Mudul 318 Dokumentation

Inhaltsverzeichnis

[Einleitung 3](#_Toc452455423)

[Planung 4](#_Toc452455424)

[Aktivitätsdiagramm 4](#_Toc452455425)

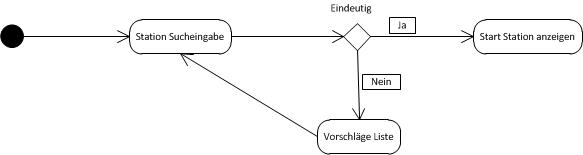
[UML-Klassendiagramm 4](#_Toc452455426)

# Einleitung

# Anforderungen

# Planung

## Aktivitätsdiagramm



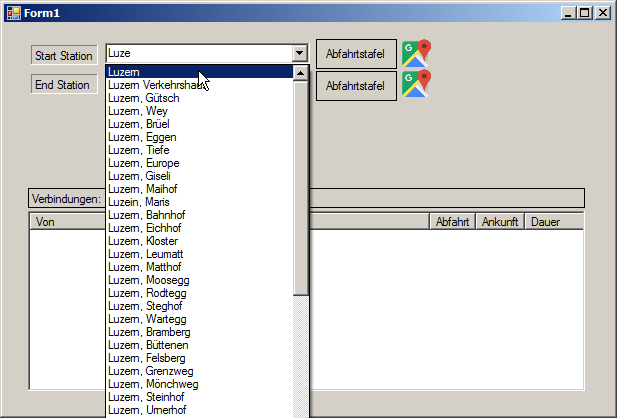
## UML-Klassendiagramm

# Programm

## Stationssuche

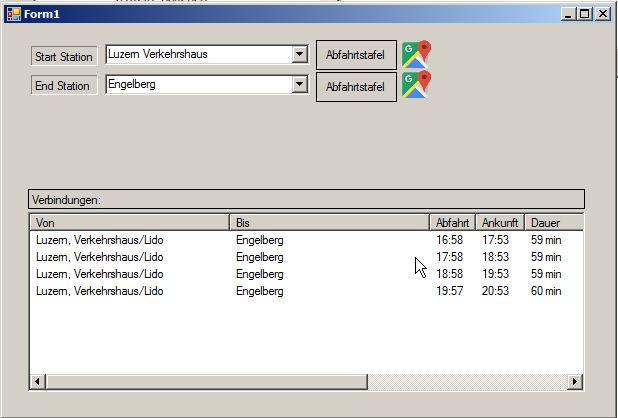
Während man tippt, wird eine DropDown-Liste mit passenden Vorschlägen angezeigt.

Man kann entweder direkt den passenden Vorschlag auswählen oder einfach fertig schreiben.



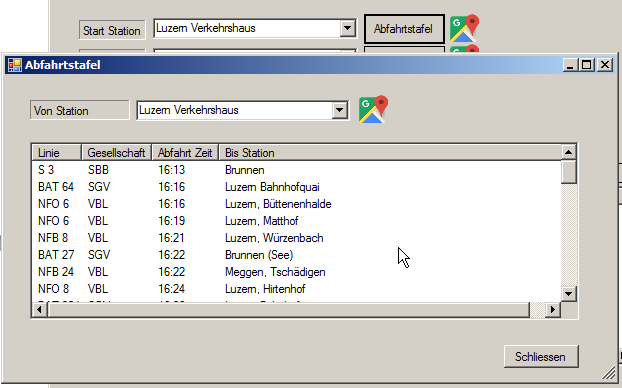
## Verbindungssuche

Sobald man beide Stationen eindeutig angegeben hat, werden passende Verbindungen angezeigt.



