

# ScheldeMonitor Handleiding: GitHub repository - T2021

Geschreven, onderhouden en geüpdatet door het ScheldeMonitor team van VLIZ - Versie 3  
(24/08/2021)

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>over</b>   | <b>1</b>  |
| <b>2</b> | <b>Toegang tot de T2021 GitHub repository</b>         | <b>2</b>  |
| 2.1      | Toegang aanvragen . . . . .                           | 2         |
| 2.2      | Toegangsbeheer . . . . .                              | 3         |
| <b>3</b> | <b>Werken met een GitHub repository</b>               | <b>5</b>  |
| 3.1      | Installatie van Git . . . . .                         | 5         |
| 3.2      | Versiebeheer gebruiken met GitHub & RStudio . . . . . | 6         |
| 3.3      | Regels voor GitHub repository van T2021 . . . . .     | 13        |
| <b>4</b> | <b>Helpdesk</b>                                       | <b>18</b> |

## 1 over

Om het functioneren en de activiteiten die plaats vinden in het Schelde estuarium te evalueren, hebben Vlaanderen en Nederland samengewerkt om een methodiek te ontwikkelen die gebruikt maakt van indicatoren om de staat van het estuarium te beoordelen. Deze evaluatie wordt om de zes jaar uitgevoerd in samenwerking met verschillende partners en leden van de projectgroep 'Evaluatie en Rapportage' (PG ER) van de Vlaams-Nederlandse Schelde Commissie (VNSC). Een derde deel van deze evaluatie is in voorbereiding voor T2021, die gebaseerd is op de T2015 evaluatie en de T2009 baseline.

Ervaringen in eerdere rapporten hebben uitgewezen dat een centraal beheer van bestanden en scripts, en het correct toepassen van versiebeheer, van groot belang zijn om de leesbaarheid, duidelijkheid en reproduceerbaarheid van de uitgevoerde evaluatie te verzekeren. Daarom heeft het ScheldeMonitor data- en informatieportaal de opdracht gekregen om een GitHub-repository op te stellen voor alle analyses uitgevoerd voor het T2021-rapport. Deze repository is beschikbaar in de ScheldeMonitor GitHub organisatie.

Deze handleiding werd gemaakt om ervoor te zorgen dat het gebruik van deze repository uniform en consistent is. Om dit te bereiken bevat de handleiding richtlijnen en regels over de werkwijze, inhoud management en structuur.

Aangezien het informatie- en dataportaal van ScheldeMonitor een RStudio omgeving bevat die bedoeld is om te worden gebruikt in combinatie met deze GitHub repository, is deze handleiding voornamelijk voor R gebruikers opgesteld. Meerdere richtlijnen kunnen echter ook in andere programmeertalen worden gebruikt. Voor sommige secties is de inhoud gebaseerd op de RStudio handleiding die beschikbaar is op de ScheldeMonitor website. De belangrijkste secties worden ook aangeduid in het 'ReadMe' bestand van de repository.

## 2 Toegang tot de T2021 GitHub repository

### 2.1 Toegang aanvragen

GitHub is een code-hosting platform voor versiebeheer en samenwerking dat het gemakkelijkst toegankelijk is via een webbrowser op <https://github.com/>. Hiermee kunnen gebruikers vanaf elke locatie samenwerken aan projecten. GitHub projecten worden opgeslagen als 'repositories', die ofwel publiekelijk ofwel privaat zichtbaar zijn.

De T2021 private GitHub is een private repository binnen de 'ScheldeMonitor' GitHub organisatie. Dit betekent dat de organisatie zichtbaar is in GitHub, maar de repository toch verborgen is voor niet-leden.

The screenshot shows the GitHub profile page for the organization 'scheldemonitor'. The top navigation bar includes the GitHub logo, the organization name 'scheldemonitor', and links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation bar is a summary table of repository statistics:

|              |          |
|--------------|----------|
| Repositories | 0        |
| Code         | 25       |
| Commits      | 0        |
| Issues       | 0        |
| Discussions  | 0 (Beta) |
| Packages     | 0        |
| Marketplace  | 0        |
| Topics       | 0        |
| Wikis        | 0        |
| Users        | 1        |

To the right of the table, it says '1 user' and shows the organization's profile: 'ScheldeMonitor scheldemonitor' with the email 'info@scheldemonitor.org'. Below this is a navigation bar with 'Repositories', 'Packages', 'People', and 'Projects'. The main content area shows a message: 'This organization has no public repositories.' To the right, there is a 'People' section with a message: 'This organization has no public members. You must be a member to see who's a part of this organization.' At the bottom, there is a footer with the GitHub logo, copyright information, and various links: 'Terms', 'Privacy', 'Cookie Preferences', 'Security', 'Status', 'Help', 'Contact GitHub', 'Pricing', 'API', 'Training', 'Blog', and 'About'.

Onderzoekers, leden van VNSC of hun werkgroepen, en partner instituten van ScheldeMonitor kunnen toegang aanvragen voor de ScheldeMonitor GitHub organisatie, of voor één specifieke openbare of private repository.

Om dit te doen kunnen gebruikers een mail sturen naar [info@scheldemonitor.org](mailto:info@scheldemonitor.org) met als onderwerp 'ScheldeMonitor GitHub', en de antwoorden op volgende vragen:

- Naam en GitHub gebruikersnaam?
- Bent u lid van VNSC of van één van de werkgroepen?
- Wilt u toegang tot specifieke repositories, en waarom?
- Met welk instituut bent u verbonden?
- Heeft u een specifiek niveau van toestemming nodig (zie Regels voor de repository van T2021)?

## 2.2 Toegangsbeheer

Met de gegeven informatie zal de helpdesk van ScheldeMonitor controleren als er toegang kan gegeven worden voor de gebruiker of niet. Dit betekent dat de gebruiker twee verschillende 'rollen' krijgt toegewezen, één binnen de 'ScheldeMonitor' GitHub organisatie en één binnen een gekozen repository, die elk een andere set rechten toekennen.

### 2.2.1 Toegang tot de organisatie

Wanneer u in aanmerking komt voor een lidmaatschap van de ScheldeMonitor GitHub organisatie, kan de gebruiker een 'eigenaar' of 'lid' status toegewezen krijgen. Indien niet wordt de gebruiker beschouwd als een 'externe medewerker' en zal enkel toegang krijgen tot de gevraagde repositories binnen de organisatie. De helpdesk van ScheldeMonitor en de VNSC worden standaard als eigenaars van de organisatie beschouwd. Het verschil tussen de eigenaar, lid of externe medewerker status wordt gegeven in de onderstaande tabellen:

