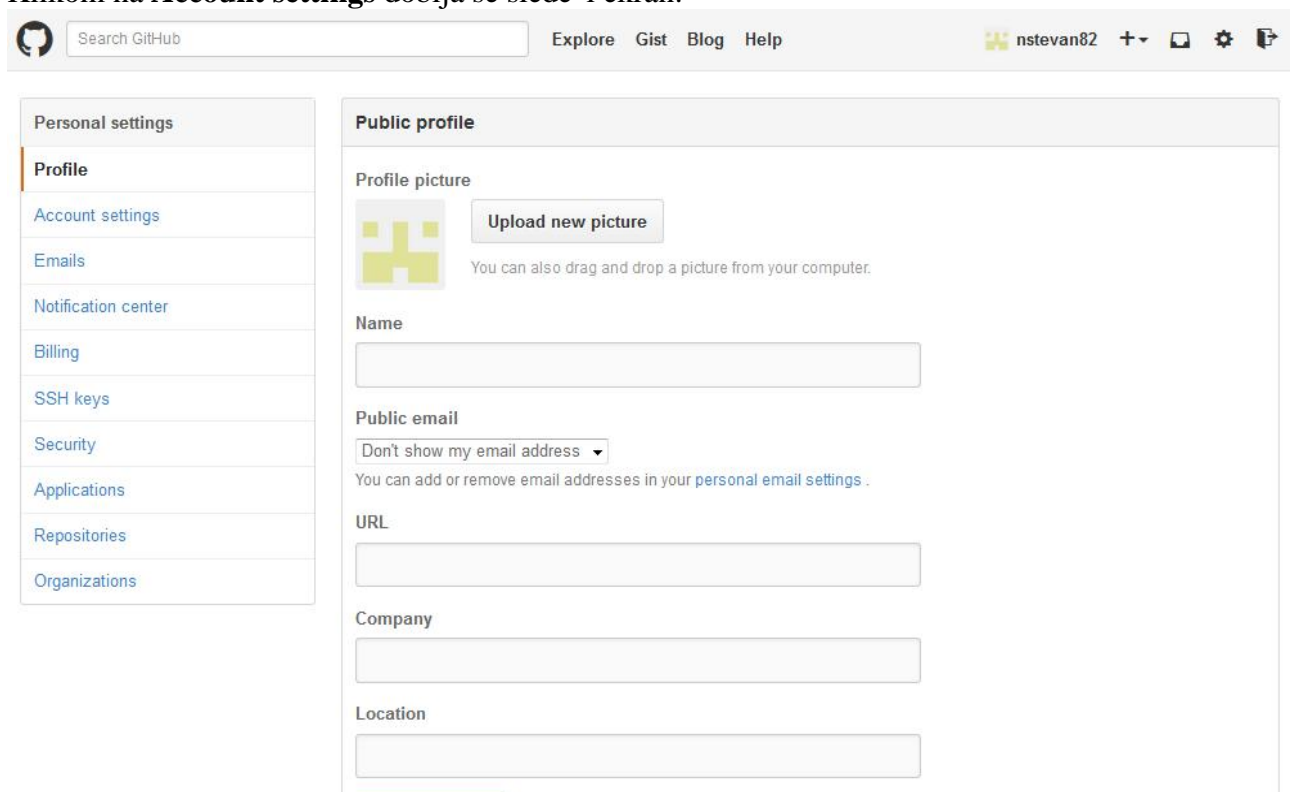
- Kreiranje novog repozitorijuma
- Nova organizacija
- Podešavanje naloga
- Log out



U donjem desnom uglu stranice se vidi koliko repozitorijuma postoji na nalogu, a klikom na dugme New repository kreira se novi repozitorijum.

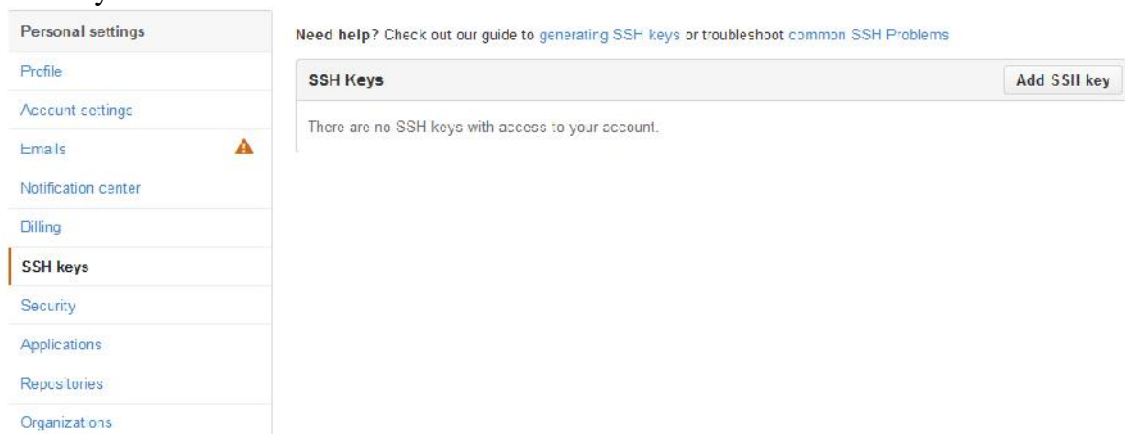


Klikom na **Account settings** dobija se sledeći ekran:



- Profile služi za podešavanja profila, dodavanje slike i nekih osnovnih podataka.
- Account settings – podešavanje korisničkog imena i lozinke.
- Emails – emailovi koji se koriste u radu sa git-om.
- Notification center – podešavanja vezana za obaveštenja u radu sa git-om.
- Billing – podaci vezani za naplatu korišćenja naloga.

## SSH keys:



SSH ključ omogućava upload fajlova na GitHub.

Da bi ovo bilo funkcionalno potreban je RSA ključ na lokalnom računaru. RSA je javni i privatni ključ. Privatni ključ mora biti na lokalnom računaru i dodati se javni ključ na GitHubu. Kada se ovo uradi GitHub će verovati korisnikovom računaru da ima prava da uploaduje kod.

Ako se koristi Gitbash, SSH ključ se mora podesiti ručno. Ako se koristi Windows Git Client, on će to uraditi automatski kada se korisnik uloguje prvi put.

Security – opcije i podaci vezani za bezbednost. Ovde se može pogledati i log vezan za ovaj nalog.

Applications – Ukoliko korisnik želi da razvija aplikaciju koristeći GitHub API-ja u ovoj opciji može naći korisne informacije i podešavanja.

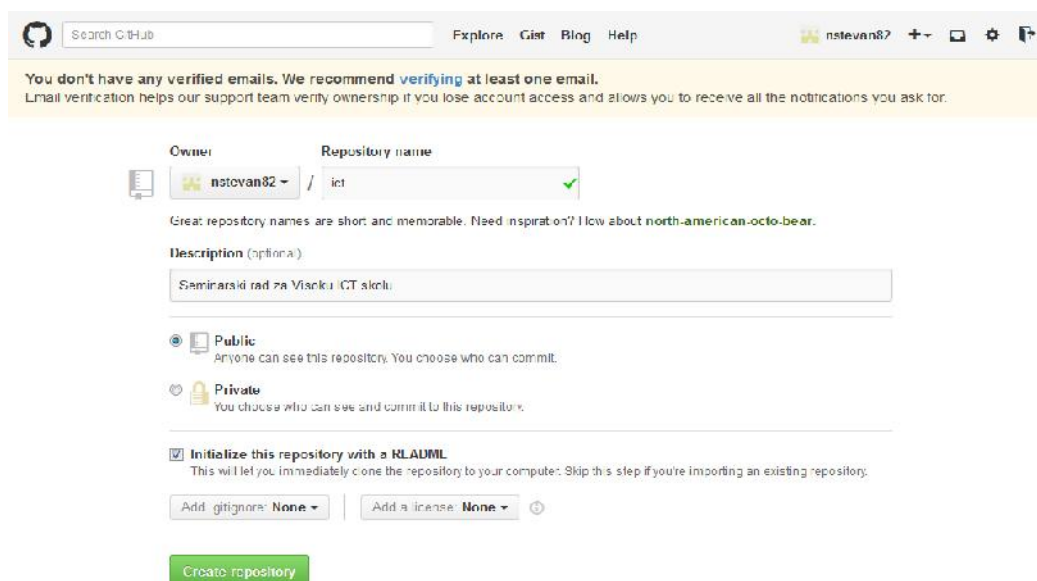
Repositories – lista remote repozitorijuma na ovom GitHub-u.

## Organisations:

Postoji mogućnost kreiranja naloga i dodati ga nekoj organizaciji ili (drugo dugme) pretvoriti nalog u organizaciju, tako da trenutni nalog bude biznis nalog. Ovako se dobija mnogo fleksibilnosti u radu. Ukoliko korisnik ima organizaciju verovatno će imati timove, a ovim timovima se mogu dodeliti dozvole da imaju mogućnost da vide određene projekte. Organizacije su potrebne samo ukoliko se radi na velikim projektima ili ako se radi na više projekata.

Ukoliko se korisnik odlučio da kreira novi repozitorij na GitHubu kliknuće na **New repository**.

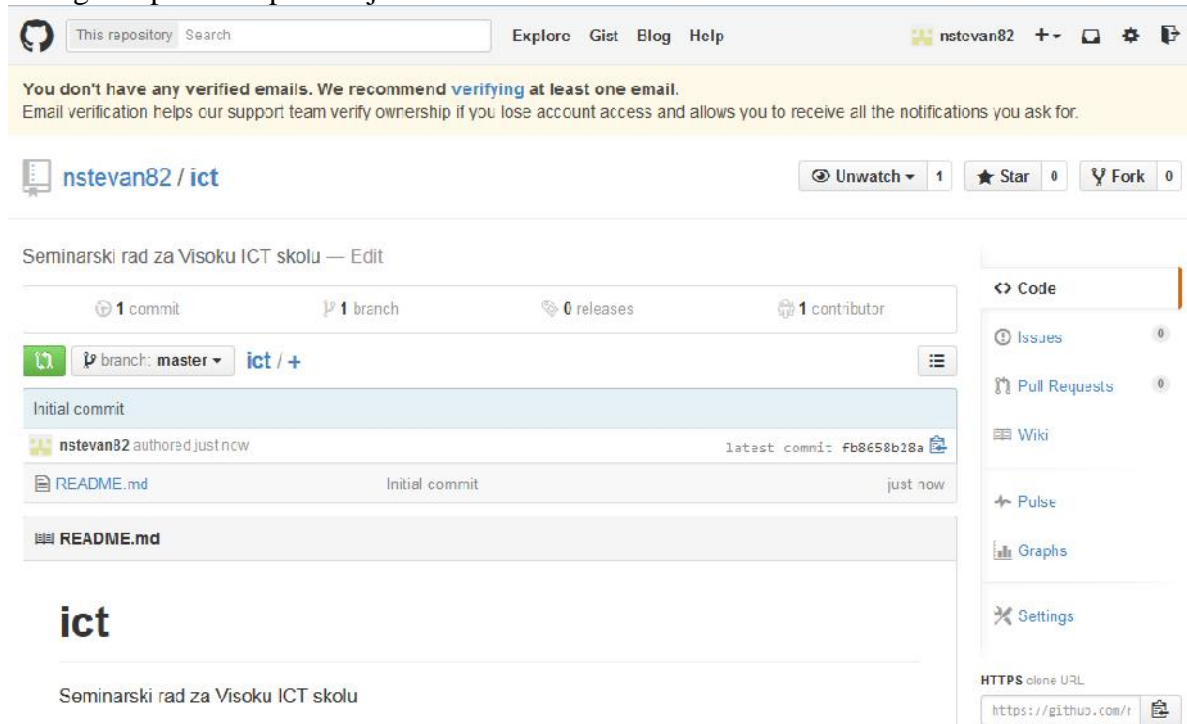
Popuniti formular za novi repozitorijum, a ako je registrovan kao besplatni nalog kliknuti na **public repository**.



Potom kliknuti na Create repository.