## Station Abfahrtstafel

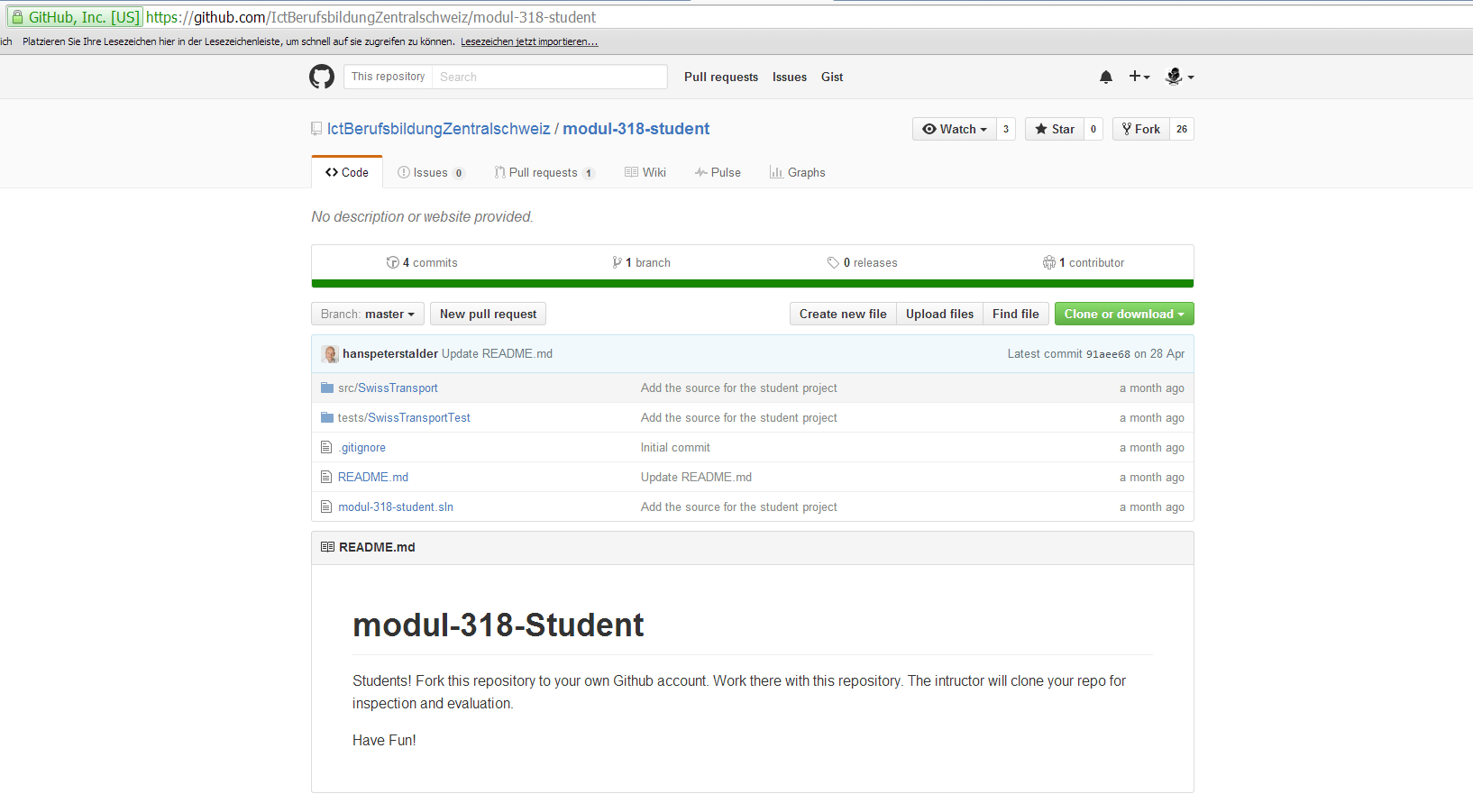
Wenn man zu einer Station verfügbare Verbindungen sehen will, kann man auf einen der Abfahrtstafel Buttons drücken. Wenn man die Station schon eindeutig angegeben hat, wird die Station zum Button automatisch übernommen. Man kann dann auch andere Stationen eingeben.



# Testen

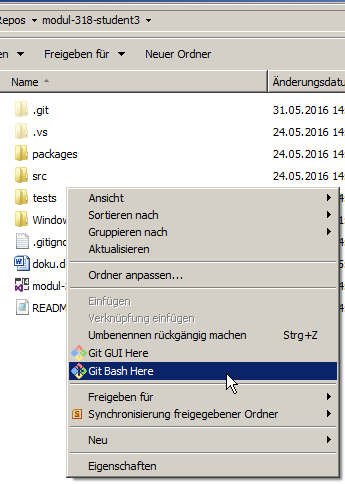
# GitHub

GitHub ist eine Seite, wo man Projekte hochladen, bearbeiten und für andere zugänglich machen kann.

Mit „Fork“ kann man sich eine eigene Version eines fremden Repositorys anlegen und diese dann unabhängig bearbeiten.

## GitBash

Die Git Bash ist eine eigne Konsole für GitHub. Mit der Git Bash kann man diverse git befehle ausführen um damit die Repositorys zu verwalten.