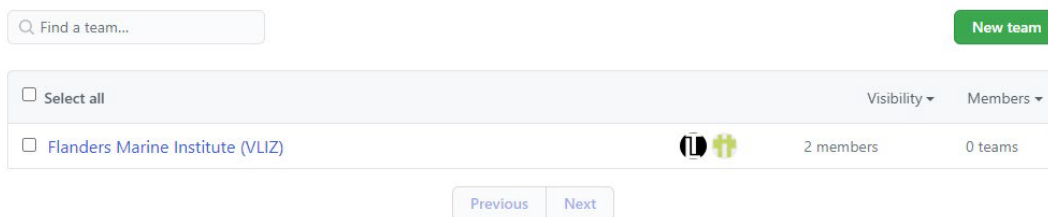
| <b>Visibility</b>                   | Owner | Member | Outside collaborator |
|-------------------------------------|-------|--------|----------------------|
| <b>See organization</b>             | X     | X      | X                    |
| <b>See public repositories</b>      | X     | X      | X                    |
| <b>See private repositories</b>     | X     | X      |                      |
| <b>See members</b>                  | X     | X      | X                    |
| <b>Change repository visibility</b> | X     |        |                      |

| <b>Management</b>                | Owner | Member | Outside collaborator |
|----------------------------------|-------|--------|----------------------|
| <b>Manage members</b>            | X     |        |                      |
| <b>Manage collaborators</b>      | X     |        |                      |
| <b>Create public repository</b>  | X     |        |                      |
| <b>Create private repository</b> | X     |        |                      |
| <b>Delete repositories</b>       | X     |        |                      |
| <b>Transfer repositories</b>     | X     |        |                      |

| <b>Base repository rights</b>      | Owner | Member | Outside collaborator |
|------------------------------------|-------|--------|----------------------|
| <b>Clone any repository</b>        | X     | X      |                      |
| <b>Pull any repository</b>         | X     | X      |                      |
| <b>Push/merge any repository</b>   | X     |        |                      |
| <b>Branch public repositories</b>  | X     | X      |                      |
| <b>Branch private repositories</b> | X     |        |                      |

| Interaction                | Owner | Member | Outside collaborator |
|----------------------------|-------|--------|----------------------|
| Create discussions         | X     | X      |                      |
| Comment discussions        | X     | X      |                      |
| Delete issues              | X     | X      |                      |
| Create teams               | X     | X      |                      |
| Manage teams               | X     | X      |                      |
| Publish GitHub Pages sites | X     | X      |                      |

Leden kunnen toegevoegd worden aan een bepaald team binnen de organisatie. Het gebruik van teams maakt interne discussies, teamvermeldingen en opdrachten mogelijk, als ook het toevoegen van volledige teams aan bepaalde repositories in plaats van individuele leden en medewerkers. Deze teams kunnen per partner instituut of ten gunste van bepaalde werkgroepen van VNSC worden opgericht.



## 2.2.2 Toegang tot de repository

Hoewel organisatieleden basisleesrechten hebben, is het nog steeds nodig om gebruikers aan repositories toe te wijzen als medewerker om repository specifieke rechten te verlenen. Deze rechten hebben voorrang op de basisrechten die op organisatieniveau zijn ingesteld, wat betekent dat externe medewerkers lees- en schrijfrechten kunnen krijgen voor een specifieke repository zonder lid te moeten worden van de organisatie. De verschillende rollen voor een repository, en hun specifieke rechten worden weergegeven in de tabel hieronder:

|                                     | Read | Triage | Write | Maintain | Admin |
|-------------------------------------|------|--------|-------|----------|-------|
| <b>Pull repository</b>              | X    | X      | X     | X        | X     |
| <b>Branch repository</b>            | X    | X      | X     | X        | X     |
| <b>Clone repository</b>             | X    | X      | X     | X        | X     |
| <b>Comment</b>                      | X    | X      | X     | X        | X     |
| <b>Manage issues</b>                | X    | X      | X     | X        | X     |
| <b>Apply labels</b>                 |      | X      | X     | X        | X     |
| <b>Manage labels</b>                |      |        | X     | X        | X     |
| <b>Push/merge repository</b>        |      |        | X     | X        | X     |
| <b>Manage comments</b>              |      |        | X     | X        | X     |
| <b>Change repository visibility</b> |      |        |       |          | X     |
| <b>Change repository settings</b>   |      |        |       |          | X     |
| <b>Manage collaborators</b>         |      |        |       |          | X     |

De medewerkers krijgen standaard een "Write" status om te kunnen werken aan nieuwe ontwikkelingen binnen de repository. De ScheldeMonitor helpdesk alsook de projectleiders zullen de "Admin" status krijgen om de bijdragen aan de repository te kunnen beheren.

## 3 Werken met een GitHub repository

De T2021 repository kan gezien worden als een 'master folder', waar alles wat met dit specifieke project te maken heeft, moet worden bewaard. Repositories, of 'repo's', kunnen mappen bevatten, of uit aparte bestanden bestaan.

Het hebben van een GitHub repo maakt het voor gebruikers gemakkelijk om samenwerkings- en persoonlijke projecten bij te houden, aangezien alle bestanden die nodig zijn voor bepaalde analyses bij elkaar kunnen worden gehouden en mensen hun code, grafieken, enz. kunnen toevoegen terwijl de projecten zich ontwikkelen. Elk bestand op GitHub heeft een geschiedenis, waardoor het gemakkelijk is om wijzigingen te bekijken die er op verschillende tijdstippen in zijn aangebracht. U kunt de code van anderen bekijken, opmerkingen toevoegen aan bepaalde regels of het algemene document, en wijzigingen voorstellen.

Voor samenwerkingsprojecten kunt u met GitHub taken toewijzen aan verschillende gebruikers, zodat het duidelijk wordt wie verantwoordelijk is voor welk deel van de analyse. U kunt ook aan bepaalde gebruikers vragen om uw code te controleren. Voor persoonlijke projecten kunt u met versiebeheer uw werk bijhouden en gemakkelijk navigeren tussen de vele versies van bestanden die u gemaakt hebt, terwijl u ook een online back-up behoudt.

Dit hoofdstuk beschrijft hoe u de GitHub repository kunt gebruiken om alle scripts, documenten en databestanden te verzamelen die worden gebruikt om het T2021 rapport van het Schelde estuarium samen te stellen.

### 3.1 Installatie van Git

Eerst is een installatie van Git nodig op de persoonlijke hardware van de gebruiker om functies te verlenen die nodig zijn voor versiebeheer. Deze installatie verschilt tussen Windows en Mac hardware.

#### 3.1.1 Windows

Indien de gebruiker Windows hardware gebruikt, download en installeer Git voor uw besturingssysteem. Hieronder zijn enkele aanbevolen installatie instructies:

*For "Select Components", check:*

- *"Git Bash Here"*
- *"Git GUI Here"*
- *"Git LFS (Large File Support)"*
- *"Associate .git\* . . ."*
- *"Associate .sh . . ."*

*When prompted to choose the default editor, select Nano (a simple terminal editor) or Notepad++ (a simple graphical editor):*

*For "Adjust your PATH environment", select: "Use Git from Git Bash only"*

*For "Choose HTTPS transport backend", select: "Use the OpenSSL library"*

*For "Configure the line ending conversions", select: "Checkout Windows-style, . . ."*

*For "Configure the terminal emulator . . .", select: "Use MinTTY . . ."*

*For "Configure extra options", select: "Enable file system caching"*

*"Enable Git Credential Manager"*

#### 3.1.2 Mac

Als een gebruiker Mac software gebruikt, installeer Git dan via Homebrew, een pakketbeheerder voor command-line-programma's op Mac. Open eerst een terminal, die gevonden kan worden op ~/Application/Utilities/Terminal.app. Kopieer en plak daarna deze lijn in de terminal en druk op "Enter":

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Voeg volgende code in om Git te installeren:

```
brew install git
```

Volg alle instructies in het terminal venster, het kan zijn dat de gebruiker het wachtwoord van een Mac moet invoeren of akkoord moet gaan met vragen door “yes” te typen.