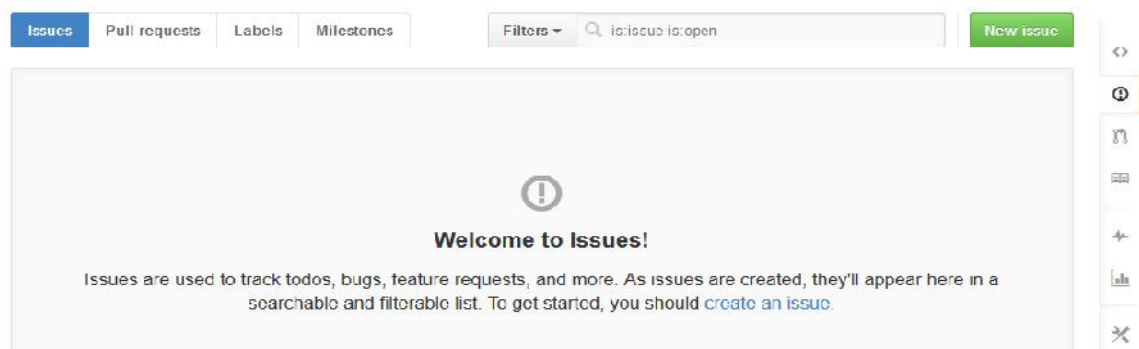


Ovako izgleda prazan repozitorijum:

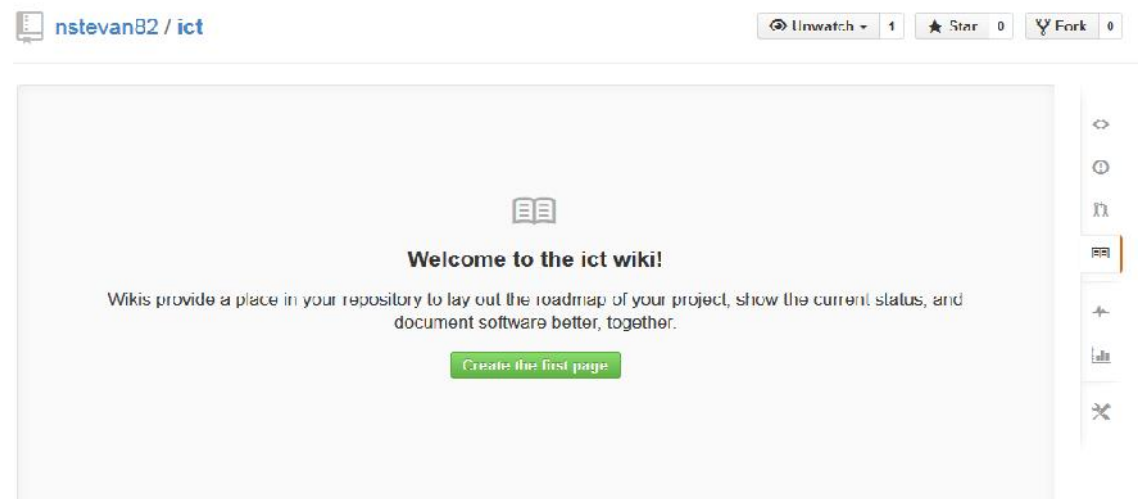


Sa desne strane se vidi nekoliko opcija:

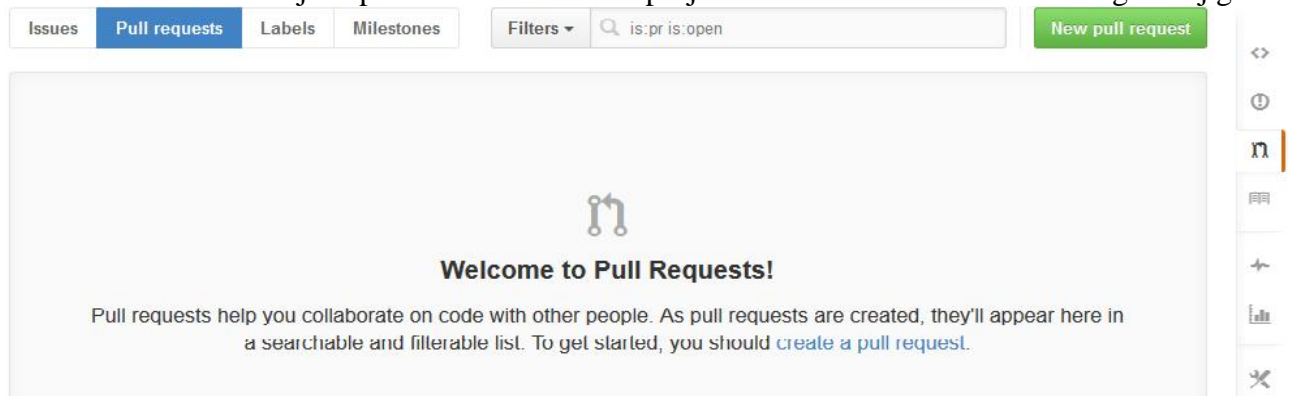
- **Issues** - govori o tome da li ima neki bag ili problem. Ljudi mogu da prijave bagove a korisnik da kaže šta je ispravljeno ili da to nije bio problem.



- **Graphs** - Grafici koji se koriste za izveštaje
- **Wikki** – mesto gde korisnik može da napiše informacije o source codu.



- Pull requests: Ukoliko neki korisnik primeti bag ili mogućnost da može da poboljša vaš projekat, on će klonirati ili forkovati vaš kod, uraditi nešto sa njime i zatražiti pull request, tj želi da kaže da je napravio neku izmenu u projektu i hoće tu izmenu da dodate glavnoj grani.



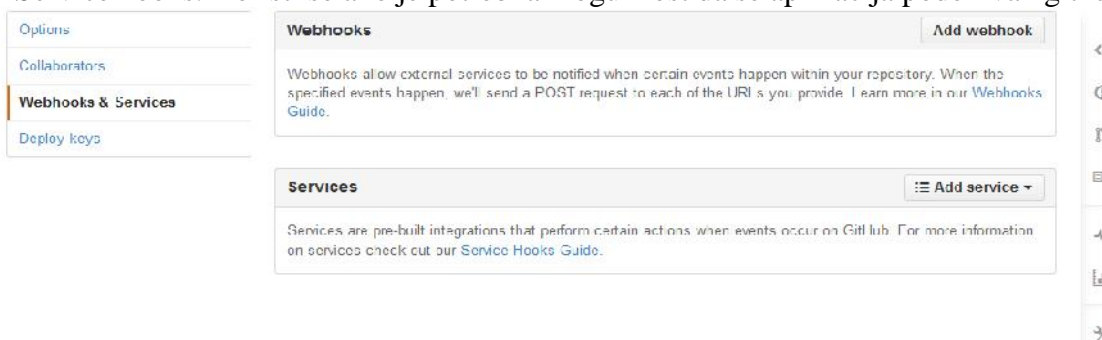
- Settings: svaki git projekat ima stranicu za administriranje. Ova stranica se odnosi samo na ovaj određeni repozitorijum. Svaki repozitorijum može imati nešto drugačiji izgled settings opcije pod ovim admin tabom.



- Collaborators - Kolaborator je korisnik koji sa vlasnikom naloga sarađuje na githubu. Ovde im se može dodati korisnik po imenu.

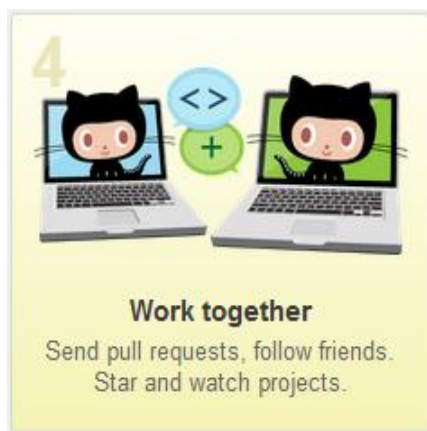


- Service hooks. Koristi se ako je potrebna mogućnost da se aplikacija podeli van githuba.



**Deploy keys** - Deploy key je SSH ključ koji se nalazi na serveru i dozvoljava pristup servera jednom repozitoriju na GitHubu. Ovaj ključ je vezan direktno sa repozitorijumom, a ne sa korisničkim nalogom. Deploy ključevi imaju full read/write pristup.

Na osnovnoj stranici se nalazi još jedna oblast – Work together. Klikom na ovu oblast dobijamo više funkcionalnosti:



### Društvenost

Jedna od bitnih karakteristika GitHub-a je mogućnost da se vidi na kome ljudi rade i sa kim su povezani.

Praćenje ljudi

Kada korisnik prati nekoga na GitHub-u, on dobija notifikacije na svom nalogu o njihovoj aktivnosti. Da bi se praćenje aktiviralo klikne se na Follow dugme na nalogu korisnika čije aktivnosti bi trebalo da se prate.



### Posmatranje projekta.

U nekom trenutku će korisnik želeći da bude obaveštavan o stanju određenog projekta. Kada se korisnik nalazi na web stranici projekta, može da klikne na Watch dugme u vrhu strane.



Kada vlasnik projekta ažurira projekat, korisnik koji je kliknuo na Watch opciju će videti na svojoj stranici šta se desilo.



Možda bi bilo interesantno pomenuti da se na **Explore GitHub** strani (<https://github.com/explore>) i **Trending strani** (<https://github.com/trending>) nalaze zanimljive sekcije sa novim projektima. Projekti se mogu obeležiti zvezdicom, da bi korisnik mogao kasnije ponovo da se vrati na te projekte. Kasnije je potrebno samo posetiti <https://github.com/stars> stranicu da bi se videla lista svih projekata koji su obeleženi zvezdicom.

## 4.0 Rad sa udaljenim repozitorijumom

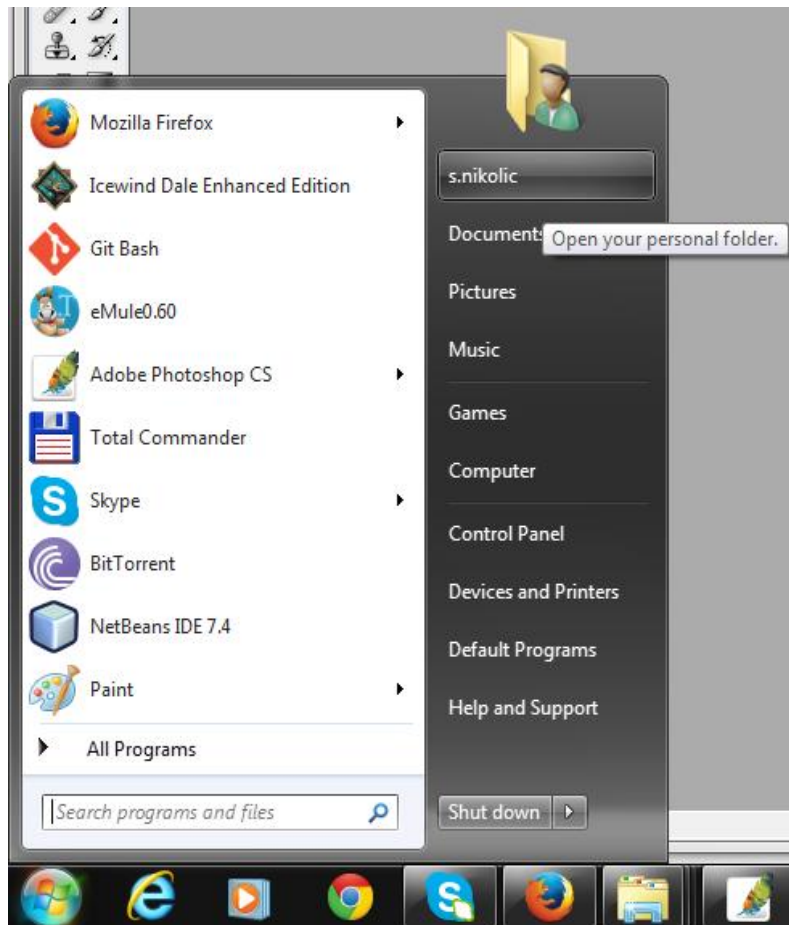
Da bi se povezao i radio sa githubom mora da postoji sigurna (secured) konekcija. Da bismo ovo omogu ili moramo da kreiramo SSH klju . SSH klju se sastoji od:

- Privatnog klju a: na našem PCu i
- Javnog klju a koji delimo sa githubom.

U kombinaciji sa Htpps-om dobija se sigurna konekcija.

KREIRANJE SSH klju a za github:

1. otvoriti li ni folder. (start pa u gornjem desnom uglu naziv naloga)



Treba proveriti da ovde ne postoji .ssh direktorijum. Ako postoji .ssh direktorijum treba uraditi bekap.

Otvoriti gitbash.

Uneti narednu komandu i kao parameter uneti email adresu registrovanu na githubu:

```
ssh-keygen -t -rsa -C "<vaša email adresa>"
```

RSA govori da želimo da generišemo RSA kod.

-C nam dozvoljava da dodamo komentar.

-t omogu ava definisanje tipa klju a koji e biti kreiran, a u ovom slu aju potrebno je kreirati RSA klju .

Nakon pritiska tastera enter ispisa e se adresa gde e klju biti snimljen. Zatim pritisnuti enter.

GitBash e sada tražiti kljuc nu frazu za lozinku.Ova lozinka e se tražiti svaki put kada se klju koristi. Potrebno je dva puta uneti lozinku zbog provere. Ovo je dupla autentifikacija i definitivno je važna.

