Voor Git beter te begrijpen hebben we al veel gedaan om het te verstaan. Ook ik wil jullie helpen door dit bestand te maken.

Er kunnen fouten instaan en je mag ze altijd zeggen.

Dit gaat echt de basis zijn.

Open GitKraken

Er is zijn 2 opties clone en init.

Init gaat een nieuwe repository maken terwijl Clone een bestaande ONLINE repository gaat kopiëren.

Ga via tablad File naar Init repository

Dan ga je naar GitLab.com of GitHub.com

Hier ga je een Repository aanmaken (dit zijn de bestanden van je project)

Vul je Account, je projectnaam, je projectomschrijving en je pad waar je LOKALE repository gaat inkomen.

Dan kan je kiezen om een .gitignore bij te maken maar dit moet niet want je kunt hem ook zelf aanmaken nadat je project is gemaakt.

Als alles goed is verlopen kun je op Create Repository and Clone klikken.

Nu kun je je project bekijken op GitLab.com (of GitHub.com).

Als je ingelogd bent kun je normaal een kijkje nemen naar het project.

Eerst gaan we de instellingen verandern

Die staan links bij ij het tandwieltje.

//zowel in github als in gitlab dien je mensen toegang te geven om te mogen pushen naar de repository, dit kan je vinden onder de term 'collaborators' op github.

Je kan hier indien je dit nog niet gedaan had bij permissions de zichtbaarheid op privaat zetten (dit moet niet maar zorgt ervoor dat er geen vreemde mensen op komen).

Dan kan je als je over instellingen hovert naar members gaan.

Via de namen van GitLab kun je personen inviten aan je projecten.

Dan kan je de permissions zetten op bijvoorbeeld master zodat ze alles kunnen veranderen in het project.

Als je members hebt toegevoegt kan je die onderaan zien.

Ga naar het pad dat je bij het project hebt opgegeven en maak daar eerst en vooral de .gitignore aan

Wat er moet instaan vind je via google of de Powerpoint van git. Bij Phpstorm ga je idea/\* erin schrijven

Dan kunnen we beginnen:

Ik verkies om mijn lokale repository te bewerken met PHPstorm.

Open je PHPstorm en kies bij het tablad file Open directory...

Ga naar het pad waar je lokale git repository plaatsvindt en open die map.

Als je een nieuw bestand maakt zal Phpstorm vragen of je het in de git wil steken, aan jou de keuze, bij het aanmaken van folders krijg je die melding niet, deze zullen automatisch meegestuurd worden wanneer je er een bestand in aanmaakt

Maak nu de basis met de juiste mappenstructuur (index.html met js en css's links, screen.css, reset.css en main.js)

Als dit gedaan is gaan we nu met Git beginnen werken.

Ga nu dus terug naar je GitKraken

Dan staat er in het rechterscherm de veranderingen die je gemaakt hebt.

Als je dit niet ziet dan staat er links bovenaan view changes.

Stage nu alle changes en voeg een mooie Commit toe met naam en beschrijving.

Commit uw wijzigingen.

Nu zie je een extra bolletje.

Dan stuur je dit naar de online repository door, door een push te doen.

Het is niet verplicht om onmiddellijk een branch aan te maken om in te werken, maar dat is wél nu net het principe van Git, je beschikt nu over 1 branch met als naam de 'master branch'.

Om op een kopie van de branch 'master' te werken, dien je een nieuwe branch aan te maken, geef dit een goede naam: best te vernoemen naar een feature die je zal maken!!!!!

//Maak een nieuwe branch aan, bijvoorbeeld: "

Push die ook naar de server en druk op submit zodat development bij de branches van remote (online) komt

Vanaf nu ga je volgende stappen overlopen:

Maak een branch met als naam de feature dat je aan gaat werken.

Als je dit nog wil opslpitsen in minifeature's kun je dit stappenplan ook volgen.

Zorg in elk geval dat je het niet te complex maakt.

Als je iets aangepast hebt, kun je daarvan een commit maken en de file 'saven', wanneer je vervolgens je commits pusht, dan zal enkel je huidige branch gewijzigd worden.

Als er een probleem is met gewijzgde code in hetzelfde bestand, dan spreken we over een 'merge conflict', dit kan je oplossen door te kiezen wat je erin wilt houden:

Je kan zien wat je hebt aangepast door te klikken op de file en er staat code met groen aangeduid voor toegevoegd en rood voor verwijderde code.

Als het merge-conflict is opgelost kan je pushen, daarna zal er een nieuw bolletje tevoorschijn komen.

Als je feature af is dan wil je dit terug mergen in de 'develop'branch (kan ook een andere naam hebben bij u). Om dit te doen, moet je

Voordat je dit in de development branch wil mergen ga je eerst je code laten reviewen door iemand anders.

Dit doe je door je branch te pushen en zo beschikbaar te stellen voor iedereen in het project.

Je kan hierbij ook werken met Pull Request.

Dit is een vraag aan je teamleden of je jouw feature in de devlopment lijn wil steken.

Als alles goed verloopt mag je beginnen met mergen.

Doe dit door rechtermuisklik op development en merge je in development.

Dan dubbelklik je op development of rechtermuisklik en doe je checkout development.

Als laatste push je naar de server.

Hij zal hier vragen (<=> je teamleden al iets gepushed hebben in development)

of je alles wil pullen en fast forwared als dat mogelijk is.

Als dit niet zo is dan heb je een merge conflict.

Dan ga je in je bestand zelf de code aanpassen die conflicten heeft zodanig dat de online code hetzelfde is.

Nog wat extra info:

Pullen in een branch zorgt ervoor dat je local branch alle files krijgt van de remote (online) branch met dezelfde naam.

Pushen in een branch stuurt alle files naar je online projectruimte.

Branches maken is nodig zodat je als je een foutje hebt gemaakt in een file (en je kunt niet ctrl-Z doen) dat je 'terug in de tijd' kunt om de files van vroeger te herbewerken.

Commits zijn je saves.

Doe ze niet te groot en zorg dat je commits een duidelijke naam en omschrijving krijgen.