Nu Git geïnstalleerd is, kan de gebruiker versiebeheer gaan gebruiken voor zowel de interne projecten als met GitHub repositories.

## 3.2 Versiebeheer gebruiken met GitHub & RStudio

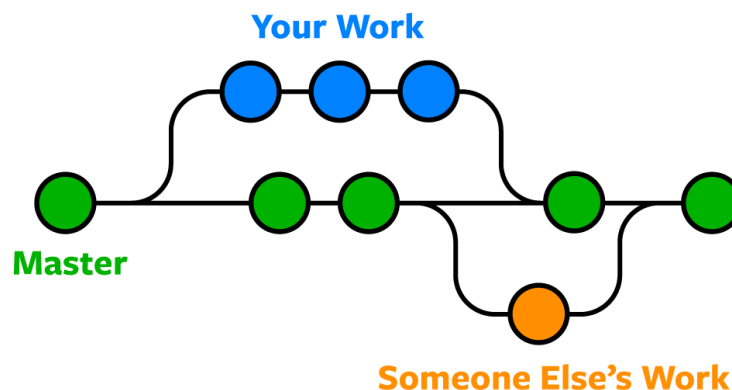
De GitHub werkwijze kan samengevat worden door de “pull-commit-push” mantra. Met deze methodologie heeft elk bestand op GitHub een geschiedenis. Dus in plaats van veel bestanden te hebben zoals `scripts_1st_May.R`, `script_2nd_May.R`, kan de gebruiker er maar één hebben. Door de geschiedenis ervan te verkennen, kan de gebruiker zien hoe het er op verschillende tijdstippen uitzag.

Omdat het de bedoeling is om de T2021 repository te gebruiken in combinatie met de RStudio omgeving van ScheldeMonitor, is het belangrijk om verbinding te maken tussen de repository en RStudio. Hoe u dit moet doen wordt in detail beschreven in de RStudio handleiding, beschikbaar in de GitHub repository of op de ScheldeMonitor website.

Eenmaal verbonden kan al het werk in de RStudio omgeving versie gecontroleerd worden uitgevoerd. De werkwijze om dit te doen wordt beschreven door de vijf stappen hieronder:

### 3.2.1 Een nieuwe projectbranch aanmaken

Het werken aan projecten gebeurt in ‘branches’. Een branch is een kopie, of versie van de project repository waarin wijzigingen kunnen worden aangebracht. Elk project start met de standaard ‘master’ branch. Deze branch moet door het hele project als rode draad worden behandeld en eindigen in het eindproduct. Het aantal rechtstreekse aanpassingen aan de masterbranch moet daarom gelimiteerd worden. Om nieuwe ontwikkelingen te maken, kan een nieuwe branch aangemaakt worden, die zich op die specifieke ontwikkelingen richten. Door deze methodologie te gebruiken, kunnen verschillende ontwikkelingen naast elkaar gestart worden, zonder dat dit invloed heeft op elkaar of de standaard master branch:



Een nieuwe branch maken kan het gemakkelijkst gedaan worden in het GitHub project repository online. Als u op de project webpagina bent, wordt de huidige gevisualiseerde branch boven de repository aangeduid:

master citation-impact / +

History Find file Web IDE 860b6cc3 Clone

Create citation\_searcher\_google\_scholar.py  
cedricdecruw authored 5 months ago

| Name                                | Last commit                                  | Last update  |
|-------------------------------------|--|--------------|
| relative-citation-index             | Folder for the Relative Citation Index added | 1 year ago   |
| .gitignore                          | first push                                   | 1 year ago   |
| Lijst_PP.csv                        | first push                                   | 1 year ago   |
| README.md                           | Initial commit                               | 1 year ago   |
| app.R                               | Update 28-2-20                               | 9 months ago |
| citation-impact.Rproj               | first push                                   | 1 year ago   |
| citation_searcher_google_scholar.py | Create citation_searcher_google_scholar.py   | 5 months ago |

README.md

De 'master' branch wordt standaard gekozen. Verschillende branches kunnen gekozen worden via het dropdown menu. Om een nieuwe branch aan te maken, kan het '+' icoontje gebruikt worden om de 'New Branch' optie voor de repository te kiezen:

master citation-impact / +

History Find file Web IDE 860b6cc3 Clone

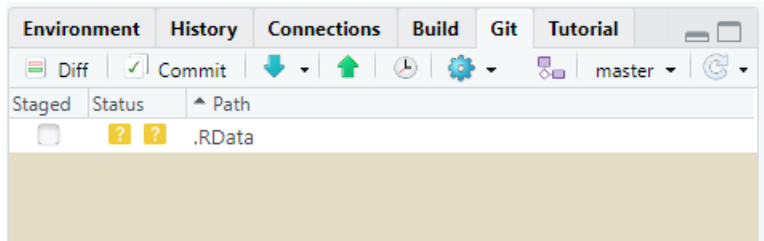
Create citation\_searcher\_google\_scholar.py  
cedricdecruw authored 5 months ago

- This directory
- New file
- Upload file
- New directory
- This repository
- New branch
- New tag

| Name                                | Last commit                                  | Last update  |
|-------------------------------------|--|--------------|
| relative-citation-index             | Folder for the Relative Citation Index added | 1 year ago   |
| .gitignore                          | first push                                   | 1 year ago   |
| Lijst_PP.csv                        | first push                                   | 1 year ago   |
| README.md                           | Initial commit                               | 1 year ago   |
| app.R                               | Update 28-2-20                               | 9 months ago |
| citation-impact.Rproj               | first push                                   | 1 year ago   |
| citation_searcher_google_scholar.py | Create citation_searcher_google_scholar.py   | 5 months ago |

Het systeem zal vragen om de branch een naam te geven en te kiezen uit welke bestaande branch een nieuwe branch moet worden aangemaakt (e.g. welk bestaande branch gekopieerd moet worden). Het is heel belangrijk om de branch duidelijk te benoemen zodat andere gebruikers een duidelijk idee hebben waarvoor de branch werd aangemaakt. Dit wordt verder uitgelegd in deze handleiding, in de sectie 'Branch nomenclatuur'.

