



دانشگاه صنعتی نوشیروانی بابل

دانشکده مهندسی برق و کامپیوتر

پروژه برای دریافت درجه کارشناسی در رشته
مهندسی کامپیوتر گرایش مهندسی نرم افزار

ورژن کنترل

نگارش

الیاس عباسی - کیهان رعیتی

استاد راهنما

دکتر حمید جزایری

دی ۱۳۹۷





دانشگاه صنعتی نوشیروانی بابل

دانشکده مهندسی برق و کامپیوتر

شناسنامه پروژه کارشناسی

عنوان: ورژن کنترل

نام و نام خانوادگی دانشجو: الیاس عباسی - کیهان رعیتی

شماره دانشجویی: ۹۵۳۱۲۰۰۵۲-۹۴۳۲۲۰۰۵۹

تاریخ ارایه (دفاع): ۱۳۹۷/۱۰/۲۰

استاد راهنما: دکتر حمید جزایری استادیار دانشگاه صنعتی نوشیروانی بابل

چکیده

چکیده بنویس در مورد موضوع و هدف

واژه‌های کلیدی: گیت ، ورژن کنترل



تأییدیه صحت و اصالت نتایج و مالکیت مادی و معنوی

اینجانب **الیا س عباسی - کیهان رعیتی** به شماره دانشجویی ۹۵۳۱۲۰۰۵۲-۹۴۳۲۲۰۰۵۹ رشته **مهندسی کامپیوتر** دانشکده **دانشکده مهندسی برق و کامپیوتر** در مقطع **کارشناسی** تأیید می‌نمایم که کلیه نتایج این **پروژه** تحت عنوان **ورژن کنترل** به استاد راهنمایی **دکتر حمید جزایری** حاصل کار اینجانب و بدون هرگونه دخل و تصرف است و موارد نسخه‌برداری شده از آثار دیگران را با ذکر کامل مشخصات منبع ذکر کرده‌ام. در صورت اثبات خلاف مندرجات فوق، به تشخیص دانشگاه مطابق با ضوابط و مقررات حاکم (قانون حمایت از حقوق مؤلفان و مصنفان و قانون ترجمه و تکثیر کتب و نشریات و آثار صوتی، ضوابط و مقررات آموزشی، پژوهشی و انضباطی...) با اینجانب رفتار خواهد شد و حق هرگونه اعتراض درخصوص احقاق حقوق مکتسب و تشخیص و تعیین تخلف و مجازات را از خویش سلب می‌نمایم. در ضمن، مسئولیت هرگونه پاسخگویی به اشخاص اعم از حقیقی و حقوقی و مراجع ذیصلاح (اعم از اداری و قضایی) به عهده‌ی اینجانب خواهد بود و دانشگاه هیچ‌گونه مسئولیتی در این خصوص نخواهد داشت. در ضمن تمام دستاوردهای مادی و معنوی حاصله از **پروژه** متعلق به **دانشگاه صنعتی نوشیروانی بابل** می‌باشد و اینجانب هیچ گونه ادعایی در قبال آن ندارم.

نام و نام خانوادگی: **الیا س عباسی - کیهان رعیتی** امضا



دانشگاه صنعتی نوشیروانی بابل

دانشکده مهندسی برق و کامپیوتر

مجوز بهره‌برداری از پایان‌نامه

بهره‌برداری از این پایان‌نامه در چهارچوب مقررات کتابخانه و با توجه به محدودیتی که توسط استاد راهنما به شرح زیر تعیین می‌شود، بلامانع است:

- ☐ بهره‌برداری از این پایان‌نامه برای همگان و با ذکر منبع، بلامانع است.
- ☐ بهره‌برداری از این پایان‌نامه با اخذ مجوز از استاد راهنما و با ذکر منبع، بلامانع است.
- ☐ بهره‌برداری از این پایان‌نامه تا تاریخ _____ ممنوع است.