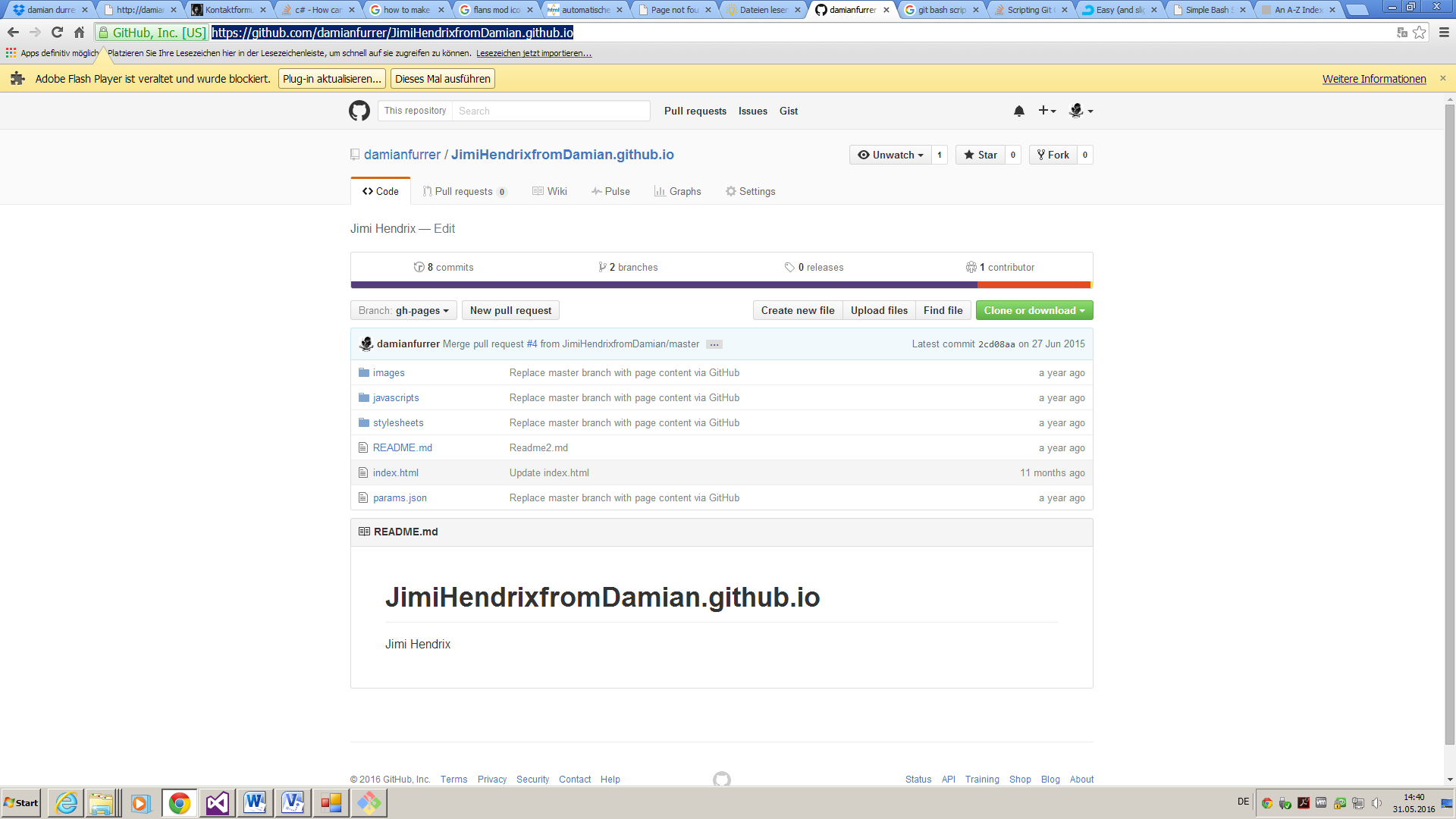
In einem lokalen Repository, oder einem Dafür vorgesehenen Ort, kann man mit „Git Bash Here“

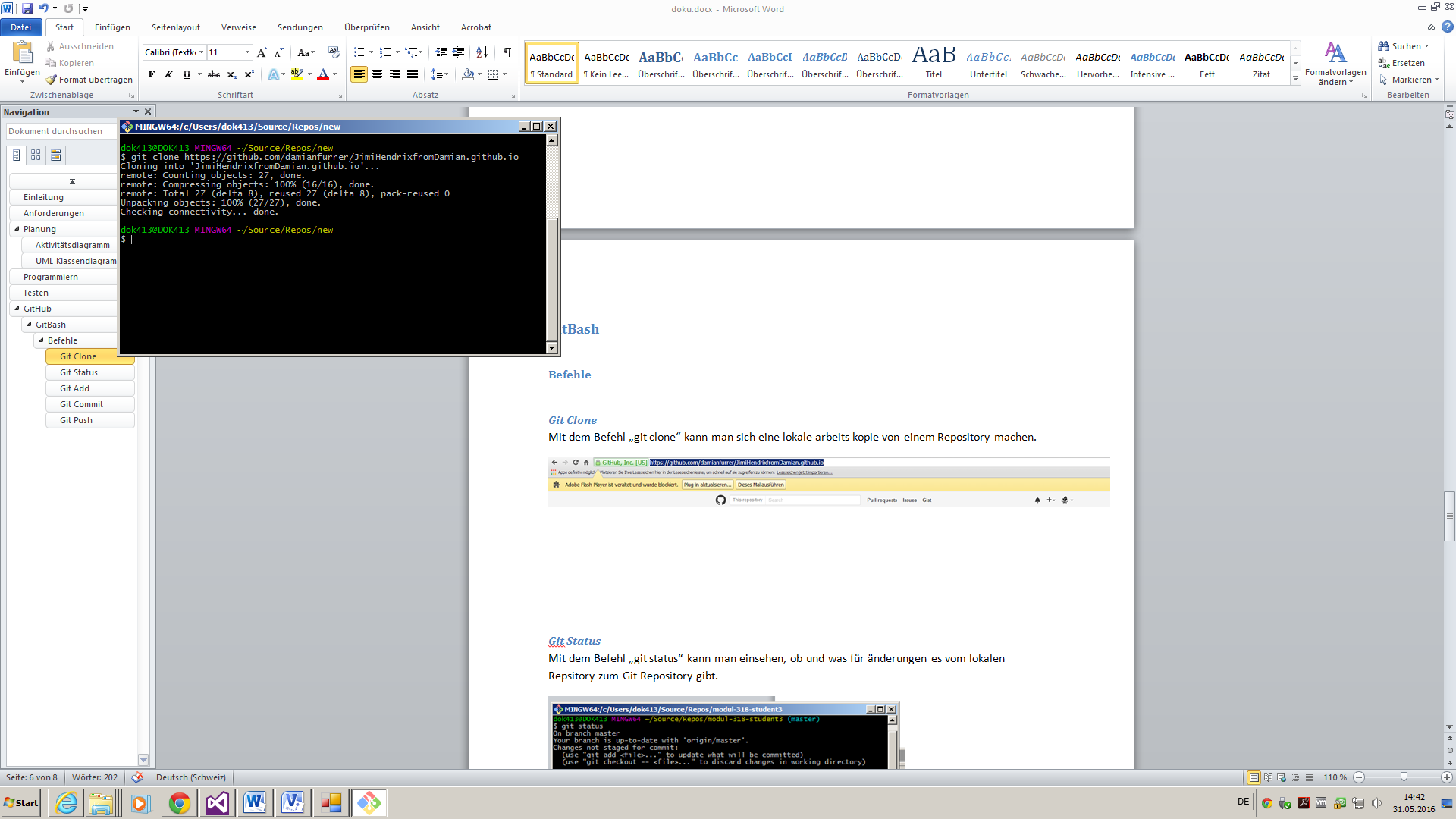
Die Git Bash für diesen Pfad öffnen.

