

Development Bootcamp

Inhoudsopgave

Inleiding	2
Git Workshops	2
Repository Opzetten	2
Branches	3
Readme.md	3
.gitignore file	4
Repository Forken	4
Debugging with Chrome Tools	4
Layouts met Grid en Flexbox	4
Companion Web App	6
App Shell	6
API, NodeJS & Postman	6
Conclusie	7
Feedback	7
Reflectie	7

Inleiding

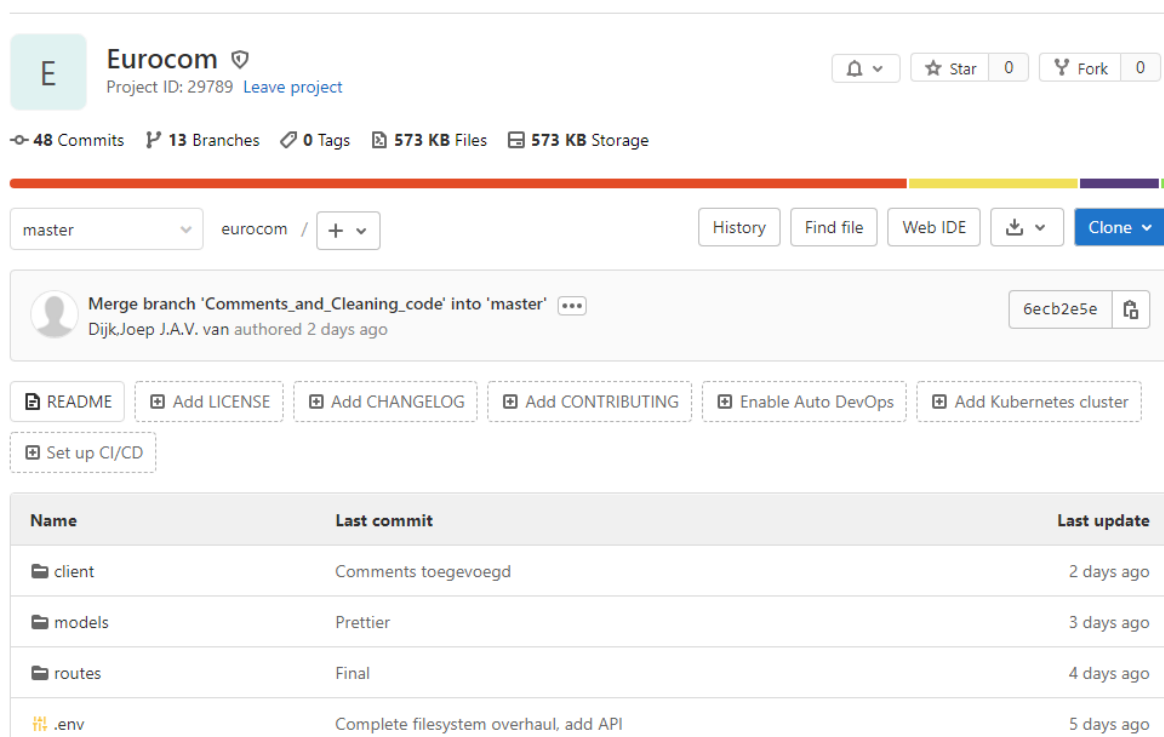
In dit semester krijgen we elke week een dag voor development. Op deze dag worden workshops gegeven over verschillende dingen waaronder flexbox, grids, Git, debuggen, coderen en nog veel meer. Bij sommige workshops hoorde ook oefeningen die ik vervolgens heb gemaakt. In dit document bespreek ik de dingen die ik heb geleerd van de workshops en bijbehorende oefeningen die ik heb gedaan.

Git Workshops

In de Git workshops hebben we veel informatie gekregen over het gebruik van Git en Gitlab. Omdat ik dit allemaal al heb gebruikt in het vorige semester wist ik al hoe dit werkte. Hierdoor kon ik makkelijk de repository opzetten voor mijn companion app en de repository voor ons Eurocom project. Ook heb ik geleerd wat forken is en hoe branches werken. Door met branches te werken kon iedereen in ons groepje aan een bepaald gedeelte van de code werken zonder elkaar in de weg te zitten. Gelukkig hebben ze in de workshops ook verteld hoe je de bijkomende merge conflicts goed kunt oplossen.

Repository Opzetten

Dit heb ik gedaan voor mijn persoonlijke code waaronder de companion app en de code voor ons Eurocom project. We hebben op git.fhict.nl een repository gemaakt en vervolgens de link gekopieerd en lokaal op de computer gezet met de clone command. Nu iedereen een lokale versie heeft kunnen we code veranderen en dit pushen, hierdoor kunnen andere mensen de code weer pullen en verder werken in de meest recente versie.