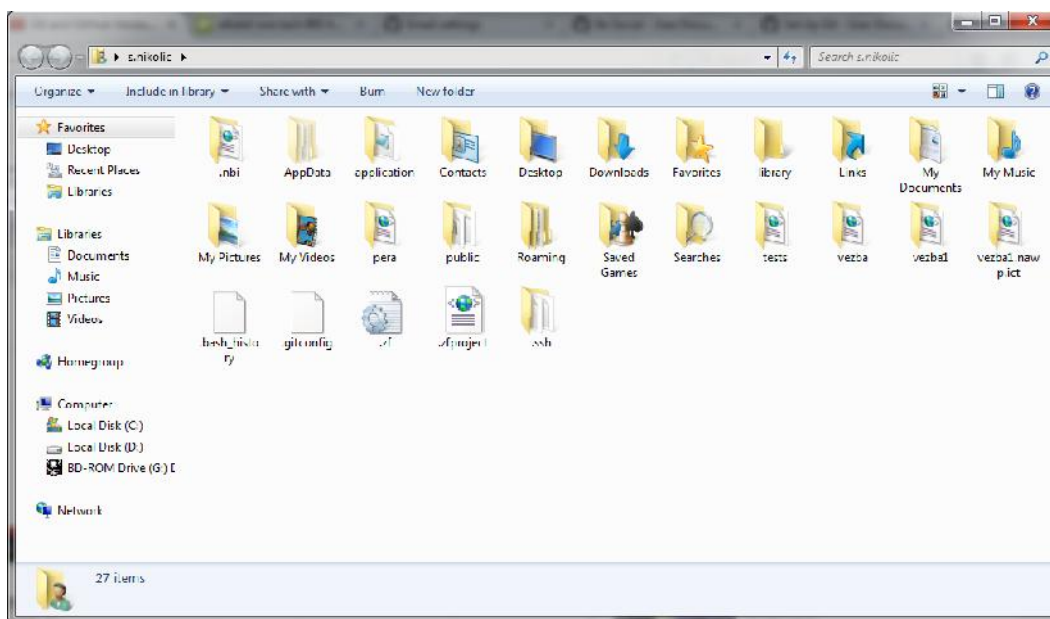


```
MINGW32/d/seminarski/github
s.nikolic@LAPTOP /d/seminarski/github
$ ssh-keygen -t rsa -C "admin@taraba.in.rs"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/s.nikolic/.ssh/id_rsa):
Created directory '/c/Users/s.nikolic/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/s.nikolic/.ssh/id_rsa.
Your public key has been saved in /c/Users/s.nikolic/.ssh/id_rsa.pub.
The key fingerprint is:
6d:14:b7:3f:59:fd:32:76:6b:ed:21:72:27:85:6e:cc admin@taraba.in.rs
The key's randomart image is:
+--[ RSA 2048 ]-----+
  .
  o .
  . . o
  o . . o
  S o .B.o
  . +..=
  . E =
  + =.
  .
+-----+

s.nikolic@LAPTOP /d/seminarski/github
$
```

Nakon ove akcije će se pojaviti folder .ssh u direktorijumu korisnikovog windows naloga i u njemu će biti vidljivi fajlovi:

- Id\_rsa – privatni ključ
- Id\_rsa.pub – javni ključ



Javni ključ se nalazi u fajlu id\_rsa.pub, potrebno je otvoriti fajl i kopirati ceo sadržaj iz njega. Otvoriti na nalogu githuba settings panel, pronaći opciju SSH keys, kliknuti na Add SSH key dugme, uneti naslov (npr svoj username) i paste-ovati svoj SSH ključ.

### Add an SSH Key

Title

Key

```
ssh-rsa
/AAnE
iuSSjnt
9R4olF
/np86O
D76msnVAjSpBP5
kVVVTDob9gds5B
K9Zltjqc23hIID7
```

Add key

Kliknuti na **Add key** dugme. Ovako je podešen javni ključ na githubu i privatni ključ na korisni kom računaru. Dobra je praksa proveriti da li sve radi, pa se u gitBashu kuca sledeća komanda:

```
Ssh -T git@github.com
```

GitBash će pitati da li želimo da nastavimo. (ukucati YES).

Tražiće nam password frazu i prikazati ekran slike ovome:

```

MINGW32:/d/seminarski/github
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ ssh -T git@github.com
Warning: Permanently added the RSA host key for IP address '192.30.252.128' to the list of known hosts.
Enter passphrase for key '/c/Users/s.nikolic/.ssh/id_rsa':
Hi nstevan82! You've successfully authenticated, but GitHub does not provide shell access.

s.nikolic@LAPTOP /d/seminarski/github
$

```

Ako se prikaže poruka *You have successfully authenticated, but GitHub does not provide shell access* kontakt sa udaljenim repozitorijum je ostvaren.

Na ovaj način smo testirali ssh da GitBash može da komunicira sa GitHubom.

Nakon ovoga u GitBash programu je potrebno da uđemo u direktorijum našeg projekta komandom:

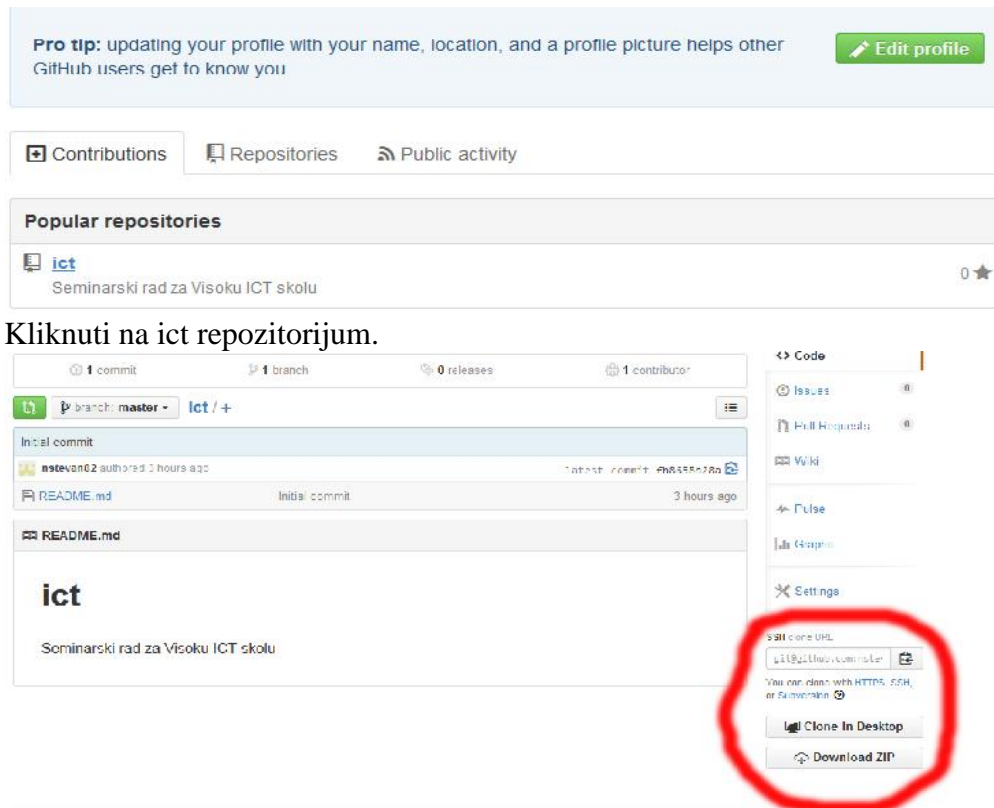
```
cd projekat
```

Da bi se registrovao udaljeni repozitorijum treba uneti komandu:

```
Git aremote add [naziv remote repozitorijuma] [remote url]
```

Na ovaj na in kažem GitBashu da želim da dodam remote repozitorijum, tako da ne želim da snimam ovo u lokalni repozitorijum.

U ovom primeru udaljeni repozitorijum se zove origin. Potrebno je navesti gde se nalazi ovaj udaljeni repozitorijum origin (origin je ovde kao primer). Adresa udaljenog repozitorijuma se nalazi na nalogu korisnika na GitHub sajtu.



Da bi se preuzeo SSH url potrebno je selektovati SSH tab u donjem desnom uglu stranice i kopirati adresu iz njega. Ovu adresu paste-ovati u GitBash nakon komande parametra origin na mesto parametra [remote url]:

```
MINGW32/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ ssh -T git@github.com
Warning: Permanently added the RSA host key for IP address '192.30.252.128' to the list of known hosts.
Enter passphrase for key '/c/Users/s.nikolic/.ssh/id_rsa':
Hi nstevan82! You've successfully authenticated, but GitHub does not provide shell access.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git remote add origin git@github.com:nstevan82/ict.git

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

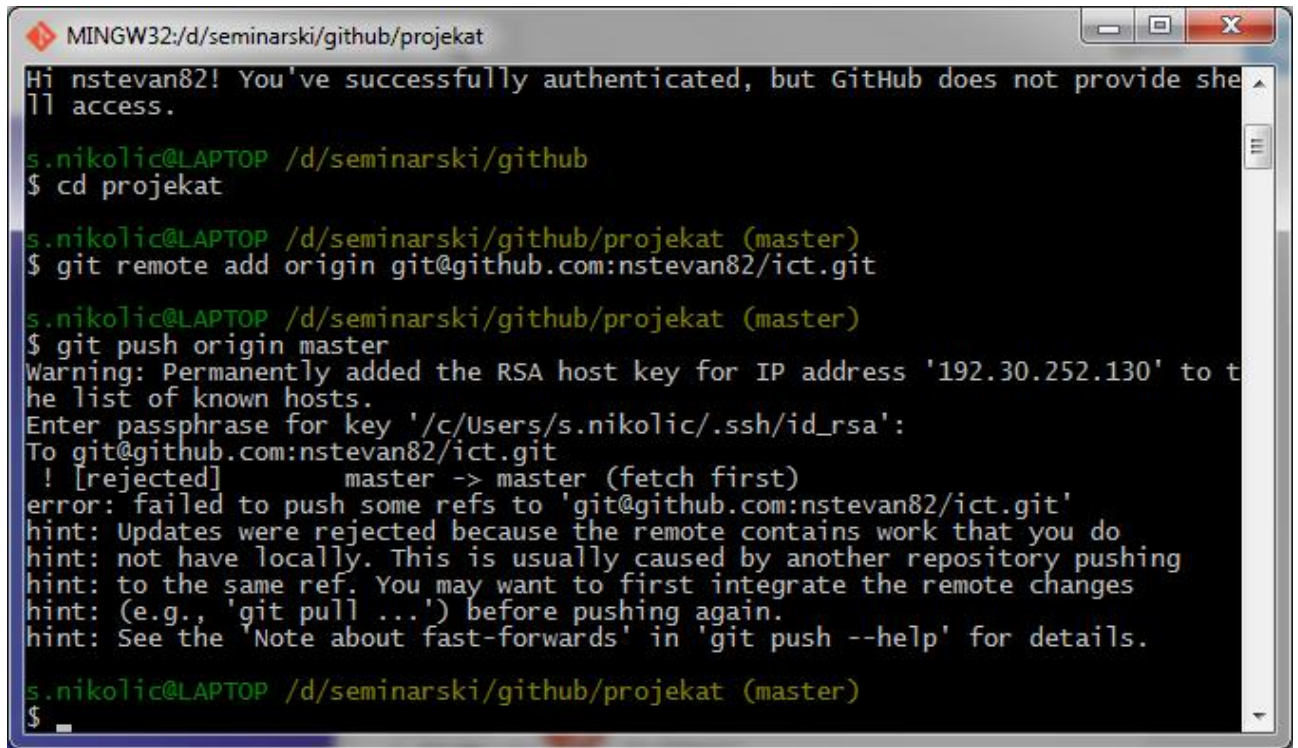
Na ovaj na in lokalni git zna gde je udaljeni repozitorijum origin.

U ovom trenutku je omogu ena komunikacija izme u GitBasha i udaljenog repozitorijuma. Da bi se itav sadržaj lokalnog repozitorijuma poslao udaljenom repozitorijumu potrebno je u konzolu uneti:



## Git push origin master

- Push – komanda kojom se govori da je potrebno slanje na udaljeni repozitorij.
- Origin – naziv mog remote repozitorijuma.
- Master – želim da uploadujem master granu (glavnu granu)



```
MINGW32:/d/seminarski/github/projekat
Hi nstevan82! You've successfully authenticated, but GitHub does not provide shell access.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git remote add origin git@github.com:nstevan82/ict.git

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git push origin master
Warning: Permanently added the RSA host key for IP address '192.30.252.130' to the list of known hosts.
Enter passphrase for key '/c/Users/s.nikolic/.ssh/id_rsa':
To git@github.com:nstevan82/ict.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'git@github.com:nstevan82/ict.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

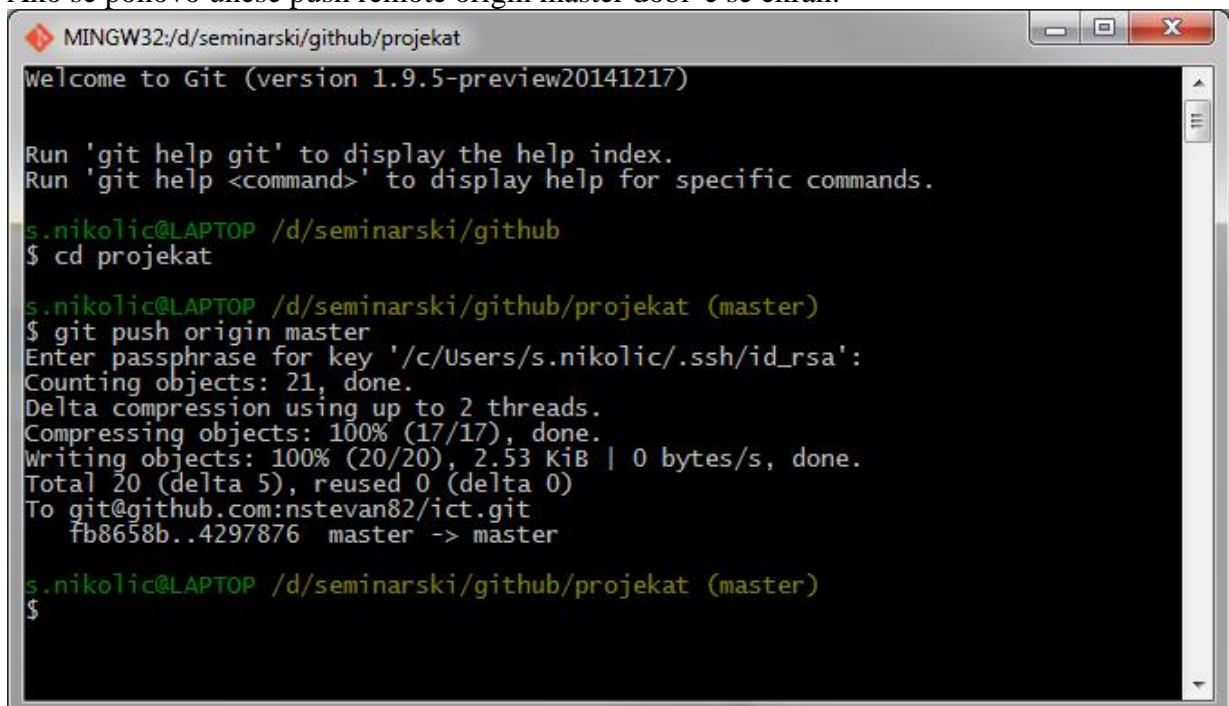
s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

U slučaju sa gore navedene slike GitHub nam govori da je slanje odbijeno jer udaljeni repozitorijum sadrži fajlove koje nemamo lokalno (u ovom slučaju je to readme.md). Potrebno je najpre da se preuzme taj fajl na lokalni repozitorijum. Zbog toga je umesto push komande potrebno uneti:

## Pull remote origin master

Sada su fajlovi prilagođeni.