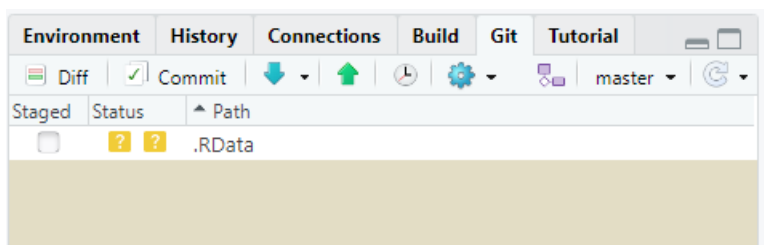
Wanneer de gebruikte RStudio instance is verbonden met GitHub (zoals uitgelegd in de ScheldeMonitor RStudio Handleiding), verschijnt er een 'Git' tabblad in het omgevingsframe van de RStudio workspace waar de gebruiker kan kiezen in welke branch wijzigingen worden gemaakt:



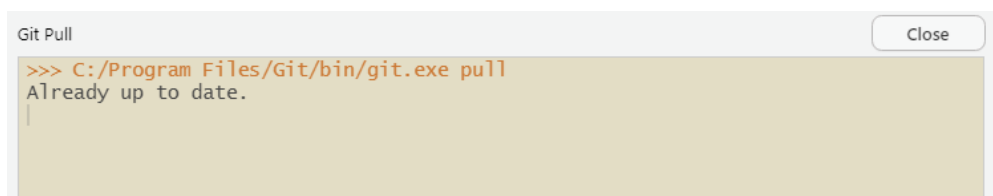
Zodra een nieuwe branch aangemaakt is of de juiste branch geselecteerd is, kan u beginnen met het bewerken van bestanden en scripts. Terwijl u dit doet, kan het versiebeheer worden beheerd in hetzelfde venster als hierboven weergegeven, met behulp van de stappen die hieronder worden besproken.

### 3.2.2 Pull een repository naar RStudio

Voordat er met werk kan begonnen worden op de repository, moet de gebruiker er zeker van zijn dat de laatste versie van de gekozen branch beschikbaar is. Om deze te verkrijgen, moet de branch gedownload of 'pulled' worden vanuit de online repository. Deze optie wordt gegeven in het 'Git' tabblad in het omgevingspaneel van de workspace, en is gemarkeerd met een blauwe pijl:



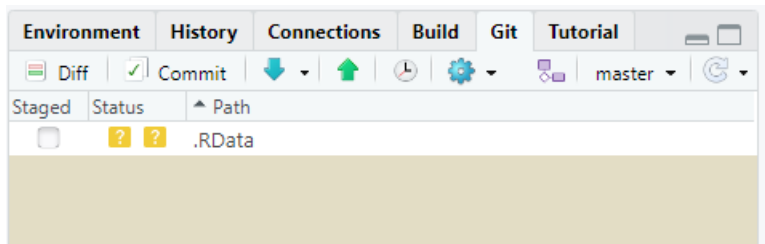
Door op het icoon te klikken wordt een 'pull' gestart van de aangegeven branch en verschijnt er een nieuw venster waarin de voortgang van de actie wordt aangegeven. Deze actie kan uitgevoerd worden wanneer de gebruiker dat wilt. Als de workspace al up-to-date is, wordt dit aangegeven in het voortgangsvenster:



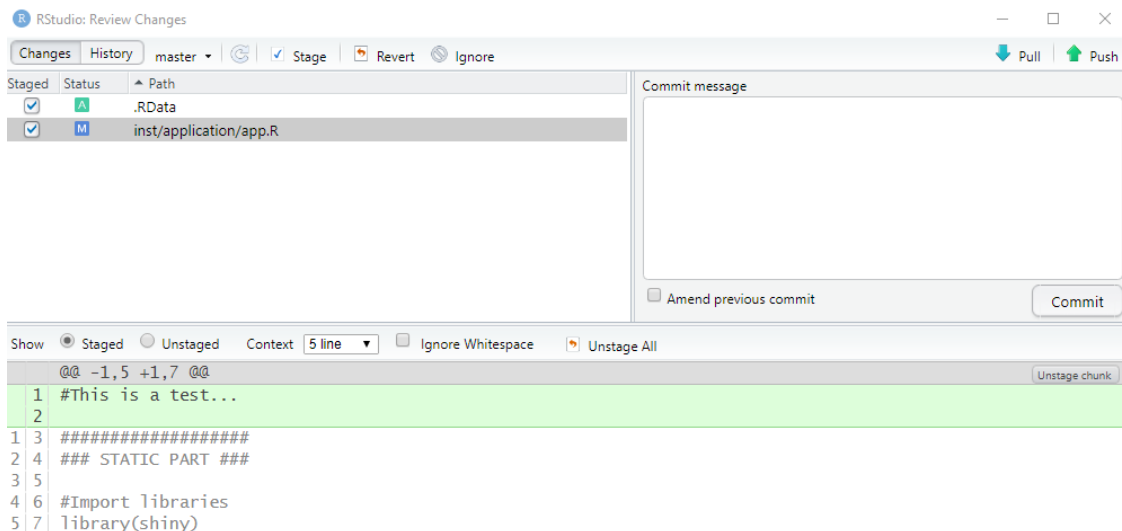
### 3.2.3 Wijzigingen aanbrengen en vastleggen

Na een pull kan de gebruiker beginnen met wijzigingen aan te brengen aan de bestanden van een gegeven branch van de repository. Eenmaal de wijzigingen zijn aangebracht, moeten de wijzigingen worden opgeslagen of 'vastgelegd' in de lokale versie van de branch. Om dit te doen kan de gebruiker de 'commit' knop in het 'Git' tabblad selecteren.



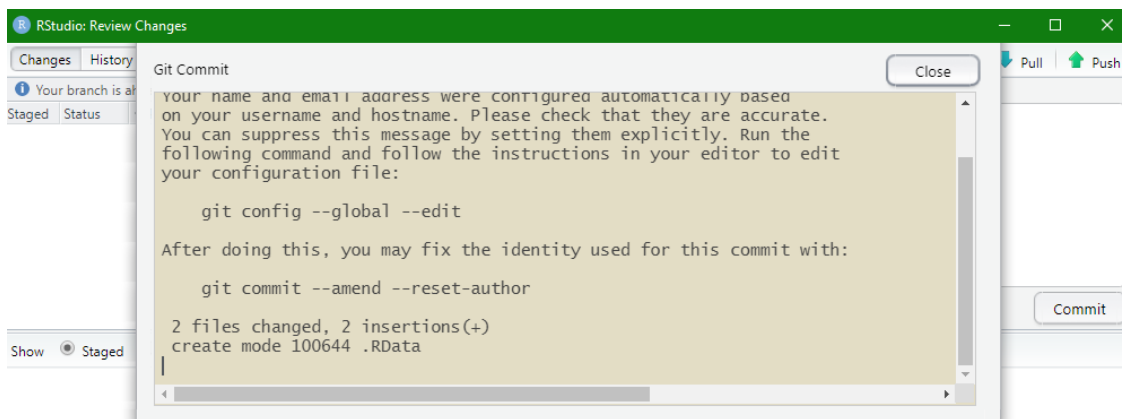


Dit zal een nieuw venster weergeven waar de gebruikers kunnen selecteren welke bestanden moeten worden vastgelegd. Over het algemeen worden alle bestanden geselecteerd. Het venster geeft aan welke wijzigingen werden aangebracht, toevoegingen zijn groen gekleurd en verwijderingen rood gekleurd.



Zodra alles is zoals het moet zijn, kan de vastlegging worden afgerond door een 'commit message' op te nemen en op de 'Commit' knop te drukken. Een bericht is verplicht om wijzigingen te kunnen vastleggen, en moet de reden aangeven waarom de vastlegging is gebeurd. Dit moet dus geen letterlijke omschrijving zijn van de code die is toegevoegd of verwijderd. Bijvoorbeeld, in voorbeeld hierboven zou het bericht 'Testing commit procedure for manual' moeten vermelden.