### Befehle

#### Git Clone

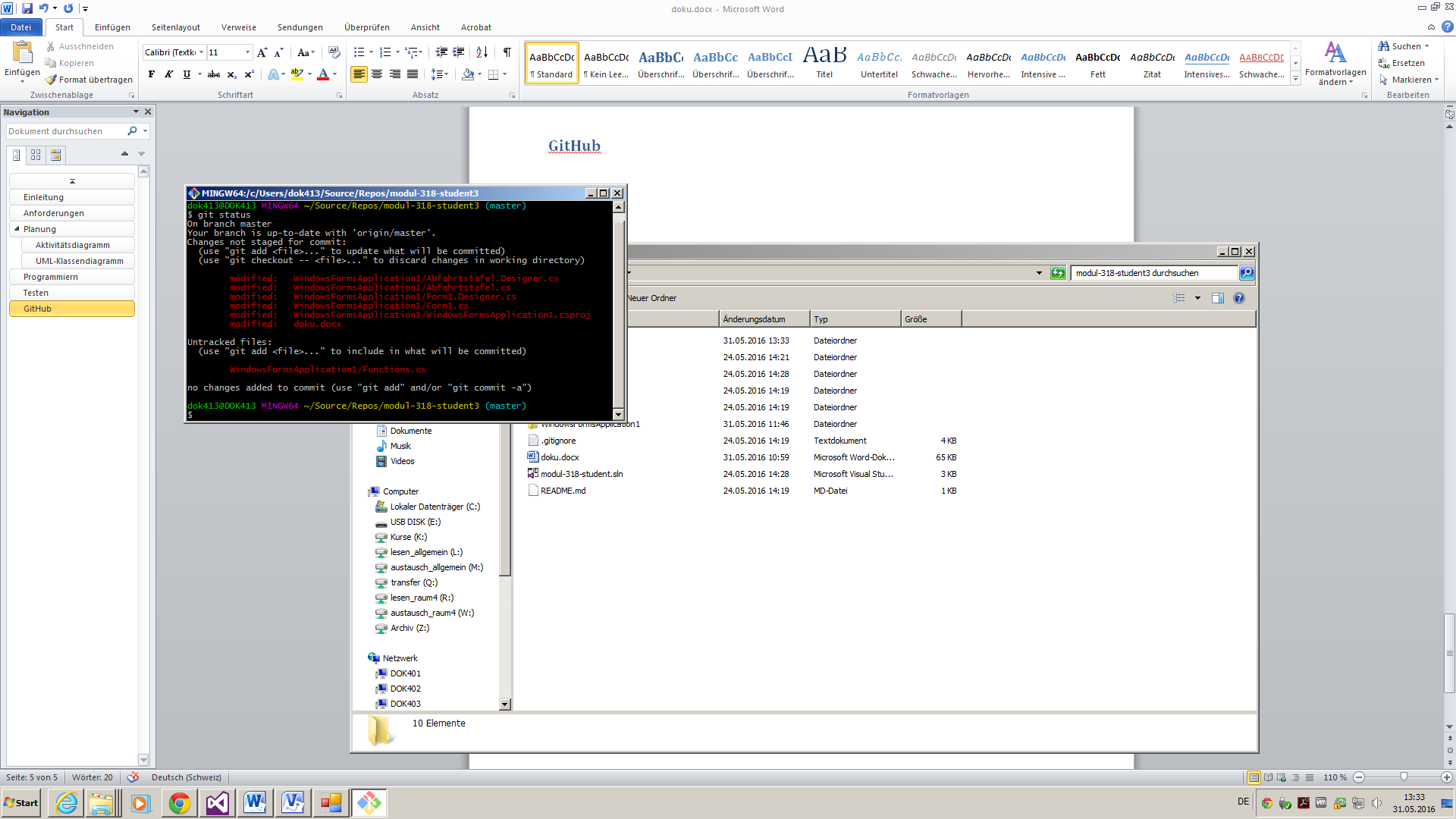
Mit dem Befehl „git clone“ kann man sich eine lokale arbeits kopie von einem Repository machen.