Ako se ponovo unese push remote origin master dobiće se ekran:



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git push origin master
Enter passphrase for key '/c/Users/s.nikolic/.ssh/id_rsa':
Counting objects: 21, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (17/17), done.
Writing objects: 100% (20/20), 2.53 KiB | 0 bytes/s, done.
Total 20 (delta 5), reused 0 (delta 0)
To git@github.com:nstevan82/ict.git
 fb8658b..4297876 master -> master

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Ovo govori da je sve poslato na udaljeni repozitorijum. Možemo proveriti tako što otvorimo svoj nalog na GitHubu i pogledamo listu fajlova:

The screenshot shows the GitHub interface for a repository named 'ict' by user 'nsteven82'. At the top, there's a navigation bar with 'This repository' selected, a search bar, and links for 'Explore', 'Gist', 'Blog', and 'Help'. A yellow banner below the navigation bar states: 'You don't have any verified emails. We recommend verifying at least one email. Email verification helps our support team verify ownership if you lose account access and allows you to receive all the notifications you ask for.' Below this, the repository name 'nsteven82 / ict' is displayed with buttons for 'Unwatch', 'Star' (0), and 'Fork' (0). The main content area shows the repository's metadata: '7 commits', '1 branch', '0 releases', and '1 contributor'. A merge commit is highlighted: 'Merge branch 'master' of github.com:nsteven82/ict:' by 'wakawaki' 4 minutes ago. Below the merge commit, a list of files is shown with their commit history: 'README.md' (Initial commit, 4 hours ago), 'game.js' (prvi commit, a day ago), 'index.html' (prvi commit, a day ago), 'init.js' (prvi commit, a day ago), 'readme.txt' (pocetni komit, 7 minutes ago), and 'style.css' (prvi commit, a day ago). On the right side, there's a sidebar with links to 'Code', 'Issues' (0), 'Pull Requests' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. At the bottom of the sidebar, the 'SSH clone URL' is provided: 'git@github.com:nsteven82:ict'.

Sada se vide svi fajlovi.

Ukoliko se izmeni neki fajl (npr index.html) desi e se slede a situacija:

Doda se preko git add . sve fajlove na pripremnu listu.

Pomo u git commit -m "izmena index.html" commituje se sve na lokalni repozitorijum.

Pomo u git push origin master pošalje se sadržaj lokalnog repozitorijuma na remote repozitorijum.

Može se videti nalogu na githubu, na listi komitova izmena koja se zove izmena index.html.

The screenshot shows a diff view for the file 'index.html'. It highlights the changes made in the commit 'izmena index.html' by 'wakawaki' 33 seconds ago. The diff shows a single addition: a new line of HTML code for a canvas element. The code is: <canvas id="myCanvas" width="550" height="600"> Vas pregledac ne podrzava HTML5 canvas tag. pokusajte sa drugim pregledacem </canvas>. The diff is shown in a green highlight.

Ako se klikne na izmena index.html vide e se izmenjena verzija fajla.

The screenshot shows the file diff view for 'izmena index.html' by 'wakawaki' 3 minutes ago. It shows the changes made in the commit 'izmena index.html' by 'wakawaki' 3 minutes ago. The diff shows a single addition: a new line of HTML code for a canvas element. The code is: <canvas id="myCanvas" width="550" height="600"> Vas pregledac ne podrzava HTML5 canvas tag. pokusajte sa drugim pregledacem </canvas>. The diff is shown in a green highlight. Below the diff, there's a button 'View' to see the full file.

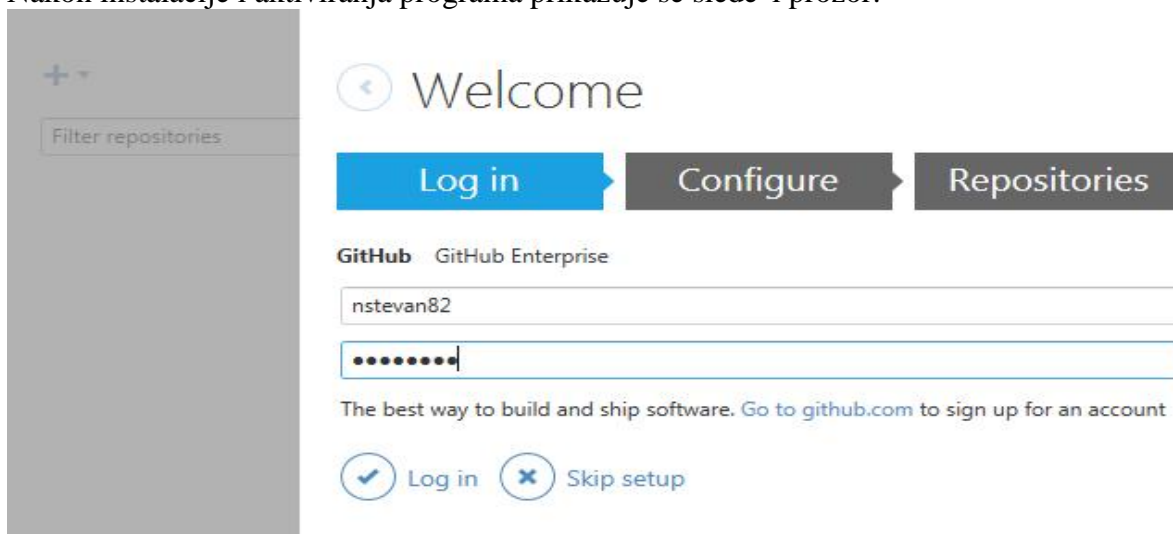


## 5.0 Windows GUI GitHub aplikacija

Da bi se preuzela GitHub, vizuelna verzija programa potrebno je posetiti sajt [www.git-scm.com](http://www.git-scm.com). Potrebno je kliknuti na windows GUI link. Link vodi do stranice <https://windows.github.com/>. I ovde se može preuzeti Github for Windows (klikom na zeleno dugme).

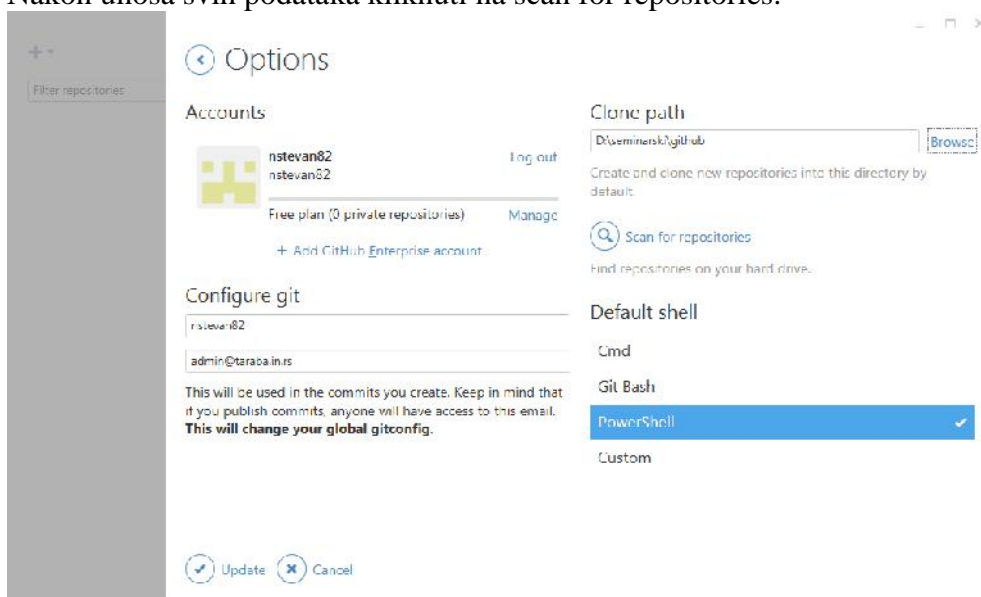


Nakon instalacije i aktiviranja programa prikazuje se sledeći prozor:

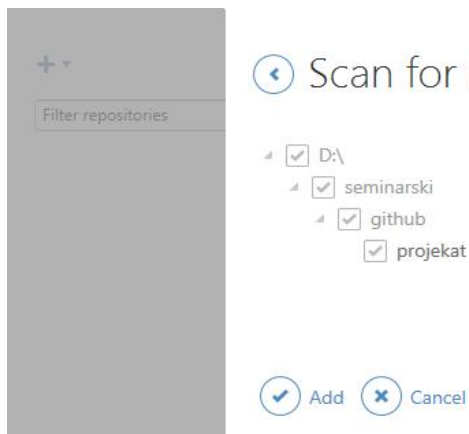


Potrebno je uneti korisničko ime i lozinku i kliknuti na log in. Nakon sveže instalacije može se primetiti da se ne vide repozitorijumi.

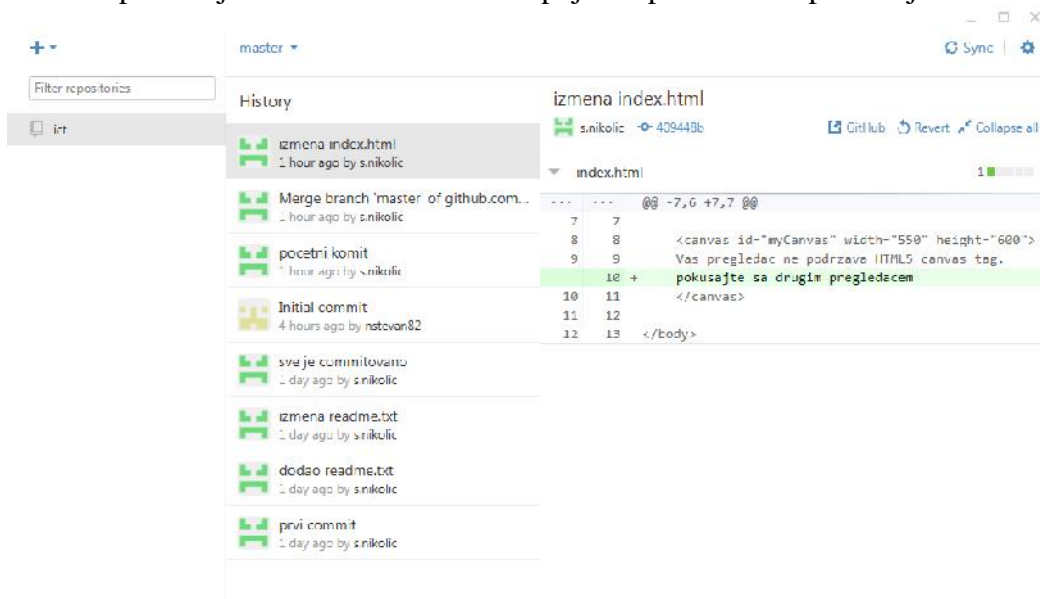
U opciji config se podešava nalog i putanja do lokalnog direktorijuma sa projektima (Clone path). Nakon unosa svih podataka kliknuti na scan for repositories.



Ukoliko su podaci pravilno uneseni aplikacija će pronaći lokalni repozitorijum. Nakon toga je potrebno kliknuti na add pa na update da bi aplikacija zapamtila lokalni repozitorijum.



U listi repozitorijuma sa leve strane ce se pojaviti pronađeni repozitorijum.