Indien alles correct uitgevoerd is, zal een derde venster verschijnen dat de voortgang toont van de uitgevoerde vastlegging. Als er geen problemen werden gevonden, zal de onderste stelling aangeven hoeveel bestanden er gewijzigd werden en op welke manier:



### 3.2.4 Push changes to the online repository

Als de gebruiker al de wijzigingen heeft vastgelegd, is het belangrijk om collega's te verwittigen dat deze wijzigingen werden aangebracht in de bestanden van deze specifieke branch. Om dit te doen moeten alle wijzigingen in de lokale repository geüpload of 'pushed' worden naar de online repository. Dit is het tegenovergestelde van de 'pull' actie die hiervoor al is uitgelegd. Het wordt sterk aanbevolen om eerst een 'pull' van de repository uit te voeren, om zeker te zijn dat de wijzigingen van de gebruiker worden geüpload naar de nieuwste versie van de online repository. Daarna kan de 'push' actie gestart worden van hetzelfde 'Git' tabblad.

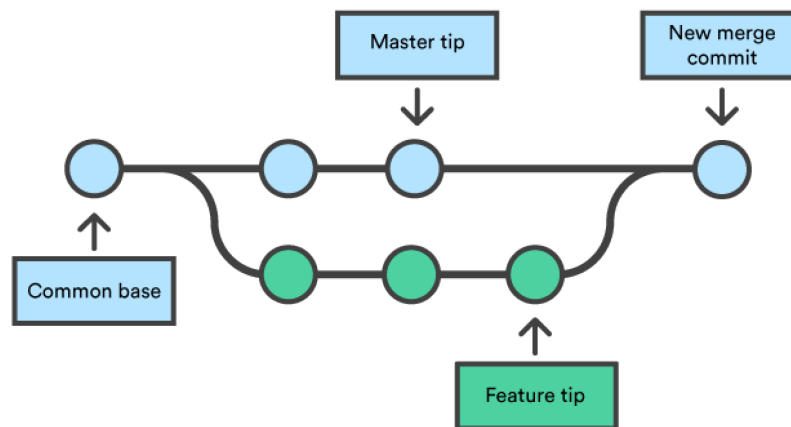
Bij het gebruik van de groene pijl verschijnt een voortgangsvenster waarin wordt aangegeven of de push succesvol is uitgevoerd of dat er problemen zijn opgetreden.

```
Git Push Close  
>>> C:/Program Files/Git/bin/git.exe push origin HEAD:refs/heads/master  
To https://github.com/iobis/findingdemo  
cff5b67..46defee HEAD -> master
```

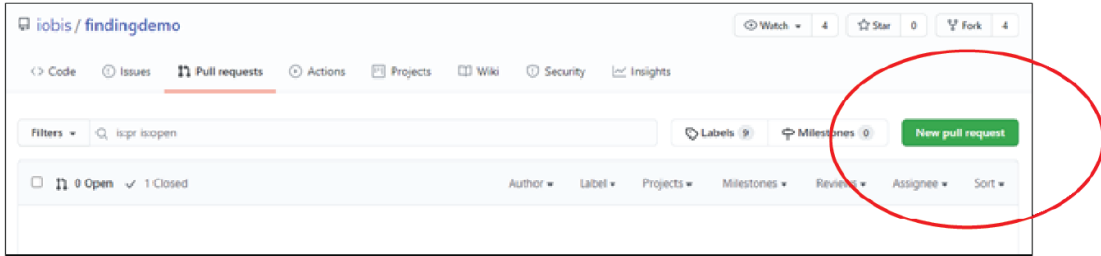
Het is gebruikelijk dat bij het uitvoeren een push voor de eerste keer, een fout optreedt met de melding "fatal: Authentication Failed". Dit betekent dat de inloggegevens van de gebruiker niet geldig zijn om toegang te krijgen tot de online repository. Om dit probleem te verhelpen moet de gebruiker eerst inloggen op de webbrowser op <https://github.com/>. Als alternatief kunnen inloggegevens worden beheerd met behulp van het inlogbeheer van de gebruikers hardware.

### 3.2.5 Branches samenvoegen

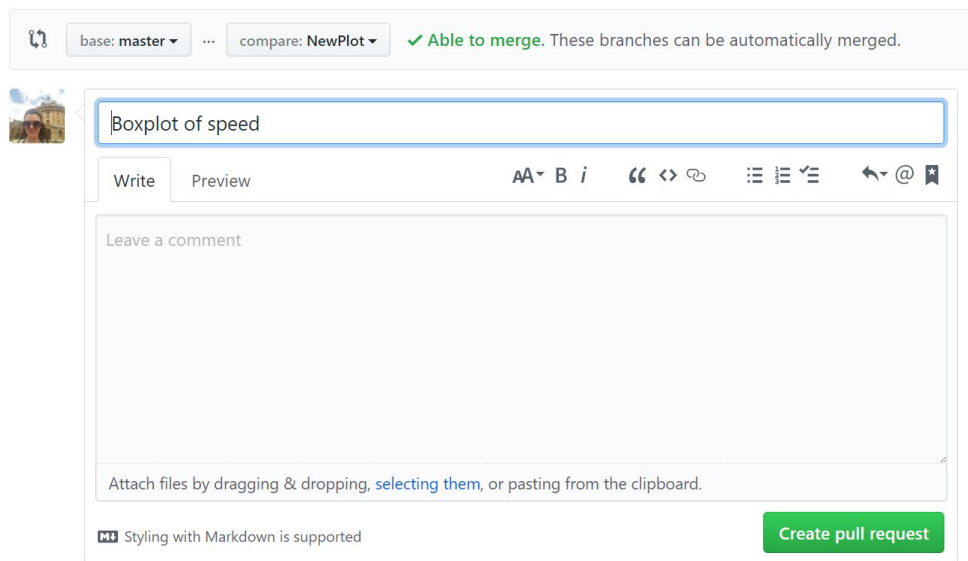
Wanneer een branch de bereikte ontwikkeling heeft voltooid, moeten de wijzigingen terug toegepast worden op bovenliggende branch voor de uitvoering in de hoofdfuncieregel. Om dit te doen worden branches samengevoegd of gecombineerd. In dit geval zal Git twee commit verwijzingen vinden, meestal de branch uiteinden, met een gemeenschappelijke basis commit ertussen. Eenmaal de gemeenschappelijke basis is gevonden, zal Git een 'merge commit' uitvoeren die de wijzigingen binnen elke branch samengevoegd, zoals hieronder getoond:



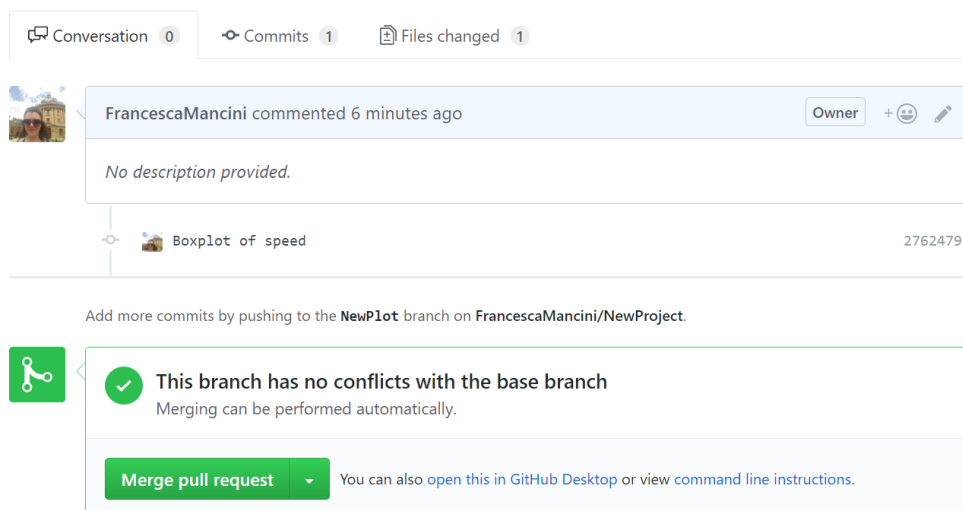
Branches samenvoegen wordt meestal gedaan vanuit de online repository op <https://github.com/>. Bij het openen van het tabblad 'Pull requests', kan een nieuwe samenvoegen plaats vinden met de 'New pull request' knop.