Mit „git clone“ und der Adresse des gewünschten Repositorys, welche man sich natürlich kopieren kann, kann man sich das Repository in den lokalen Ordner, wo man die Git Bash öffnete, herunterladen

#### Git Status

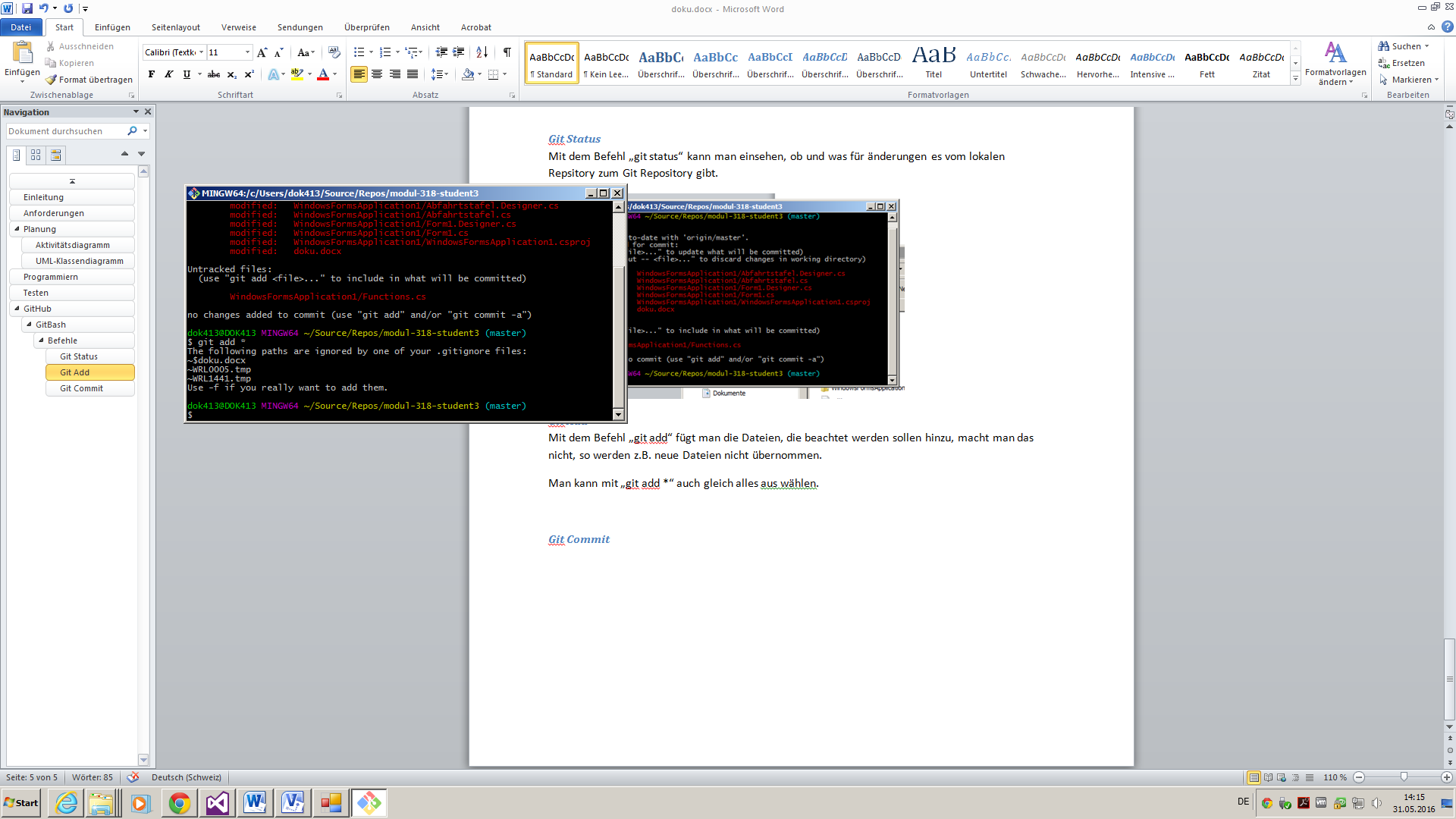
Mit dem Befehl „git status“ kann man einsehen, ob und was für änderungen es vom lokalen Repsitory zum Git Repository gibt.



#### Git Add

Mit dem Befehl „git add“ fügt man die Dateien, die beachtet werden sollen hinzu, macht man das nicht, so werden z.B. neue Dateien nicht übernommen.