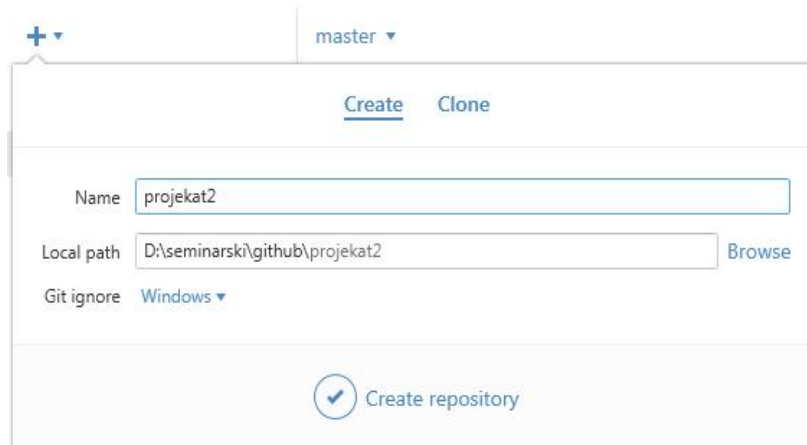
Da bi se videli svi commitovi na nekom repozitorijumu treba kliknuti na ime repozitorijuma (u ovom primeru je kliknuto na ICT jer je to naziv repozitorijuma ovog projekta).

## 5.1 Kreiranje novog projekta

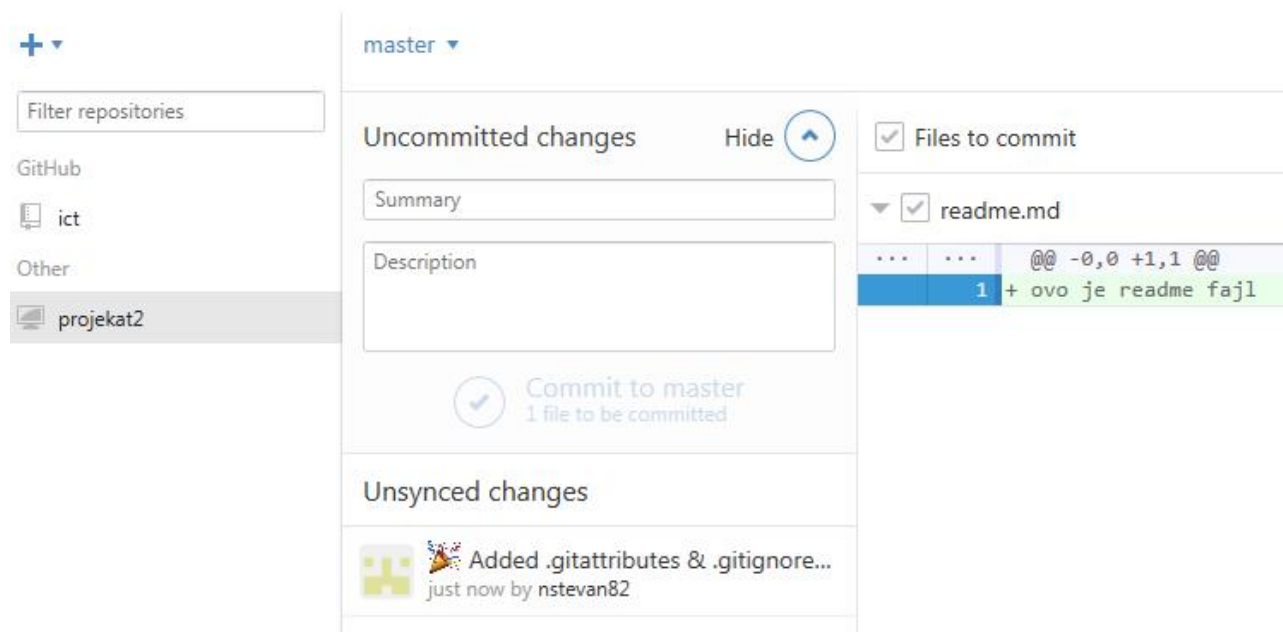
U ovoj sekciji će biti opisan rad sa potpuno novim projektom sa kojim će se raditi u aplikaciji sa grafički korisničkim interfejsom. U direktorijumu sa projektima se kreira direktorijum projekat2.

Uobičajena praksa je da se kreira readme.md fajl, te je to i urađeno u samom projektu.

Za iniciranje novog lokalnog repozitorijuma kliknuti na krstić u gornjem levom uglu. Softver će nakon unosa imena projekta sam generisati lokalnu putanju projekta.

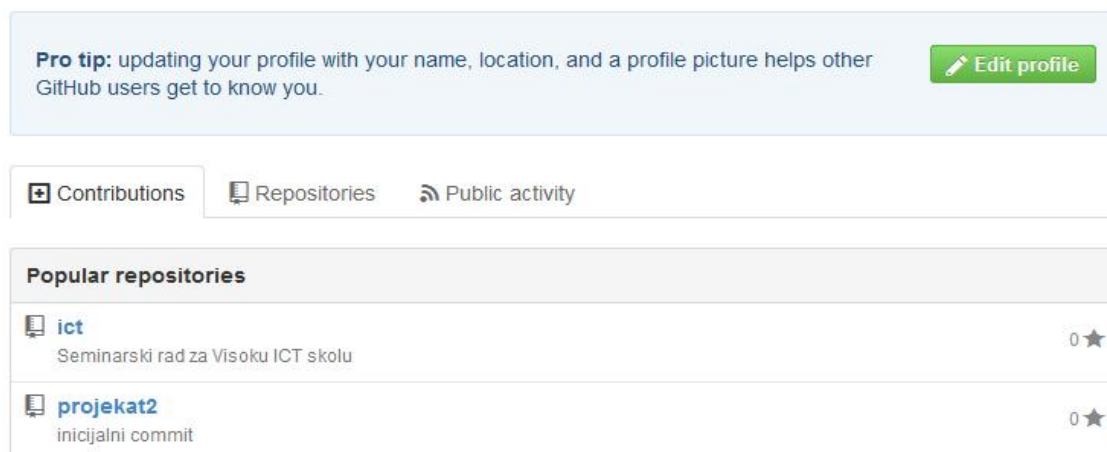


Ukoliko su uneti podaci ispravni kliknuti na Create repository. Prikaza e se sl. prozor:

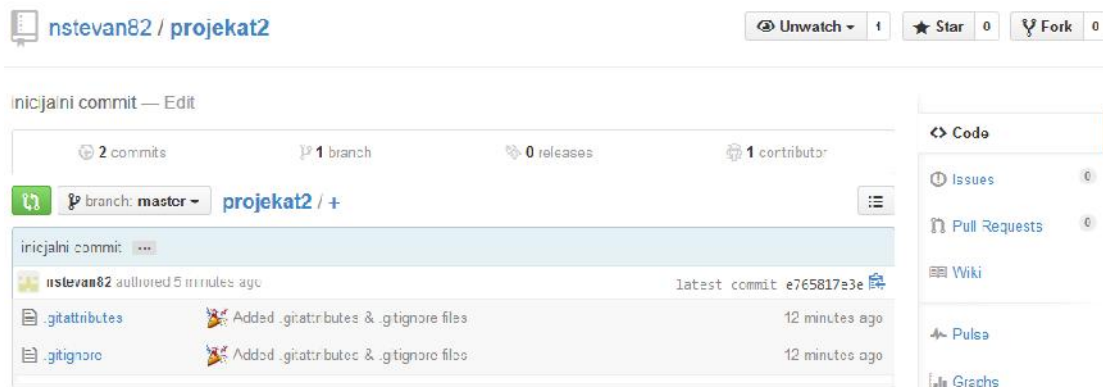


Sa leve strane se može videti lista repozitorijuma, u sredini su polja za commitovanje novih promena, a sa desne strane promene koje su na injene. U donjem delu sredine prozora se može vidi lista promena koja nije sinhronizovana sa GitHubom. U ovom konkretnom primeru dodati su .gitattributes i .gitignore fajlovi, a to se dešava automatski pri iniciranju novog repozitorijuma iz ovog softerskog alata.

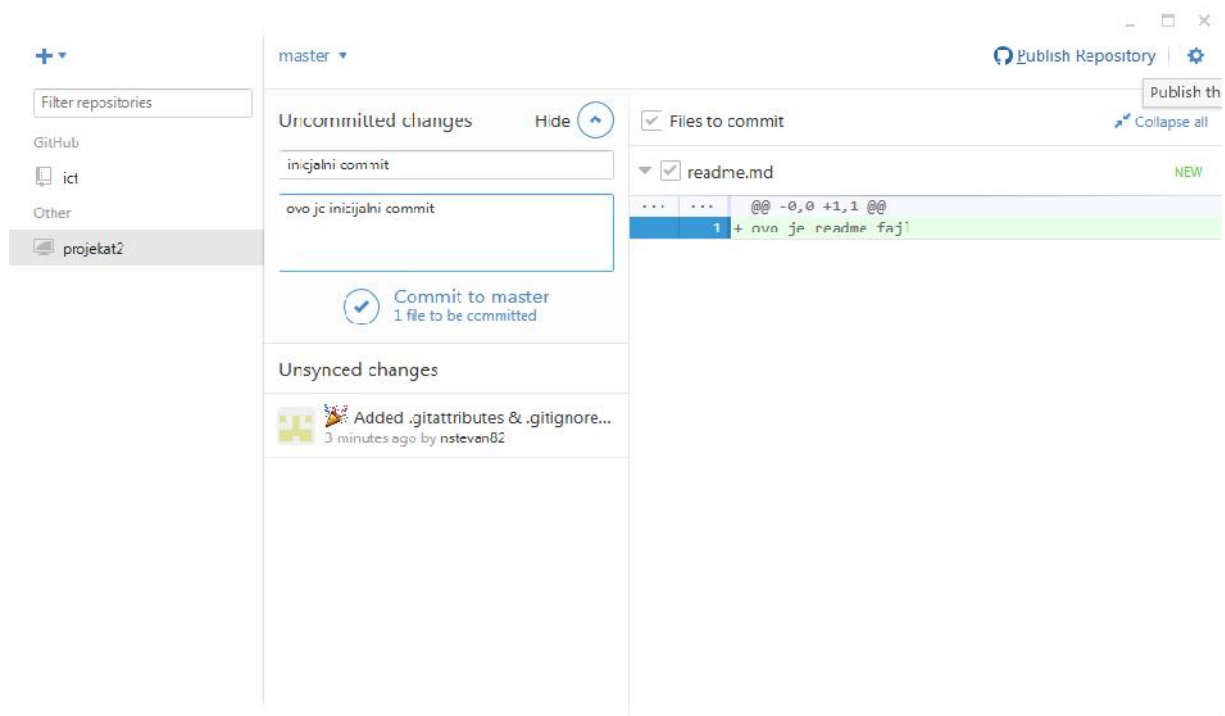
Na GitHub sajtu je sada vidljiv novi repozitorijum projekta pod nazivom **projekat2**.



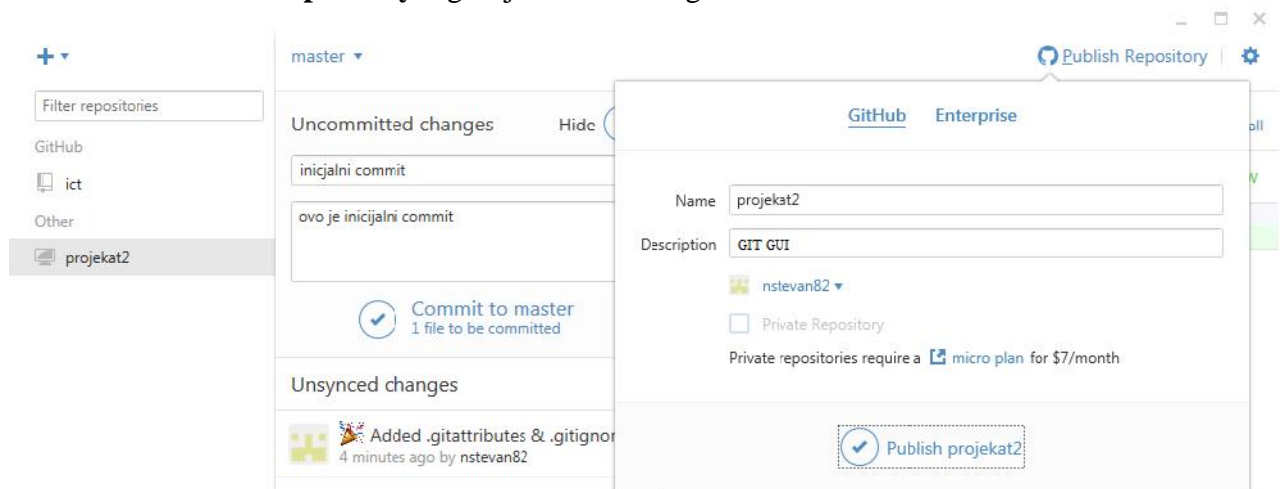
Kada se klikne na naziv novog projekta mogu se videti po etne promene (.gitattributes i .gitignore)



Sada je potrebno uraditi inicijalni commit da bi se promene poslale na lokalni repozitorijum. U odgovarajuća polja uneti naslov commita i komentar. Nakon toga kliknuti na Commit to master.