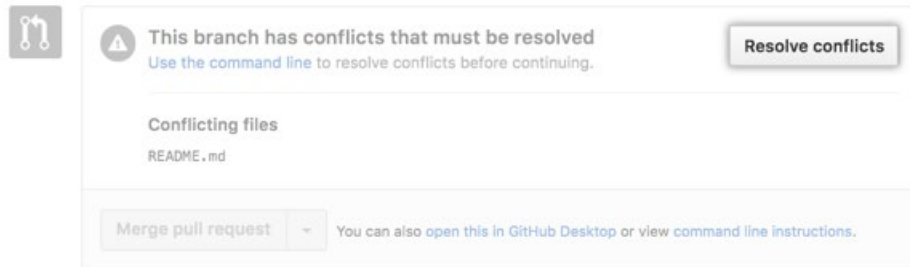
In het volgende venster moet een basis en een te vergelijken branch geselecteerd worden. De basisbranch is de bovenliggende branch waarmee de andere branch wordt samengevoegd. Voordat de pull aanvraag wordt gemaakt, kan een beschrijving worden toegevoegd van wat er is veranderd in vergelijking met de basis branch::



Zodra de aanvraag is gemaakt, vergelijkt GitHub de twee branches van het project. Als er geen overlappende of tegenstrijdige wijzigingen zijn, zal GitHub aangeven dat de branches rechtstreeks kunnen worden samengevoegd:



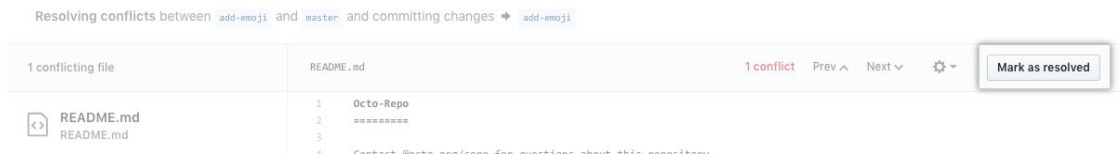
In dit geval kan indien nodig de onderliggende branch veilig verwijderd worden. Het is echter mogelijk dat er een conflict is. In dit geval wordt dit ook aangegeven door GitHub, en moet dit worden opgelost met de 'resolve conflicts' knop voordat het samenvoegen mogelijk is:



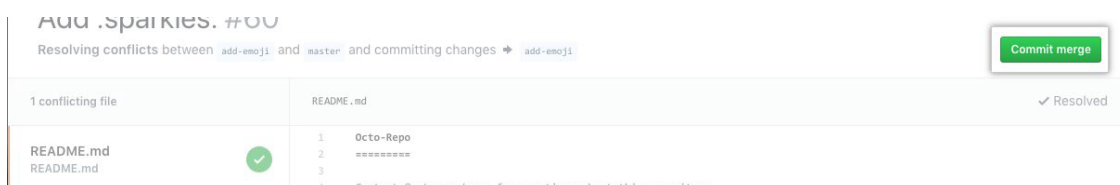
Indien deze knop gedeactiveerd is, zijn de conflicten te uitgebreid om opgelost te worden in GitHub. Anders zal een nieuw venster verschijnen waarin wordt aangegeven welke bestanden en welke secties de conflicten veroorzaken. Op dit punt moet er beslist worden als de gebruiker de bovenliggende branch versie, de onderliggende branch versie, of een geheel nieuwe wijziging wil maken die wijzigingen van beide branches bevat:



Enmaal alle problemen in het bestand opgelost zijn, klik 'Mark as resolved' voor dat bestand:

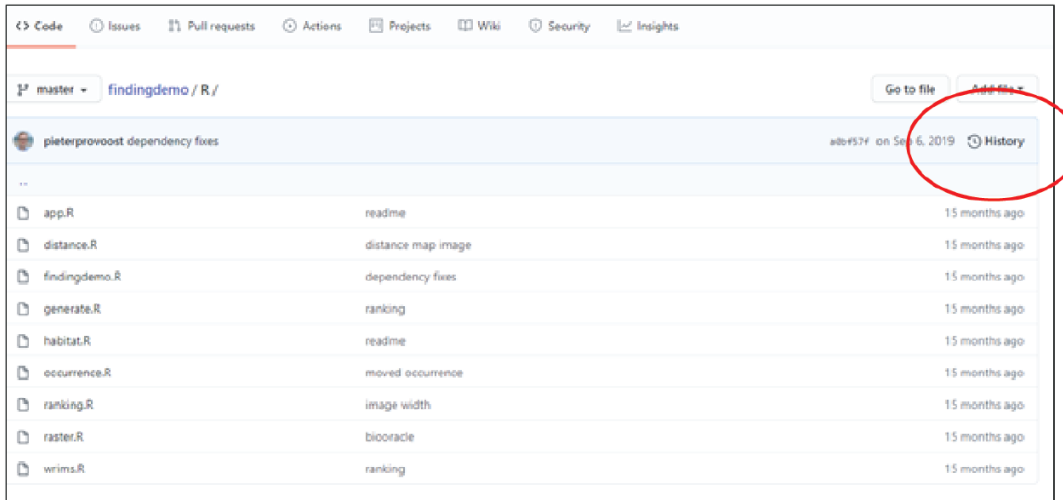


Als alle problemen in alle bestanden zijn opgelost, kan de samenvoeging uitgevoerd worden met de 'Commit merge' knop:

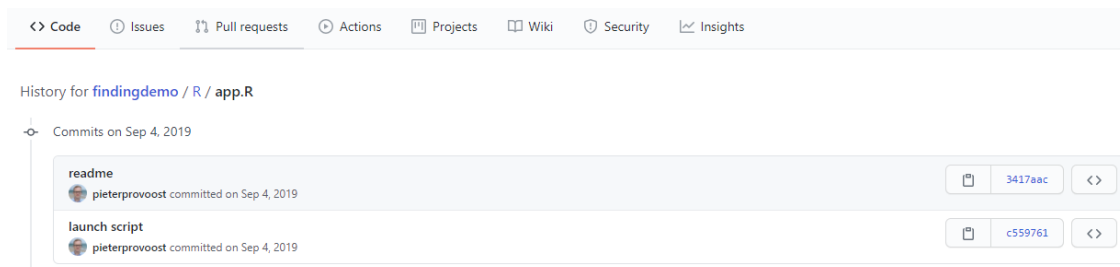


### 3.2.6 Versie bijhouden

Alle commits die in een GitHub repository worden uitgevoerd, worden gedurende het hele project bijgehouden. Zo kunnen gebruikers eerdere versies van mappen en bestanden verkennen om te zien wat is veranderd. Om dit te doen, heeft een gebruiker toegang tot de geschiedenis van elk gekozen bestand of elke map, of zelfs de hele repository:



De geschiedenis zal een overzicht geven van alle uitgevoerde vastleggingen. De exacte wijzigingen binnen elke vastlegging kunnen getoond worden door op een bepaalde commit te klikken:



## 3.3 Regels voor GitHub repository van T2021

### 3.3.1 Branches aanmaken

In de sectie Een nieuw project branch aanmaken, wordt uitgelegd dat het werken in branches nodig is om nieuwe ontwikkelingen binnen een GitHub project te maken. Aangezien de T2021 GitHub repository onderdeel is van een basis factureringsplan, kunnen alle medewerkers een nieuwe branch binnen de repository aanmaken. Het is daarom belangrijk om een duidelijk beeld te hebben van waarom takken worden gemaakt. Nomenclatuur is hierbij een belangrijk aspect.

Hoewel branch nomenclatuur vaak repository- of project specifiek is, zijn er drie basisregels die moeten worden gevolgd.

1. De branch naam moet een korte, bruikbare beschrijving bevatten van waar de ontwikkeling over gaat.

- Als een branch wordt aangemaakt om een probleem op te lossen, voeg dan het probleem tracker ID toe aan de branch naam.
- Gebruik koppelttekens (“-“) als scheidingstekens.

short, actionable description of what the task is about  


---

# 722-add-billing-module

lead with issue tracker ID for the task
use hyphens as separators

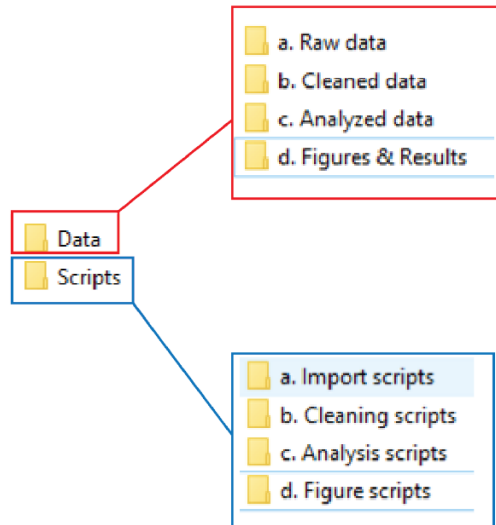
### 3.3.2 Mappen aanmaken

De mappenstructuur van de T2021 GitHub repository is gebaseerd op de hoofdstukken van de bijbehorende publicatie, zoals het geval was voor het T2015-data archief. Deze bestaat uit acht hoofdmappen die elk submappen bevatten. Deze hoofdstructuur mag enkel gewijzigd worden in overleg met de administrators van de repository.

The image shows two screenshots of a GitHub repository interface. The top screenshot displays the commit history for the repository '722-add-billing-module'. It shows a commit by 'Djro33' titled 'Create .gitkeep' with 40 commits. Below this, a list of files and folders is shown, including '0\_Algemeen', '3\_Dynamiek waterbeweging', '4\_Bevaarbaarheid', '5\_Plaat- en geulsysteem', '6\_Waterkwaliteit', '7\_Leefomgeving', '8\_Flora en Fauna', '9\_Ecologisch functioneren', and 'README.md'. The bottom screenshot shows the view of the '3\_Dynamiek waterbeweging' directory, listing sub-directories like '3.2\_Toetsparameter Hoogwater', '3.3\_Toetsparameter Golven', and '3.4\_Verklarende parameters', all created by 'Djro33'.

Databestanden en scripts die betrekking hebben op een bepaald hoofdstuk moeten worden opgeslagen in de daarvoor bestemde map. Om dit te doen kan een duidelijke en uniforme methodologie worden gebruikt. Databestanden

en scripts moeten in een aparte map worden opgeslagen. Hoewel het handiger lijkt om deze bestanden bij elkaar te houden, heeft het algemeen overzicht voordeel bij de twee-mappen structuur. Scripts en databestanden hebben vaak geen 1 op 1 relatie, aangezien één script meerdere databestanden kan gebruiken terwijl deze databestanden door meerdere scripts gerund kunnen worden. De structuur van elke map zou echter hetzelfde moeten zijn, met een map voor elke fase van het project:

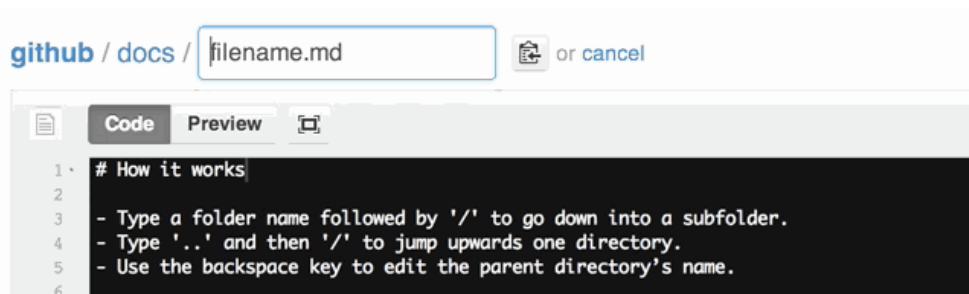


Door deze structuur te gebruiken kan een uniforme werkwijze opgesteld worden binnen de project directory. De werkwijze volgt vier vaste stappen die worden uitgelegd in volgende tabel:

|  | Using data from: | Using scripts or functions from: | Saving new data or results in: |
|--|------------------|----------------------------------|--------------------------------|
| <b>Step 1 - Import data (if necessary)</b> | n/a              | a. Import scripts                | a. Raw data                    |
| <b>Step 2 - Clean data</b>                 | a. Raw data      | b. Cleaning scripts              | b. Cleaned data                |
| <b>Step 3 - Anayze data</b>                | b. Cleaned data  | c. Analysis scripts              | c. Analyzed data               |
| <b>Step 4 - Create figures or results</b>  | c. Analyzed data | d. Figure scripts                | d. Figures & Results           |

Meer informatie over het aanbrengen van structuur in deze mappen, bestanden, scripts en code kan teruggevonden worden in de RStudio Handleiding van ScheldeMonitor.

Het aanmaken of verplaatsen van een map binnen GitHub kan worden gedaan door het aanmaken of wijzigen van een bestand binnen deze specifieke map. Als u dit doet verschijnt een venster onderaan die de gebruikers toelaat om de bestandsnaam te veranderen. Naast het veranderen van de bestandsnaam, kunnen gebruikers ook een map aanmaken door '/' te typen na de naam, waardoor het verandert van een bestandsnaam naar een mapnaam. Op deze manier kunnen mappen aangemaakt, verplaatst of verwijderd worden:



Belangrijk is dat GitHub het aanmaken van lege mappen niet toelaat. Daarom, als dit de bedoeling van de gebruiker is, moet een '.gitkeep' bestand toegevoegd worden in de nieuw aangemaakte map om het systeem te vertellen dat de map bewaard moet blijven.