استاد راهنما: دکتر حمید جزایری امضا

فهرست مطالب

ت	چکیده
۱	۱ آموزش ورژن کنترل
۲	۱-۱ مقدمه
۳	۲-۱ تاریخچه
۳	۳-۱ کنترل منابع (Source control)
۳	۴-۱ گیت (Git)
۳	۱-۴-۱ شروع کار با گیت
۵	۲-۴-۱ بررسی و دستیابی
۸	۳-۴-۱ اعمال تغییرات
۱۰	۴-۴-۱ توزیع
۱۰	۵-۴-۱ گسترش
۱۲	نمایه
۱۲	چکیده انگلیسی

فهرست شکل‌ها

۱-۱	چگونگی ساختن گیت bash	۳
۲-۱	اعمال دستور init بر روی یک شاخه دلخواه در محیط گیت bash	۴
۳-۱	دانلود روند همین پروژه از گیت لب از پروژه مشترکان	۵
۴-۱	گیت استوس بر روی روند مقاله فعلی بر شاخه git	۶
۵-۱	مشاهده کامیت‌های انجام شده بر روی مقاله کنونی تا بدینجای کار با دستور	
۷	گیت log	
۶-۱	به ورژن‌های قبلی	۸
۷-۱	برگشت به ورژن master	۸
۸-۱	مثالی از گیت add	۹
۹-۱	مثالی از گیت reset	۹
۱۰-۱	مثالی از گیت commit	۹

فصل ۱

آموزش ورژن کنترل

۱-۱ مقدمه

ورژن کنترل چیست ؟ و چرا باید به آن اهمیت دهیم ؟ ورژن کنترل یک سیستم ذخیره سازی یک یا چند فایل در طول زمان است [۱] ورژن کنترل سیستمی است که به توسعه دهندگان نرم افزار کمک می کند تا علاوه بر امکان مشارکت روی پروژه های نرم افزاری، بتوانند به تاریخچه ای از کدهایی که قبلاً نوشته اند نیز دست پیدا کنند و به طور کلی اهداف استفاده از سیستم های ورژن کنترل را می توان در موارد زیر خلاصه نمود: - فراهم آوردن فرصتی برای توسعه دهندگان به منظور کار کردن به صورت هم زمان - مجزاسازی نسخه های توسعه داده شده اختصاصی تک تک توسعه دهندگان - نگهداری تاریخچه ای از هر نسخه از هر چیزی که به اشتراک گذاشته شود. با استفاده از ورژن کنترل شما می توانید ایده های جدید خود را بدون نگرانی آزمایش کنید و در صورت نیاز به ورژن های قبلی برگردید [۲]. ورژن کنترل سیستمی ضروری برای کار گروهی بر روی یک پروژه نرم افزاری است [۳]. گیت یک نرم افزار کنترل نسخه و از مدل نرم افزارهای متن باز برای بازنگری کد منبع توزیع شده و مدیریت منبع کد است که برای دنبال کردن تغییر فایل های کامپیوتری و دنبال کردن کارهای انجام شده روی آن ها توسط افراد مختلف است. که در تمامی سیستم عامل های اصلی توسعه داده شده است [۴]. گیت یک راه قدرتمند برای ردیابی و مقایسه نسخه ها، رفع خطاها، کشف رویکردهای جدید به شیوه ای ساختاری است [۵]. گیت ابتدا برای توسعه لینوکس توسط لینوس توروالدز به وجود آمد و اکنون پروژه های فراوانی از آن الهام گرفته اند. هر دایرکتوری کاری در گیت یک مخزن کامل با تاریخچه کامل تغییرها و قابلیت بازنگری آن ها است و برای کار با آن نیازی به دسترسی به شبکه یا سرور مرکزی وجود ندارد. امروزه برنامه نویسی که به گیت مسلط نباشد را عملاً برنامه نویس نمیدادند، به عبارت دیگر تسلط به گیت وظیفه ی هر برنامه نویس میباشد. در هر پروژه ای در هر سطحی در دنیا و در هر شرکتی از گیت استفاده میشود حتی شرکت های غیر برنامه نویسی مانند مقاله نویسی، پایان نامه و هر نوع فایل متنی دیگری میتوان از گیت استفاده نمود؛ مانند همین مقاله که با استفاده از گیت انجام شده است! پس با ما در ادامه این آموزش همراه باشید چون با همین اطلاعات و تمرین آن ها میتوانید آشنایی با گیت را در رزومه خود اضافه کنید.

۲-۱ تاریخچه

اینجا ی چیزایی راجبه تاریخش بنویس و سع یکن رفرنس بدی بجز اینجا جای دیگ رفرنس نداریم

۳-۱ کنترل منابع (Source control)