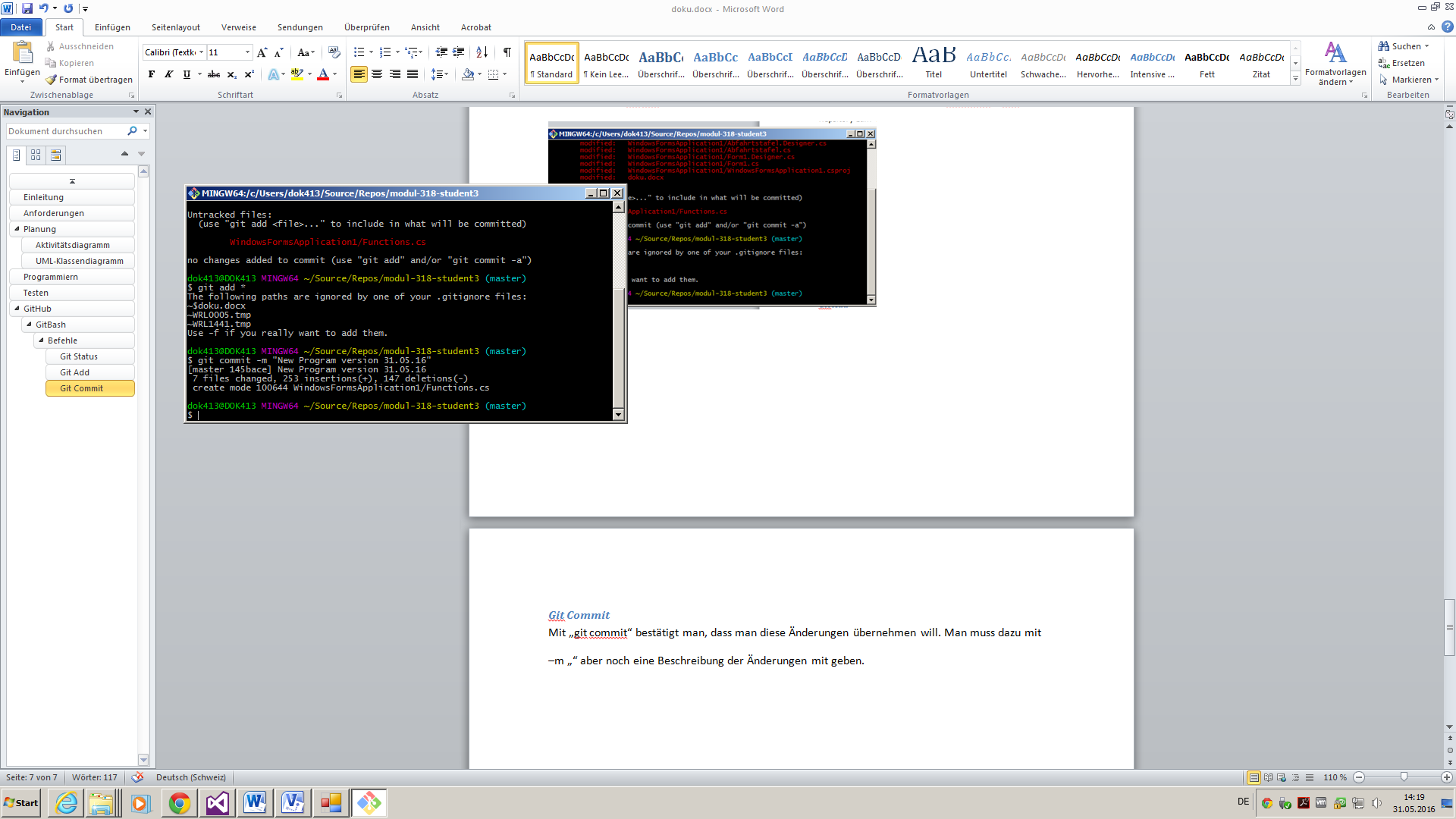
Man kann mit „git add \*“ auch gleich alles auswählen, sofern es nicht in einem .gitignore – file steht.



#### Git Commit

Mit „git commit“ bestätigt man, dass man diese Änderungen übernehmen will. Man muss dazu mit

–m „“ aber noch eine Beschreibung der Änderungen mit geben.

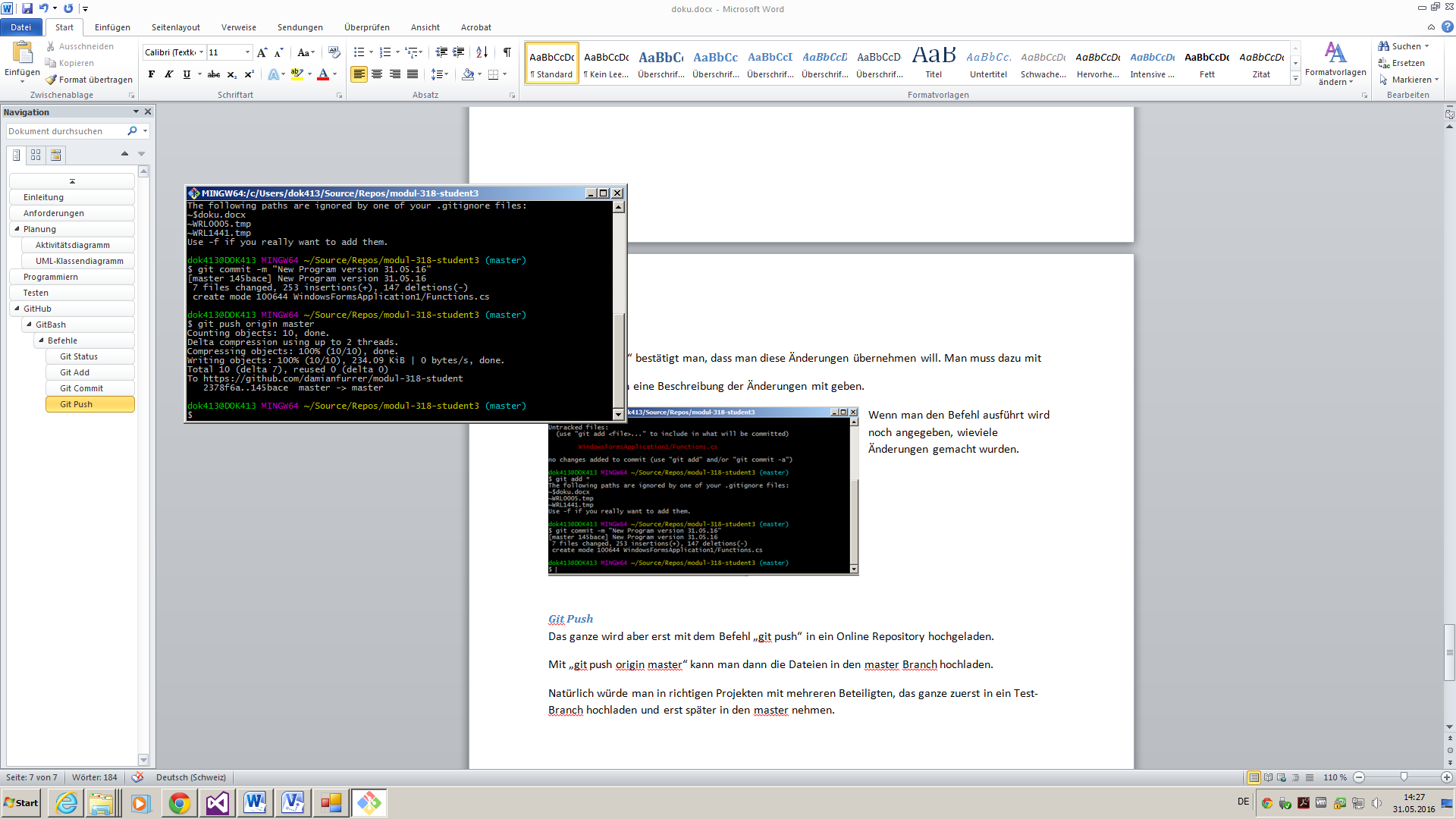
Wenn man den Befehl ausführt wird noch angegeben, wieviele Änderungen gemacht wurden.