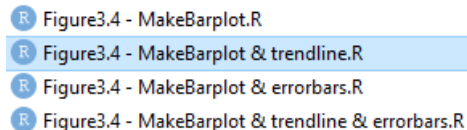
T2021 / 3\_Dynamiek waterbeweging / 3.2\_Toetsparameter Hoogwater /

Het is niet verplicht om een '.gitkeep' bestand op de lokale Git installatie te maken. Het '.gitkeep' bestand en andere wijzigingen kunnen ook gemaakt worden in de online GitHub repository T2021.

### 3.3.3 Bestanden aanmaken

Bestanden moeten zo genoemd worden zodat gebruikers makkelijk hun doel kunnen afleiden. Dit is voornamelijk belangrijk bij het werken met scripts die functies uit andere scripts in verschillende fasen van de code zoeken.

Bijvoorbeeld, wanneer u verschillende scripts voor verschillende grafieken gebruikt, moet de nomenclatuur duidelijk aangeven welk plot werd gemaakt met behulp van het script. Als het werk in de RStudio omgeving is gelinkt aan een bepaald rapport of publicatie, kan de figuurnummer van de publicatie opgenomen worden in de bestandsnaam. Het is ook mogelijk dat meerdere scripts gebruikt worden voor dezelfde figuur, waarbij in dit geval de nomenclatuur de verschillen moet tonen tussen de scripts. Deze regels kunnen resulteren in volgende nomenclatuur:



```
Figure3.4 - MakeBarplot.R
Figure3.4 - MakeBarplot & trendline.R
Figure3.4 - MakeBarplot & errorbars.R
Figure3.4 - MakeBarplot & trendline & errorbars.R
```

Meer informatie over dit topic, als ook de beschrijving van de algemeenste benamingsconventies kunnen teruggevonden worden in RStudio guide op de ScheldeMonitor website.

### 3.3.4 Code aanmaken

De samenwerking met meerdere gebruikers in deze GitHub repository vraagt duidelijkheid en reproduceerbaarheid van de code. Het is daarom belangrijk om richtlijnen te gebruiken en structuur toe te passen aan uw code, of het nu de R of Python programmeertaal is.

Zoals aangehaald in de introductie, focust deze handleiding zich voornamelijk op R, aangezien het bedoeld is om het gebruik van GitHub repository te combineren met de ScheldeMonitor Rstudio omgeving. Voor R is er geen duidelijke consensus over de beste werkwijze. De RStudio handleiding op de ScheldeMonitor website voorziet echter een uitgebreid overzicht van algemene richtlijnen voor:

- Hardcoding
- Spacing
- Code blokken
- Long lines of code
- Gebruik van pipes

De belangrijkste aspecten die kunnen worden toegepast op alle programmeertalen zijn het gebruik van structuur en aantekeningen. Structuur in de code maakt een uitgebreid script beter leesbaar en begrijpbaar. Voor deze handleiding wordt de volgende structuur voorgesteld:



- Wie, wanneer, wat en hoe: Dit is een grote kop die in het begin van elk script moet staan, en aangeven wie het script schreef, wanneer het werd geschreven, hoe de schrijver gecontacteerd kan worden en wat het doel is.
- 0 – Load libraries: In deze sectie worden alle ‘libraries’ opgelijst die moeten geladen worden voordat het volledige script wordt gerund. Deze sectie kan ook wat meer uitleg geven over het gebruik van de ‘libraries’.
- 1 – Static part: In dit deel worden alle statische zaken uitgevoerd zoals databestanden laden, deze data voorbereiden voor analyse, zoeken naar andere scripts functies of argumenten benoemen die later in het script zullen worden gebruikt.
- 2 – Script: Deze sectie bevat de eigenlijke code die ervoor zorgt dat het script zijn doel vervult.

```
#####
## This is an example for the manual
##
## written by Jelle Rondelez of VLIZ
## info@scheldemonitor.org - Oct 2020
#####

#####
# 0 - Load libraries
#####
library(dplyr) # package to clean datatable
library(lubridate) # package to change date formats

#####
# 1 - Static part
#####

#Assign variable
newvar <- ""

#Source script from within directory
source("Scripts/a. Import scripts/ImportsWFS")

#open data file
datafile <- read.csv(file = "Data/b. Cleaned data/dataRWS.csv")

#####
# 2 - Scripts
#####
code...
```

Aantekenen van de code is belangrijk voor een aantal redenen. Het helpt om in detail uit te leggen wat een regel, chunk of zelf sectie van de code probeert uit te voeren. Dit is behulpzaam voor gebruikers en andere mensen die de code lezen.

De code aantekenen wordt gedaan met het # (hashtag) symbool, dat boven een volledige chunk code staat, zoals wanneer het doel van een bepaalde functie wordt uitgelegd.

```
#Reactive values for user location
data_of_click <- reactiveValues(clicked = NULL)
longitude_click <- reactiveValues (lng = NULL)
latitude_click <- reactiveVaules (lat = NULL)

#If user clicks on map, new coordinates are saved and map is adjusted
```

```
observeEvent(input$Map_click, {
  data_of_click$clicked <- input$Map_click
  longitude_click <- input$Map_click$lng
  latitude_click <- input$Map_click$lat
  leafletProxy('Map')%>%
    clearMarkers()%>%
    addMarkers(lng = input$Map_click$lng,
              lat = input$Map_click$lat,
              popup = paste("Longitude=", round(input$Map_click$lng, 2),
                            "and",
                            "Latitude=", round(input$Map_click$lat, 2)))
})
```

## 4 Helpdesk

VLIZ is verantwoordelijk voor het beheer van de GitHub organisatie van ScheldeMonitor. Dit impliceert het beheer van de organisatie-instellingen, de inhoud ervan en alle medewerkers. De GitHub organisatie is bedoeld als de centrale hub voor alle repositories gewijd aan onderzoek en analyse gebaseerd op de ScheldeMonitor data en informatie, en projecten die door de VNSC worden gefinancierd.

Om aan deze en andere noden van gebruikers en medewerkers te voldoen, heeft VLIZ een permanente helpdesk. Deze helpdesk kan gecontacteerd worden via het algemene adres van de ScheldeMonitor:

### Helpdesk ScheldeMonitor

Data Centre - Local Services & Projects



**Vlaams Instituut voor de Zee vzw** T +32(0)59340172  
**Flanders Marine Institute** info@scheldemonitor.org  
 InnovOcean site, Wandelaarkaai 7 www.vliz.be  
 8400 Oostende, Belgium

Voor dringende zaken of vragen, of indien gebruikers en medewerkers het gebruik van de RStudio omgeving willen bespreken voor bepaalde projecten, kan de project manager van ScheldeMonitor gecontacteerd worden:

### Jelle Rondelez

Project Manager

Data Centre - Local Services & Projects



**Vlaams Instituut voor de Zee vzw** M +32(0)473510828  
**Flanders Marine Institute** jelle.rondelez@vliz.be  
 InnovOcean site, Wandelaarkaai 7 www.vliz.be  
 8400 Oostende, Belgium