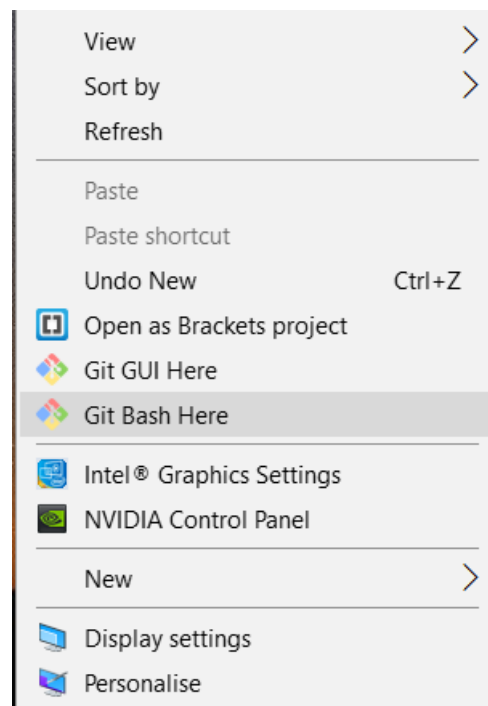
ی چیز در مورد سرس کنترل بنویس بازار و cvn و اینا رو هم بگو

۴-۱ گیت (Git)

در این پژوهش به کارایی گیت و همچنین نحوه اجرای دستورات مختلف آن آشنا خواهید شد.

۱-۴-۱ شروع کار با گیت

برای شروع کار با گیت باید نرم افزار گیت را در سیستم خود نصب نمایید. سپس با ساختن یک git dash در مسیری که میخواهید پروژه خود را شروع کنید صفحه ای باز میشود و میتوانیم در آن دستورات گیت را اجرا کنیم. برای شروع نیاز به دانستن دو نوع شروع پروژه یا ساخت مخزن داریم:



شکل ۱-۱: چگونگی ساختن گیت bash

init

دستور `git init` یک مخزن (repository) جدید ایجاد میکند. این دستور برای تبدیل یک پروژه موجود که ورژنش مشخص نشده به یک مخزن گیت یا ساختن مخزن جدید خالی مورد استفاده قرار میگیرد.

بقیه دستورات گیت خارج از محدوده ی یک مخزن گیت ساخته شده نیستند. پس بطور معمول این دستور اولین دستور است که شما آن را در یک پروژه جدید اجرا میکنید.

اجرای دستور `git init` یک زیرشاخه دستور `git` در شاخه ی جاری در حال استفاده میسازد که شامل تمامی متادیتاهایی است که گیت برای ساختن یم مخزن جدید به آن ها نیاز دارد. این متادیتا شامل زیرشاخه ها، مراجع ، قالب فایل ها است.

در کنار شاخه `git` در شاخه ریشه پروژه ، یک پروژه موجود باقی می ماند بدون تغییر میماند. به طور پیش فرض، دستور `init` پیکربندی `Git` را به مسیر زیرشاخه `git` راه اندازی خواهد کرد. مسیر دلخواه را می توانید تغییر دهید و سفارشی کنید اگر دوست دارید آن را در جای دیگر اسفاده کنید. شما می توانید متغیر محدوده `GIT-DIR` را به یک مسیر سفارشی تنظیم کنید و در ابتدا دستور `init` فایل های پیکربندی `Git` را در آنجا راه اندازی خواهد کرد. علاوه بر این شما می توانید یک آرگومان `separate-git-dir` برای نتیجه ی مشابه منتقل کنید. یک مورد استفاده معمول برای یک زیرگروه جداگانه `git`. این است که پیکربندی سیستم شما را در پوشه اصلی حفظ کند و پوشه `git` را در جای دیگر نگه دارد.

```
Comp Center@Keyhan MINGW64 /d/algorithm
$ git init
Initialized empty Git repository in D:/algorithm/.git/

Comp Center@Keyhan MINGW64 /d/algorithm (master)
$ |
```

شکل ۱-۲: اعمال دستور `init` بر روی یک شاخه دلخواه در محیط گیت `bash`

clone

این دستور برای دانلود یک ریپازیتوری موجود در شبکه استفاده میشود. در واقع ما میتوانیم با استفاده از این دستور یک فایل git. موجود در شبکه که میتواند سایت هایی مانند gitlab یا github و یا گیت شرکت شما باشد را در یکی از شاخه های سیستم خودمان داشته باشیم. مانند شکل ۱-۳ این دستور عینا پروژه را در مسیر منتخب شما کپی میکند، و شما میتوانید آن را تغییر داده و ادامه دهید.