#### Git Push

Das ganze wird aber erst mit dem Befehl „git push“ in ein Online Repository hochgeladen.

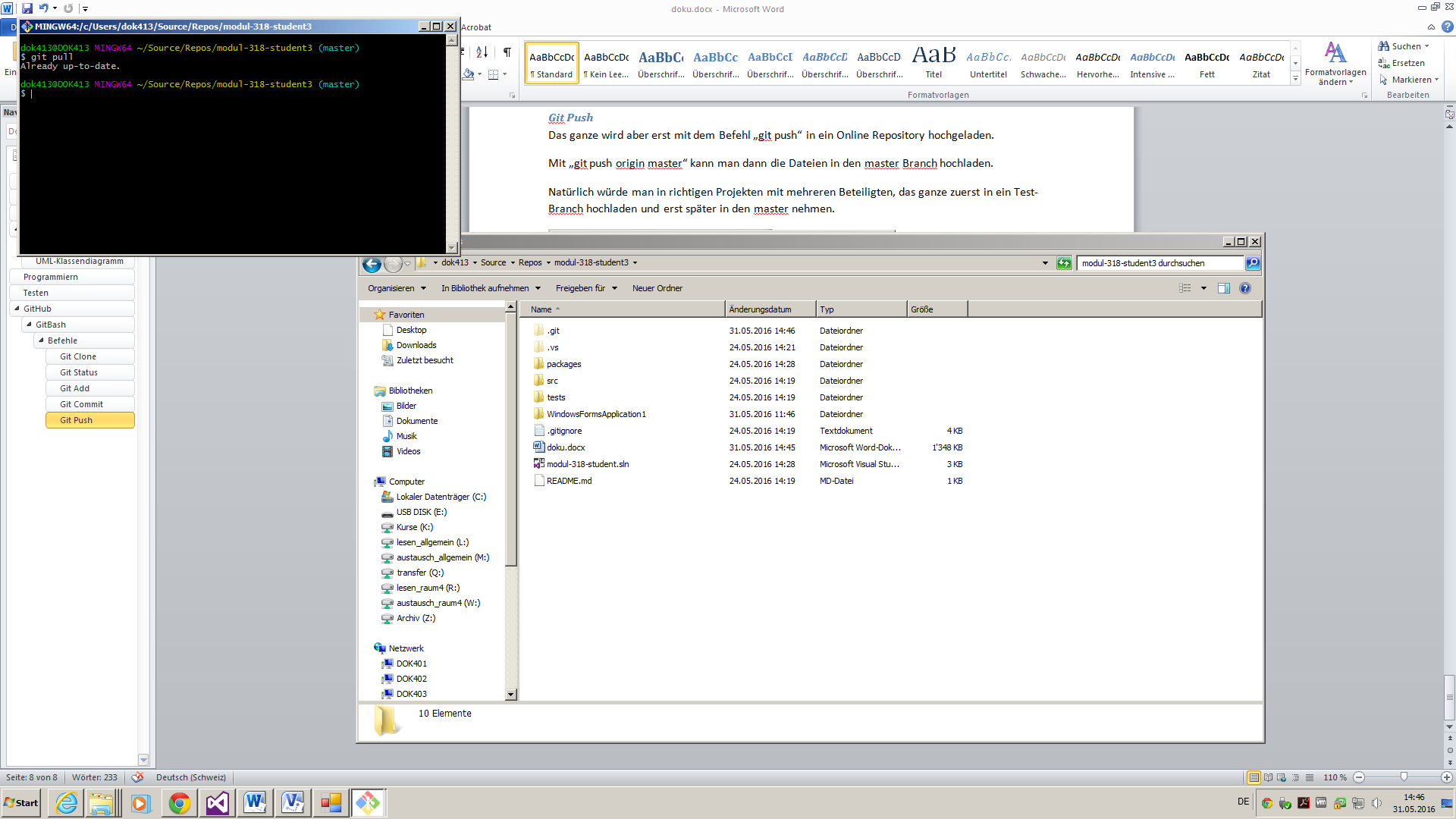
Mit „git push origin master“ kann man dann die Dateien in den master Branch hochladen.