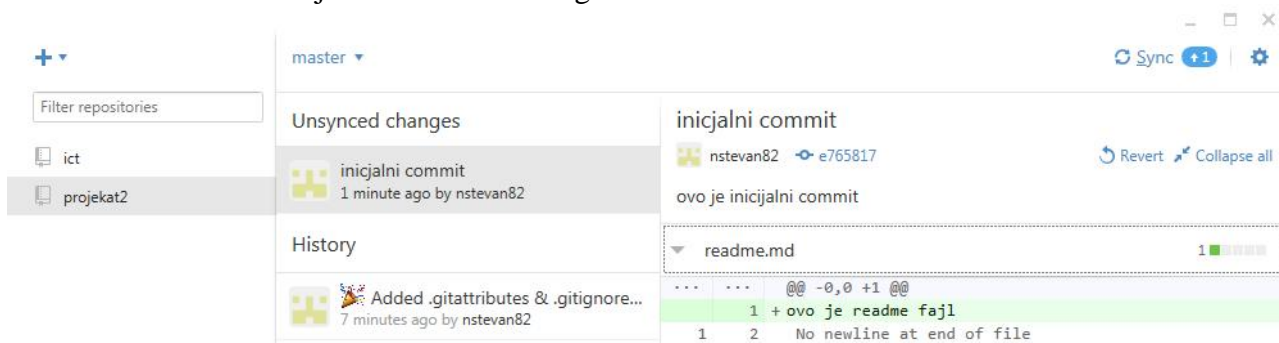


Kada su promene poslate na lokalni repozitorijum, potrebno je objaviti ih i na udaljenom repozitorijumu. Najpre udaljeni repozitorijum mora biti prijavljen. Da bi se ovo učinilo treba kliknuti na **Publish Repository** u gornjem desnom uglu.

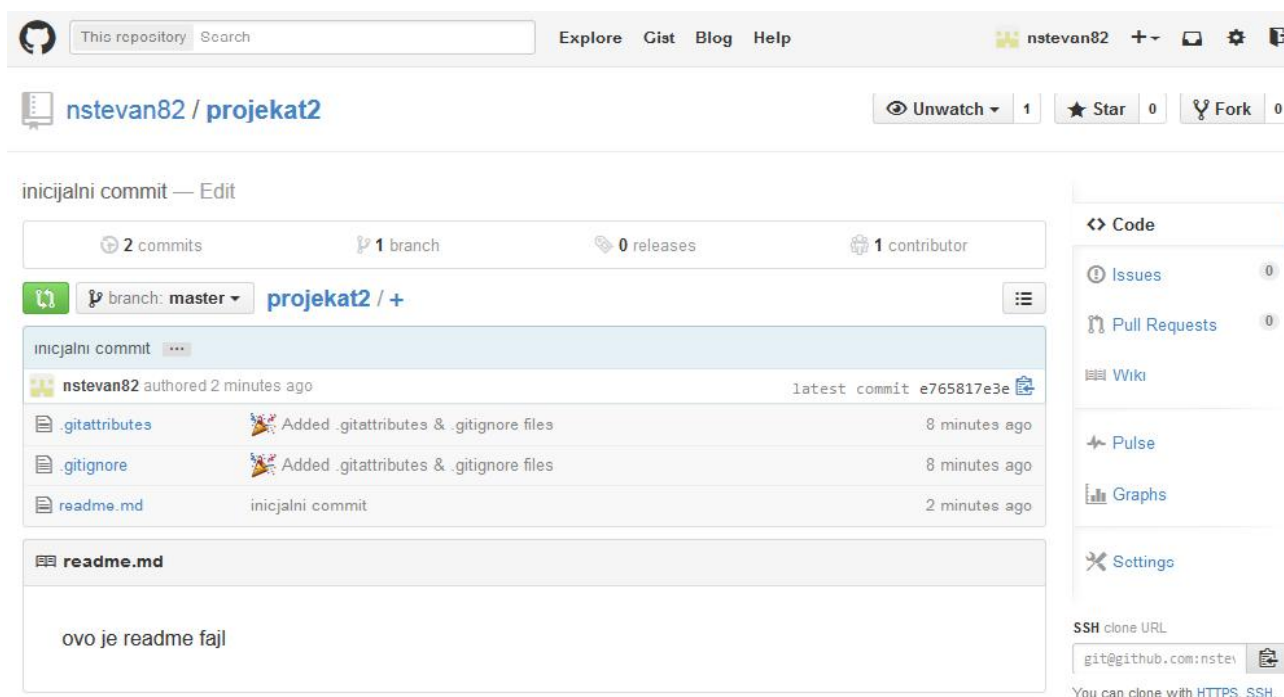


Nakon što se popuni forma imenom repozitorijuma i opisom, pritisne se **Publish [naziv udaljenog repozitorijuma]**

Da bi se sadržaj lokalnog i udaljenog repozitorijuma sinhronizovali potrebno je kliknuti na opciju Sync u gornjem desnom uglu prozora. Za razliku od svog konzolnog pandana, ovaj softver će uraditi Pull ili Push akciju u zavisnosti od toga šta GitHub zahteva.



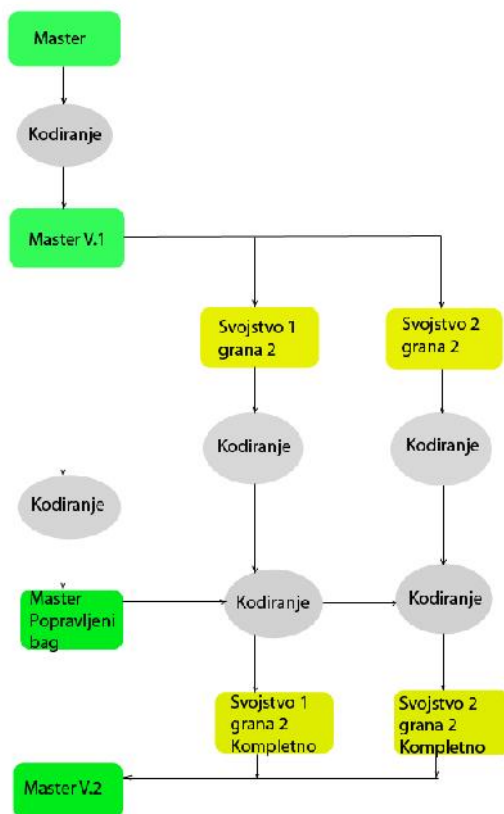
Na GitHub nalogu, na stranici repozitorijuma je sada vidljiv i inicijalni commit fajla readme.md.





## 6.0 Branching, forking i merging.

U realnim situacijama se za izradu jednog projekta aktivira više timova ljudi. Jedan tim bi mogao da radi na verziji 1.0 programa sa ciljem da se ta verzija što pre završi i objavi na tržištu, dok drugi tim ve paralelno radi na verziji 2.0 koja e imati neke dodatne karakteristike koje nisu planirane u verziji 1.0.



## 6.1 Branching primer u radu sa GitBash-om

U narednom tekstu će biti opisan primer rada na projektu koristeći git i grananje.

Projekat se otpo inje sa master granom. Pretpostavimo da je cilj uraditi program da bi se objavlva verzija 1. Pošto je jedan od ciljeva da prvo izdanje bude objavljeno u kratkom periodu nije predvi eno da sve stvari da u u u prvo izdanje. Druga objava e biti mnogo ve i projekat i sav postoje i kod e se ubaciti u drugu verziju. U drugoj verziji e biti dodat kod sa novim funkcionalnostima. Prvi tim e raditi na verzijama za verziju 1, a Drugi tim ce raditi na funkcionalnostima namenjenim drugoj verziji. Za rad na ovom projektu bi e potrebne 2 grane.

Prvi tim radi na verziji 1, a drugi tim nema kod, pa se morati da klonira kod sa master grane, kreira se sopstvenu granu i tada mogu da krenu sa radom na stvarima za verziju 2. Svako se radi na svojoj verziji, ali potencijalna situacija je da se jednog dana desi bag. Menadžer se drugom timu reči da ovaj bag mora biti ispravljen u drugoj verziji. Osnovna ideja je da se bag ispravi što pre verziji i primeni u verziji 2, pa se drugi tim zaustavi rad na verziji 2 i posvetiti se ispravci бага dok prvi tim radi na verziji 1. Postoji mogućnost da se uradi ispravka бага, pa da se ubaci u verziju 1.1. Nakon ispravke бага i kreiranja verzije 1.1 može da se nastavi da se razvija verzija 2.

Zbog ispravke бага i dobijanja verzije 1.1 , master grana se promenila u odnosu na prethodnu verziju projekta pa se morati da se spoje obe grane da bi se sinhronizovale. Da bi se ovo uradilo koristi se merge komanda. Ovo je možda najvažnija funkcionalnost zbog koje se koristi git. Nakon spajanja grana dobra je praksa izvršiti testiranje fajlova na integritet fajlova da bi se uverili da sve radi. Nakon toga se nastavlja razvoj projekta do finalnog kreiranja verzije 2.0.

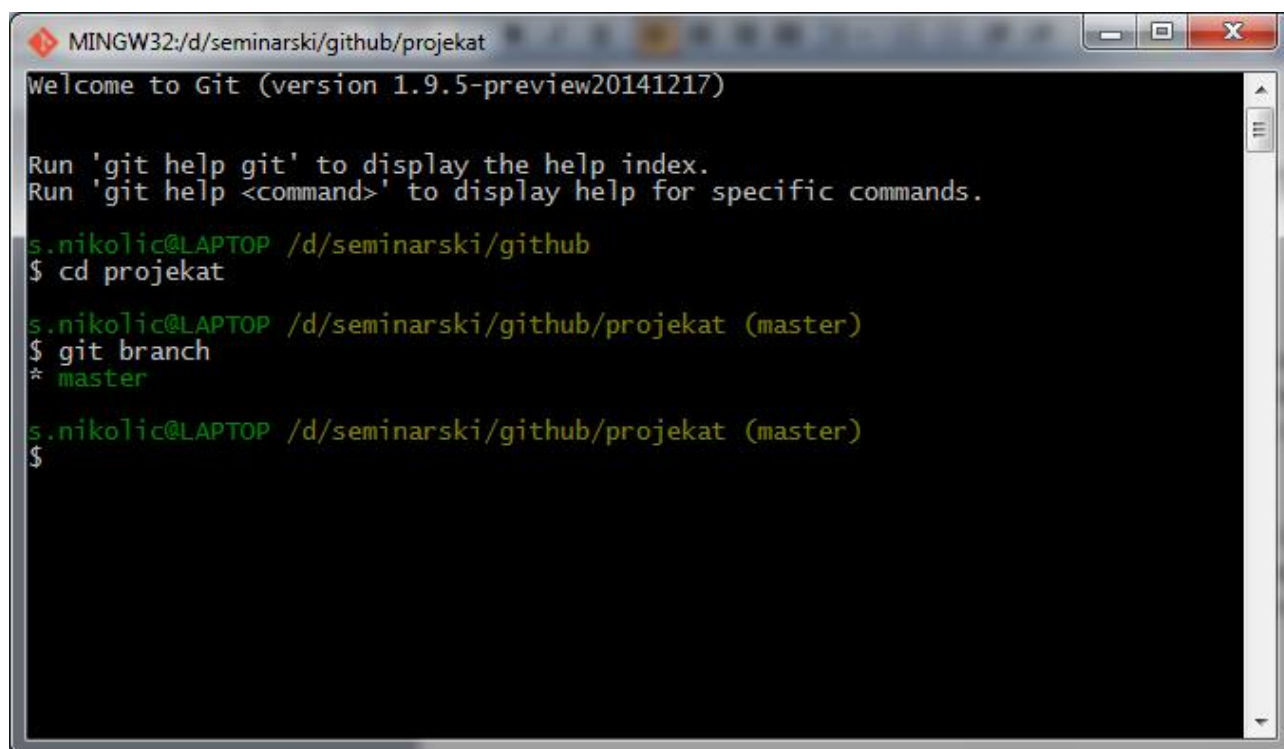
Realizacija primera:

Pretpostavimo da je cilj prvog tima da napravi index.html stranicu, a drugi tim da radi na dodatnoj funkcionalnosti koja će biti kontakt strana, a plan je da se ona implementira u drugoj verziji.

Prvi tim radi na index.html a drugi na contact.html.

Za prikaz liste svih grana u repozitorijumu uneti sledeću komandu:

Git branch



```
MINGW32:/d/seminarski/github/projekat
Welcome to Git (version 1.9.5-preview20141217)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

s.nikolic@LAPTOP /d/seminarski/github
$ cd projekat

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$ git branch
* master

s.nikolic@LAPTOP /d/seminarski/github/projekat (master)
$
```

Kreiranje novih grana omogućiti rad Tima 2 na kodu bez izmena na master grani.

Nova grana se kreira unosom sledećih komandi:

Git branch “<naziv grane>”

Da bi se trenutni korisnik prebacio na korišćenje te grane treba uneti:

Git checkout “<naziv grane>” prebacice nas na tu granu.

Kada bi se ponovo unelo git branch na ekranu bi zelenom bojom bila obojena grana na kojoj se nalazi korisnik, a vidi se i naziv u zagradi.