گیت کلون تنها یک کپی کاری ساده نیست، بلکه با این دستور شما در واقع یک کپی از تمامی اطلاعاتی که سرور دارد است. تمامی ورژن ها، تغییرات اعمال شده بر آنها در طول زمان است. در حقیقت، اگر دیسک سرور شما خراب شود، شما اغلب می توانید تقریبا هر یک از کلون ها را بر روی هر سیستمی در شبکه بکار ببندید تا سرور را به حالت ای که در زمان کپی شدن آن بود، بازگردانید.

```
Comp Center@Keyhan MINGW64 /d/algorithm (master)
$ git clone https://github.com/elyasneo/Latex.git
Cloning into 'Latex'...
remote: Enumerating objects: 106, done.
remote: Counting objects: 100% (106/106), done.
remote: Compressing objects: 100% (70/70), done.
```

شکل ۱-۳: دانلود روند همین پروژه از گیت لب از پروژه مشترکان

۱-۴-۲ بررسی و دستیابی

یکی از محاسن گیت این است که ما میتوانیم در هر لحظه تغییرات ایجاد شده بر روی هر فایل را مشاهده کنیم! فایل های تغییر یافته را بشناسیم و در کل وضعیتی که داریم را مشاهده کنیم. همینطور میتوانیم به گذشته برویم و به ورژن های مختلف دسترسی داشته باشیم. این ویژگی بسیار در کار ما با گیت اهمیت دارد زیرا گاهی پیش می آید که فرضا در حین انجام یک بخش از کار نیاز به نصب نرم افزار بر یک سیستم یا ایجاد یک باگ شود که نیاز به دسترسی به نسخه های قبلی برای نصب و یا ایراد یابی باشد.

status

دستور گیت status وضعیت دایرکتوری کار و منطقه پیمایش را نمایش می دهد. این به شما اجازه می دهد ببینید کدام تغییرات مرتب شده اند و کدام فایل ها توسط Git ردیابی نمی شوند. خروجی این دستور شما هیچ اطلاعاتی را نشان نمی دهد برای این کار باید از گیت log استفاده

کنید. شکل های را ببینید: ۴-۱ در این تصویر فایل های قرمز ابتدایی فایل های تغییر یافته و فایل های قسمت دوم فایل های که توسط گیت ردیابی نمیشوند را نمایش میدهد.

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

        modified:   Chapters/chapter1.aux
        modified:   Chapters/chapter1.tex
        modified:   Figures/GitInit.PNG
        modified:   MainThesis.aux
        modified:   MainThesis.lof
        modified:   MainThesis.log
        modified:   MainThesis.out
        modified:   MainThesis.pdf
        modified:   MainThesis.synctex.gz
        modified:   MainThesis.toc

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

        Figures/GitClone.PNG

no changes added to commit (use "git add" and/or "git commit -a")
```

شکل ۴-۱: گیت استتوس بر روی روند مقاله فعلی بر شاخه git.

log

با اجرای این دستور لیستی از کامیت هایی که تاکنون بر روی شاخه انجام شده است مشاهده میشود، شکل را ببینید: ۱-۵

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git log
commit 629a2080e60c966f433ce588b47f652e21c16247 (HEAD -> master)
commit 629a2080e60c966f433ce588b47f652e21c16247 (HEAD -> master)
Author: keyhan <keyhan@gmail.com>
Date: Thu Jan 10 12:50:59 2019 +0330

    commit #31

commit 09fbe3ab9103e910656ee4ce7944ed5cd98eb621 (origin/master, origin/HEAD)
Author: keyhan <keyhan@gmail.com>
Date: Thu Jan 10 11:12:55 2019 +0330

    avale sob

commit ba0c87e94e2a0b50018f0454759ab57efa07209d
Author: keyhan <keyhan@gmail.com>
Date: Thu Jan 10 03:14:19 2019 +0330

    shabe nokhost

commit 55a4fc885106e0d3fea98edd6e5427b2014ed7e4
Author: Elyas <elyasneo@gmail.com>
Date: Thu Jan 10 00:24:03 2019 +0330

    m1

commit 8bbca46ee10965e069b5b8ffbf37fdbd000ad2c8
Author: Elyas <elyasneo@gmail.com>
Date: Thu Jan 10 00:22:30 2019 +0330

    m1
```

شکل ۱-۵: مشاهده کامیت های انجام شده بر روی مقاله کنونی تا بدینجای کار با دستور گیت log

checkout

دستور checkout عمل تغییر بین نسخه های مختلف یک موجودیت هدف است. این دستور بر روی سه موجودیت مجزا عمل می کند: فایل ها، commit ها، و شاخه ها. علاوه بر تعریف «checkout» عبارت «چک کردن» معمولاً به معنای اجرای دستور فرمان گیت checkout است. در موضوع لغو تغییرات، ما شاهد چگونگی استفاده از گیت Checkout برای مشاهده اعمال قدیمی بودیم. اکثر تمرکز این سند بر عملیات checkout در شاخه ها است.