Natürlich würde man in richtigen Projekten mit mehreren Beteiligten, das ganze zuerst in ein Test-Branch hochladen und erst später in den master nehmen.



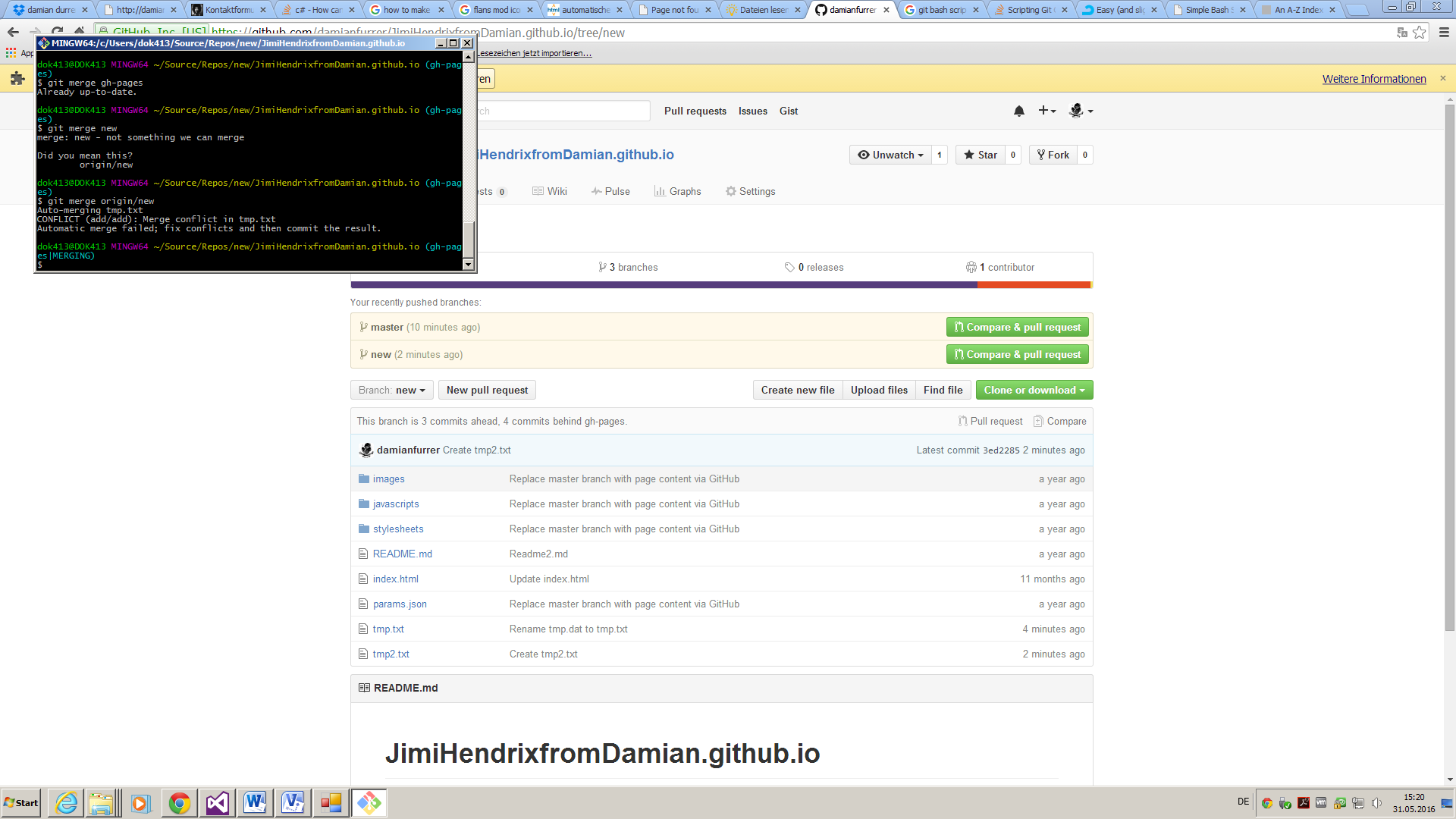
#### Git Pull

Mit „git pull“ kann man sein lokales Repository aktualisieren.

Ist es aktuell, so wird nur ausgegeben, dass es schon aktuell ist.

#### Git Merge

Mit „git merge“ und dem Branch name kann man den angegebenen Branch mit dem aktuellen mergen.

So mergt man z.B. den „new“ stand in den aktuellen „gh-pages“ stand:

Dabei kann es natürlich konflikte geben, die man von hand lösen muss und dann mit „git add“ wieder hinzu fügen muss.

#### Git Diff

Mit „git dif“ kann man dann die Unterschiede zwischen der Quell-Branch und der Ziel-Branch ausgeben.