Tim 2 na ovoj grani nema fajlove projekta pa je potrebno da ga preuzme sa udaljenog repozitorijuma. Da bi se ovo uradilo u konzoli je potrebno uneti:

GIT CLONE <SSH link>

GitBash obično traži lozinku posle ove akcije.

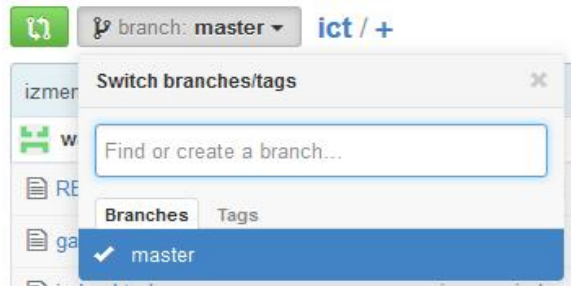
Kada korisnik tima 2 skine fajlove na lokalni repozitorijum, on mora da napravi svoju granu.

U ovom primeru je napravljena grana **r2\_contact**:

Git branch “r2\_contact”

Kasnije e u toku rada na projektu, kada je potrebno commitovane fajlove poslati na udaljeni repozitorijum na granu r2\_contact, korisnik uneti

```
git push origin "r2_contact"
```



Mogu e je pogledati novokreiranu granu na sajtu GitHub-a i nakon slanja fajlova na udaljeni repozitorijum može se videti i contact.html na r2\_contact grani, a treba napomenuti da ovaj fajl ne postoji na master grani jer ga je drugi tim postavio na svojoj r2\_contact grani.

Paralelno sa timom 2, tim jedan radi na grani branch2\_index, na index.html fajlu.

Tim 1 e napraviti fajl fajl, podesiti sve što je potrebno u GtBashu i poslati lokalni repozitorijum na ovu granu komandom:

```
Git push branch2_index
```

U ovom trenutku oba tima rade na svom delu projekta. U potencijalnoj situaciji pojavi e se menadžer i re i da postoji bag koji mora da se ispravi. Tim dva e ispraviti bag i izmenjeni kod poslati na master granu:

```
Git push origin master
```

Da bi se prebacile promene sa master grane na r2\_index potrebno je koristiti komandu merge.

```
Git merge "<naziv grane>"
```

U ovom konkretno primeru kuca se git merge "master" nakon što se korisnik pozicionira na r2\_index granu. Kada grane budu spojene, mogu e je nastaviti sa razvojem verzije 2.

Tim 2 zna da je master grana promenjena (merge-ovana). Sada tim 2 mora da uradi PULL da bi se sinhronizovao kod udaljenog repozitorijuma sa kodom lokalnog repozitorijuma.

Nakon toga bi trebalo da se prebacimo na **r2\_contacts granu**.

Tim 2 e morati da uradi git PULL master i nakon toga Git merge master.

Nakon toga e tim 2 izvršiti izmena koda u contact.html tj. doda se kod koji bi trebalo da bude u verziji 2. Zatim e to sve biti spojeno u master grani i treba pripremiti kod za objavu verzije 2.

Uradimo git checkout master da bi se prebacili na master granu.

Za spajanje master i r2\_contacts grane potrebno je sa master grane uneti:

```
Git merge r2_contacts.
```

Sada master grana ima sav kod iz naše r2\_contacts grane. Kodiranje je završeno i prvi tim može da aktivira MERGE komandu. Ovo je dobar trenutak da se sve promene da pošalju na github udaljeni repozitorijum.

```
git push origin r2_contacts
```

```
git push origin master.
```

Napomena: kada se radi u timu dobra praksa je uraditi PULL pre PUSH-a da ne bi doslo do konflikta kada se radi PUSH. PULL e sinhronizovati lokalni repozitorijum sa githubom.

U ovom trenutku je poznato da je drugi tim završio sa svojim programom i da je spojio svoju granu sa master granom.

Sada je potrebno uraditi PULL i pokupiti fajlove sa master grane da bi korisnik mogao da pošalje fajlove sa svojim izmenama. Da bi se ovo u inilo potrebno je uneti komandu:

Git pull origin master.

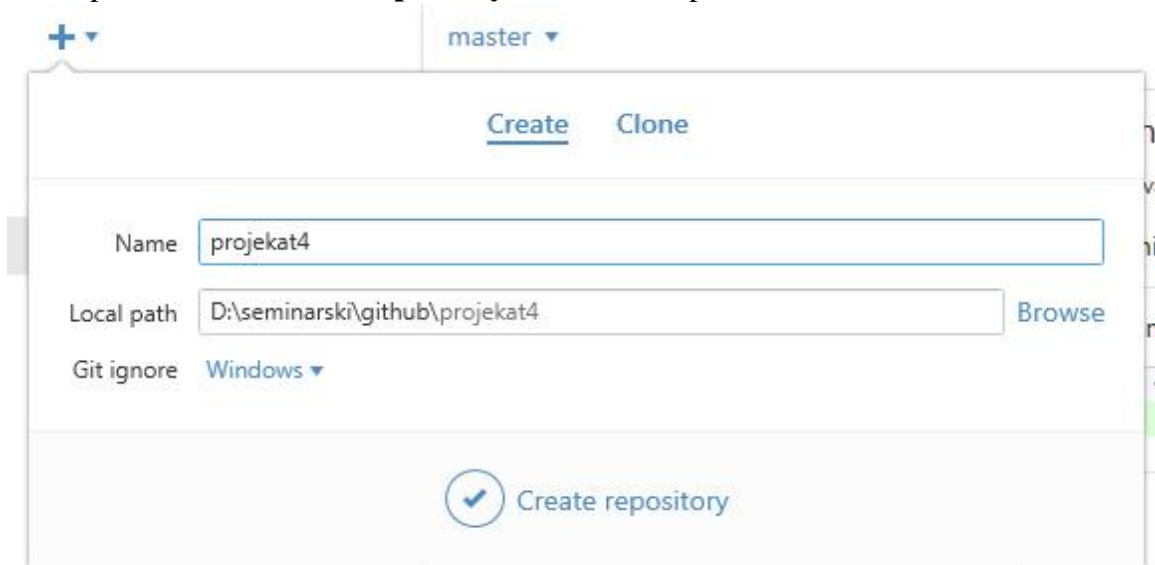
Da bi se spojile **r2\_index** i master grana iz master grane uneti:

git merge r2\_index.

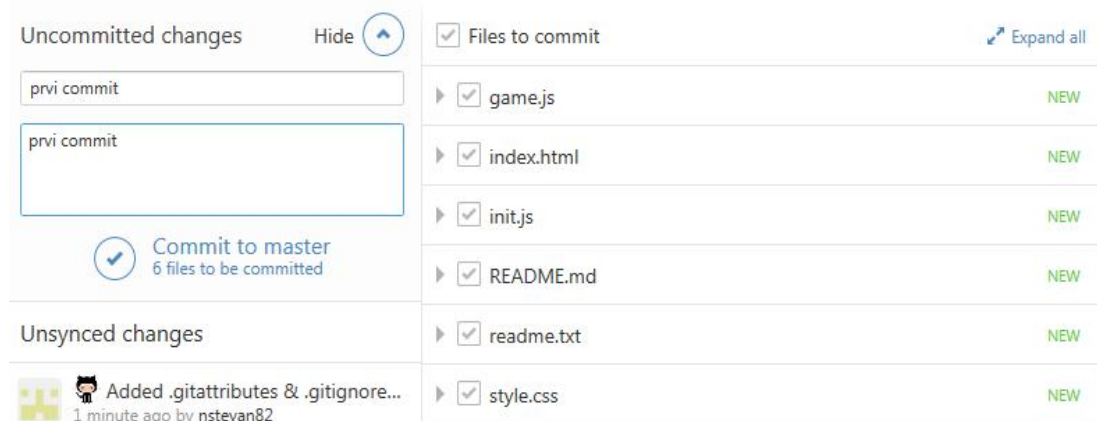
Sada master sadrži promene sa obe grane i sve je spremno za integracione testove i objavu verzije 2 softvera na kome je radio tim 2.

## 6.2 Grananje u windows GUI aplikaciji

U ovom delu rada e biti prikazan rad sa grananjem u vizuelnom okruženju. Ovo e biti demonstrirano na potpuno novom projektu. Potrebno je inicijalizovati novi projekat u direktorijumu **projekat4**. Kliknuti na znak plus u gornjem levom uglu i nazvati novi lokalni repozitorijum **projekat4**, pa kliknuti na **create repository** kao na slici ispod.



Na master grani treba napraviti inicijalni commit:



Napravimo malu izmenu u **index.html**, npr. u ovom primeru sam dodao span tag sa porukom. Sledi commitovanje izmena da bi se simuliralo slanje verzije 1 programa na lokalni repozitorijum.

Uncommitted changes

Hide

Index izmenjen

Dodat span i index završen za verziju 1

Commit to master

1 file to be committed

Unsynced changes

prvi commit

2 minutes ago by nstevan82

Added .gitattributes & .gitignore...

5 minutes ago by nstevan82

Files to commit

index.html

...	...	@@ -11,6 +11,7 @@
11	11	</canvas>
12	12	
13	13	</body>
	14	+ <span>ovo je canvas</span>
14	15	<script type="text/javascript" src="init.js"
15	16	<script type="text/javascript" src="game.js"
16	17	</html>
17	18	\ No newline at end of file

Da bi se sadržaj lokalnog repozitorijuma poslao udaljenom repozitorijumu mor da se odabere Publish Repository u gornjem desnom uglu prozora. Github je u GUI verziji zamenio termin PUSH terminom PUBLISH. Nakon popunjene forme kliknuti na publish projekat4.

Publish Repository

GitHub

Enterprise

Name

projekat4

Description

nstevan82

Private Repository

Private repositories require a [micro plan](#) for \$7/month

Publish projekat4

U ovom trenutku je vidljiv **projekat4** na GitHub sajtu, tj. na korisni kom nalogu koji se trenutno koristi.

Pro tip: updating your profile with your name, location, and a profile picture helps other GitHub users get to know you.

Edit profile

Contributions

Repositories

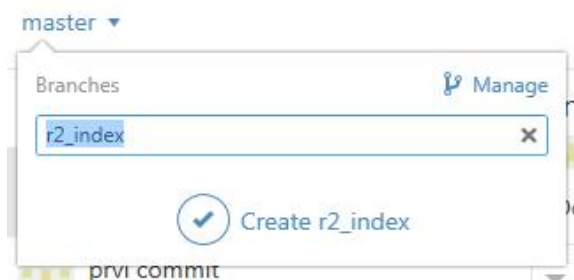
Public activity

Popular repositories

<div>ict</div> <div>Seminarski rad za V.soku ICT skolu</div>	0 ★
<div>projekat2</div> <div>inicijalni commit</div>	0 ★
<div>projekat4</div> <div>projekat4</div>	0 ★



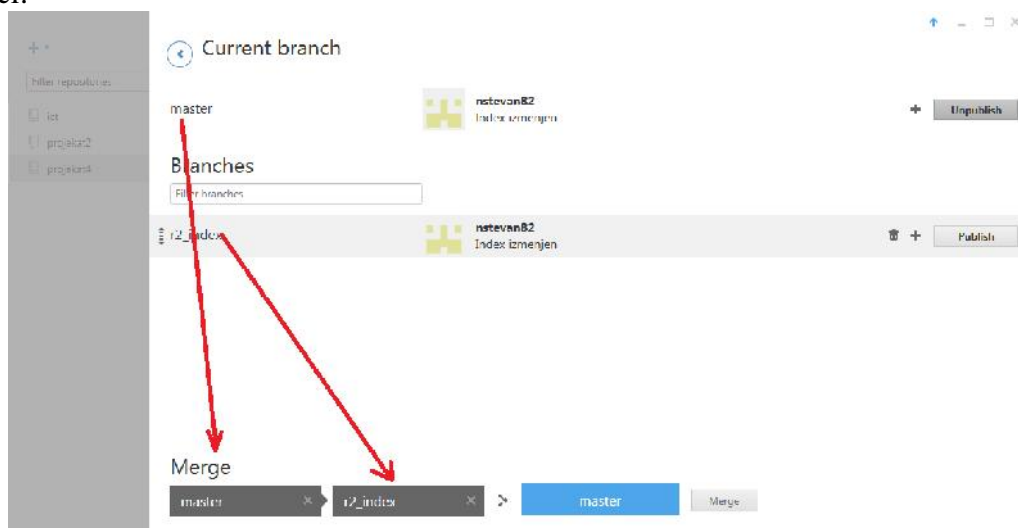
Kada postoji udaljeni repozitorijum, mogu e je kreirati novu granu. U gornjem delu prozora treba kliknuti na master, pa ce se otvoriti prozor za unos nove grane.



Nakon unosa nove grane, git e se automatski prebaciti na tu granu.

Napomena: kada se aktivira git commit u vizuelnom gitu pojavi se .gitignore fajl.

Fajl .gitignore koristi git na taj na in što ignoriše fajlove koji odgovaraju pravilu u .gitignore. Ignorisani falovi su uglavnom fajlovi za konfigurisanje IDE programa, kao što je npr. Eclipse, ije promene ne bi trebalo da budu zanimljive te se s toga ne prate. Fajlovi koji se ve posmatraju ne e biti skinuti sa liste pra enja. Ukoliko je potrebno spojiti dve grane u jednu treba kliknuti opet na master granu, ali se ovog puta odabere manage u gornjem desnom uglu prozora. Pojavi e se prozor kao na slici:



Da bi se spojile dve grane potrebno je prevu i svaku od grana u po jednu ku icu koja se nalazi u donjem delu prozora i kliknuti na dugme MERGE. Nakon ovoga je potrebno izvršiti sinhronizaciju.