چک کردن شاخه ها مشابه چک کردن قرارداد های قدیمی و فایل ها در آن است که دایرکتوری کاری برای مطابقت با شاخه انتخاب شده به روز می شود؛ با این حال، تغییرات جدید در تاریخ پروژه ذخیره می شود - یعنی یک عملیات فقط خواندنی نیست.

در شکل ۱-۶ ابتدا به یک نسخه قدیمی میرویم و بعد در شکل ۱-۷ با دستور بعدی به ورژن فعلی

برمیگردیم!

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git checkout 8bbca46ee10965e069b5b8ffbf37fdbd000ad2c8
Note: checking out '8bbca46ee10965e069b5b8ffbf37fdbd000ad2c8'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

    git checkout -b <new-branch-name>

HEAD is now at 8bbca46 m1
```

شکل ۱-۶: به ورژن های قبلی

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex ((8bbca46...))
$ git checkout master
Previous HEAD position was 8bbca46 m1
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 2 commits.
(use "git push" to publish your local commits)
```

شکل ۱-۷: برگشت به ورژن master

۱-۴-۳ اعمال تغییرات

پس از تغییر دادن هر فایل اگر از انجام آن تغییر مطمئن هستید میتوانید آن تغییرات را ذخیره کنید. در اینجا روند تغییر هر فایل را با هم یاد میگیریم:

add

پس از اعمال تغییر روی هر فایل برای ثبت کردن این تغییرات ابتدا باید آن فایل را در لبه یا همان stage قراردسیم! با این دستور فایل های تغییر یافته در آستانه ی ثبت تغییر قرار میگیرند.؟؟

reset

دستور ریست برای خالی کردن stage استفاده میشود. هنگامی که ما از یک کامیت پشیمان شدیم قبل از اعمالش میتوانیم با این دستور لبه را خالی کنیم.


```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git add -A
warning: LF will be replaced by CRLF in Chapters/chapter1.aux.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.aux.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.ind.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.lof.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.log.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.out.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in MainThesis.toc.
The file will have its original line endings in your working directory
```

شکل ۱-۸: مثالی از گیت add

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git reset
Unstaged changes after reset:
M      Chapters/chapter1.aux
M      Chapters/chapter1.tex
M      MainThesis.aux
M      MainThesis.lof
M      MainThesis.log
M      MainThesis.out
M      MainThesis.pdf
M      MainThesis.synctex.gz
M      MainThesis.toc
```

شکل ۱-۹: مثالی از گیت reset

commit

دستور commit محتویات فعلی فهرست لبه را در یک index جدید ذخیره می کند به همراه یک پیام توصیف تغییرات که توسط کاربر وارد میشود.

```
Comp Center@Keyhan MINGW64 ~/Desktop/latec project/Latex (master)
$ git commit -m "commit #31"
[master 629a208] commit #31
12 files changed, 65 insertions(+), 35 deletions(-)
create mode 100644 Figures/GitClone.PNG
rewrite Figures/GitInit.PNG (99%)
create mode 100644 Figures/GitStatus.PNG
rewrite MainThesis.synctex.gz (99%)
```

شکل ۱-۱۰: مثالی از گیت commit

۴-۴-۱ توزیع

push

pull

۵-۴-۱ گسترش

branch new

merge

مراجع

- [1] J. D. Blischak, E. R. Davenport, and G. Wilson, “A quick introduction to version control with Git and GitHub,” *PLoS computational biology*, vol. 12, no. 1, p. e1004668, 2016.
- [2] S. Chacon and B. Straub, *Pro git*. Apress, 2014.
- [3] B. De Alwis and J. Sillito, “Why are software projects moving from centralized to decentralized version control systems?,” in *Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, pp. 36–39, 2009.
- [4] K. Ram, “Git can facilitate greater reproducibility and increased transparency in science,” *Source code for biology and medicine*, vol. 8, no. 1, p. 7, 2013.
- [5] D. Spinellis, “Git,” *IEEE software*, no. 3, pp. 100–101, 2012.

Abstract

English abstract goes here.

Keywords: *git, version control*



Babol Noshirvani University of Technology
Faculty of Electrical and Computer Engineering

Dissertation Submitted in Partial Fulfillment of The Requirements
For The Degree of Bachelor of Science in Software Engineering

Version Control

by

Elyas Abbasi -Keyhan Rayati

Supervisor

Dr. Hamid Jazayeri

2019 Jan