## 6.2 Forking

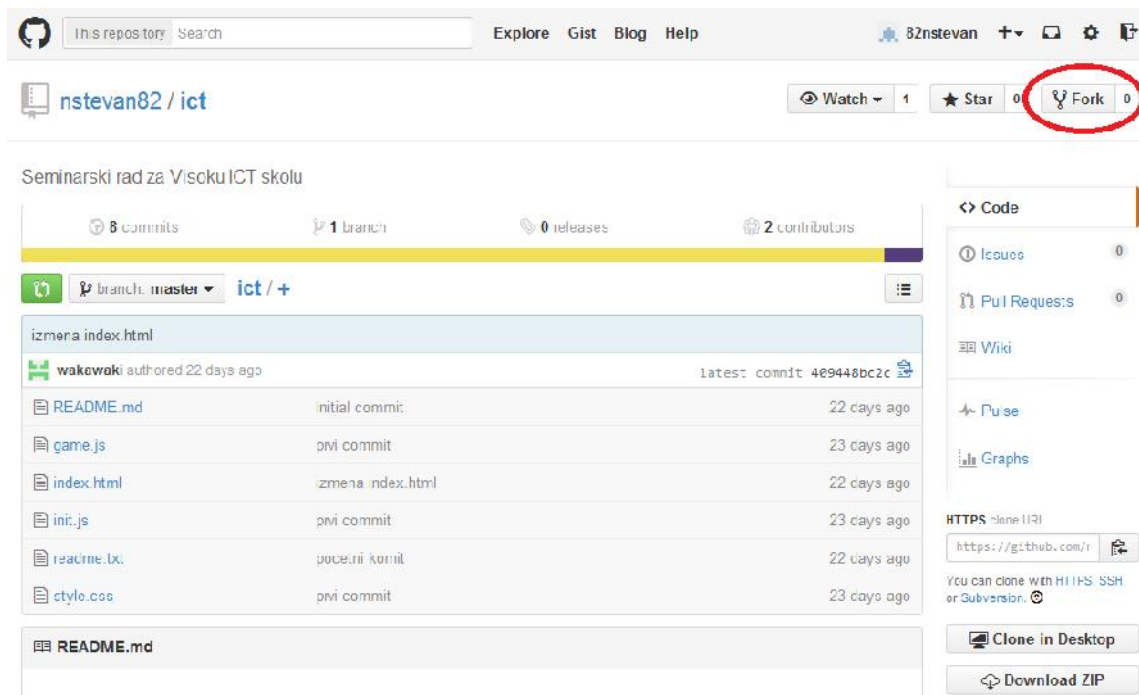
Forking nije git termin niti git komanda, to je pojava pri radu sa GitHub-om:

Kada korisnik želi da doprinese projektu nekog drugog korisnika

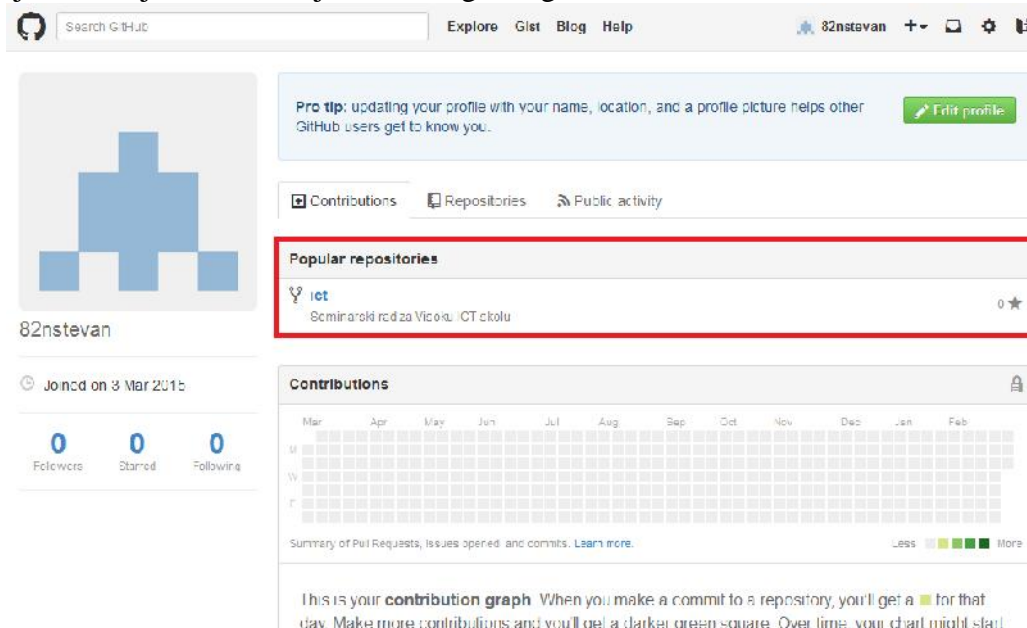
Kada korisnik želi da promeni pravac projekta u potpuno druga ijem smeru.

Primer:

Moj sadašnji nalog je nstevan82. Otvorio sam još jedan nalog na githubu pod imenom 82nstevan da bih simulirao nalog drugog korisnika. Ukoliko ulogovan pod nalogom 82nstevan pristupim projektu ICT naloga nstevan82 vide e se slede a strana:



U gornjem desnom uglu ekrana se može videti opcija Fork. Ova opcija se koristi ako je korisnik zainteresovan da doda nešto ovom projektu. Klikom na opciju Fork Github e klonirati taj projekat u moj nalog **82nsteven** na kome sam trenutno logovan. Kada se sada vratim na stranicu naloga **82nsteven** vide e se da je projekat ICT deo mog naloga i ispred naziva projekta e biti indikator da je ovaj projekat dobijen fork-ovanjem sa tu eg naloga:



Nakon izvršenih izmena na projektu, mogu e je spojiti te izmene sa originalnim projektom na nalogu nstevan82. Ova funkcionalnost se aktivira tako što se u projektu na nalogu 82nsteven odabere Pull Requests opcija.

Seminarski rad za Visoku ICT skolu — Edit

8 commits 1 branch 0 releases 2 contributors

branch: master ict / +

This branch is even with nstevan82:master Pull Request Compare

izmena index.html

wakawaki authored 22 days ago latest commit 409448bc2c

README.md	Initial commit	22 days ago
game.js	prvi commit	23 days ago
index.html	izmena index.html	22 days ago
init.js	prvi commit	23 days ago
readme.txt	pocetni komit	22 days ago
style.css	prvi commit	23 days ago

README.md

Code

**Pull Requests** 0

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

https://github.com/82nstevan/ict

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Klikom na **Pull Requests** prikaza e se ekran za unos komentara

base fork: nstevan82/ict base: master head fork: 82nstevan/ict compare: master

dodat komentar

Write Preview Markdown supported Edit in fullscreen

Dodao sam komentar u index.html fajl koji mislim da će pomoći vašem projektu

Attach images by dragging & dropping or selecting them

Able to merge. These branches can be automatically merged.

Create pull request

1 commit 1 file changed 0 commit comments 1 contributor

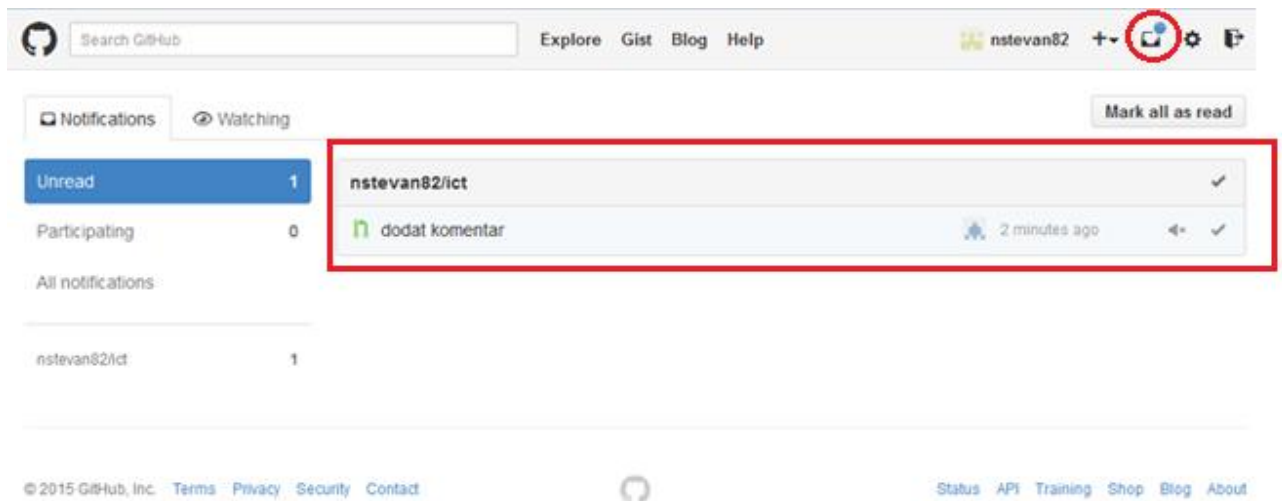
Commits on Mar 03, 2015

82nstevan dodat komentar d3d72e5

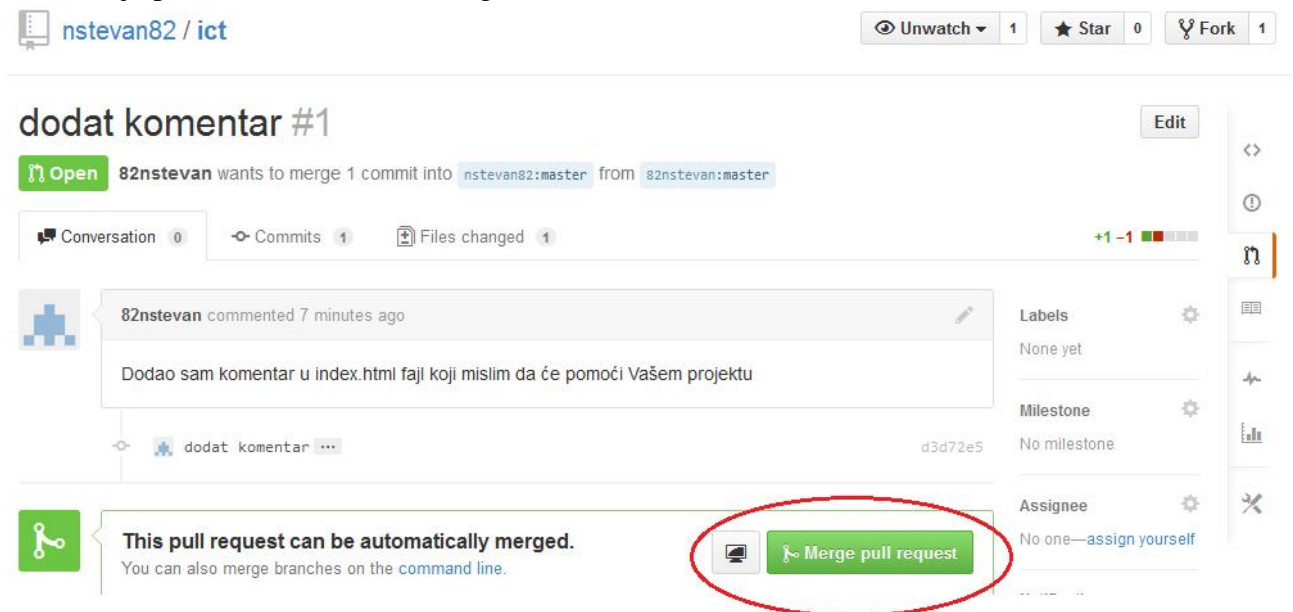
Showing 1 changed file with 1 addition and 1 deletion.

Unified Split

Nakon poslatog pull requesta, korisnik naloga kome je poslat pull request (u ovom sluaju je to nstevan82) e dobiti notifikaciju u gornjem desnom uglu githuba da je dobio novi pull request.



Kada se otvori poruka može se videti telo poruke, izmena koja je izvršena i dugme **Merge pull requests**. Ukoliko vlasniku ovog projekta odgovara da se ova izmena priključi njegovom projektu može da je potvrdi klikom na ovo dugme.



## 7.0 Zaključak

Github je koristan alat za programere koji rade timski. Dobro je što se fajlovi nalaze na udaljenom serveru i lokalnim računarima što umanjuje mogućnost gubitka ili oštećenja fajlova. Ova usluga može biti besplatna što još više pomaže širenju primene ovog alata. U ovom radu nisu opisane sve mogućnosti Git-a ali je već na ovom nivou moguće koristiti Git u realnim projektima. Meni se posebno dopalo što je sintaksa GitBasha intuitivno napravljena tako da se komande lako pamte. Verujem da će se ovaj alat sve više primenjivati u budućnosti i da pravi programerski tim ne može da funkcioniše bez ovakvog softvera.