Eurocom Project ID: 29789 [Leave project](#)

48 Commits 13 Branches 0 Tags 573 KB Files 573 KB Storage

master eurocom / +

History Find file Web IDE Clone

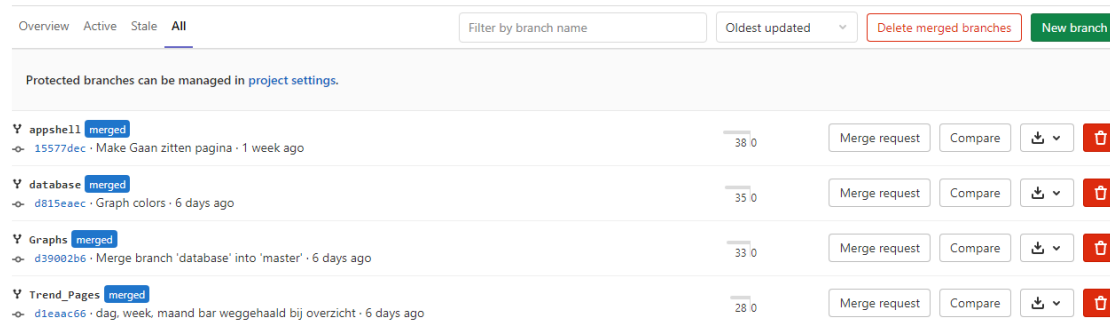
Merge branch 'Comments_and_Cleaning_code' into 'master' 6ecb2e5e
Dijk,Joep J.A.V. van authored 2 days ago

[README](#) [Add LICENSE](#) [Add CHANGELOG](#) [Add CONTRIBUTING](#) [Enable Auto DevOps](#) [Add Kubernetes cluster](#)
[Set up CI/CD](#)

Name	Last commit	Last update
client	Comments toegevoegd	2 days ago
models	Prettier	3 days ago
routes	Final	4 days ago
.env	Complete filesystem overhaul, add API	5 days ago

Branches

Als mensen tegelijk willen werken aan een project kan het voorkomen dat iemand niet de meest recente versie pulled of code overschrijft waardoor er code verloren gaat. Om samenwerken makkelijker te maken kun je gebruik maken van branches. Als je deze opdeelt per onderdeel van een project kan iedereen in een andere branch werken en deze uiteindelijk mergen. Zoals je hieronder kan zien hebben we veel verschillende branches gemaakt voor bijvoorbeeld de appshell, grafieken en connectie met de database.

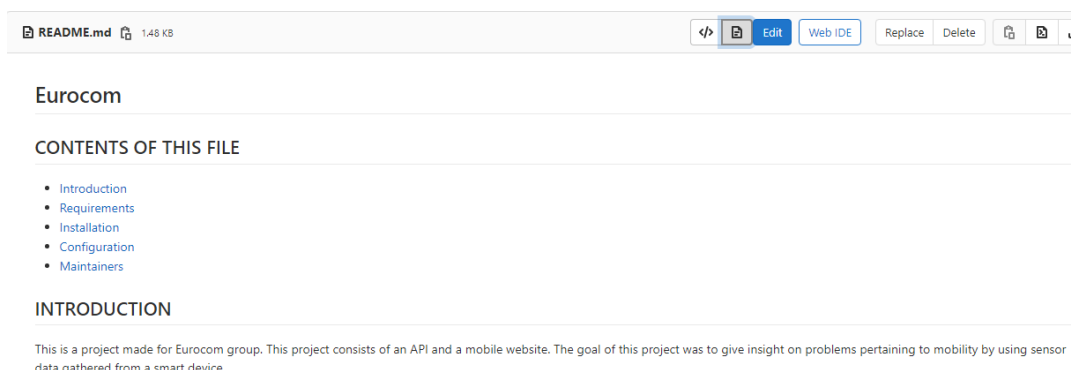


Nadat je klaar bent in een branch kun je deze mergen. Wanneer je compleet nieuwe code schrijft is er vaak geen probleem maar als mensen dingen hebben veranderd in dezelfde file kan er soms een merge conflict ontstaan. Bij een merge conflict kan de maintainer kijken welke code goed is en dus welke code gebruikt gaat worden. Je kunt er vaak ook voor kiezen om beide codes bij elkaar te voegen en te bewaren. Hieronder zie je hoe zo'n merge conflict eruit kan zien.

```
14 let button = document.createElement('button');
    Accept current change | Accept incoming change
15 <<<<<< HEAD (Current change)
16 button.textContent = "Say Hey";
17 =====
18 button.textContent = "Say Yo";
19 >>>>>> feature-1 (Incoming change)
20 button.onclick = function() {
21     alert(greeter.greet());
22 }
```

Readme.md

In elke repository hoort ook een Readme bestand (en soms ook meerdere). Hierin geef je een soort introductie van je project en de informatie die iemand nodig heeft om deze te gebruiken. Je verteld dus kort wat het doet, wat je nodig hebt om dit te gebruiken of te installeren en andere handige commands of functies die je kunt gebruiken. Hieronder zie je een deel van de readme die we voor ons Eurocom project hebben gemaakt.



.gitignore file

Ook is het soms handig om een .gitignore file aan te maken. Dit is een bestand in je project die niet naar git gepusht wordt maar dus genegeerd wordt. Dit is handig wanneer je met wachtwoorden werkt of node.js die anders enorm veel bestanden zou versturen. In ons project hebben we ook gebruik gemaakt van een .gitignore file omdat we ook node.js hebben gebruikt.

Repository Forken

Naast het opzetten van een repository en clonen kun je ook forken. Hiermee kopieer je de complete repository (van een ander persoon) en plakt deze als een eigen repository in je Gitlab. Hierna kun je deze lokaal op je computer zetten en eventueel zelf aanpassingen maken. Ik en Coen hebben dit uitgetoetst en alles werkte meteen.

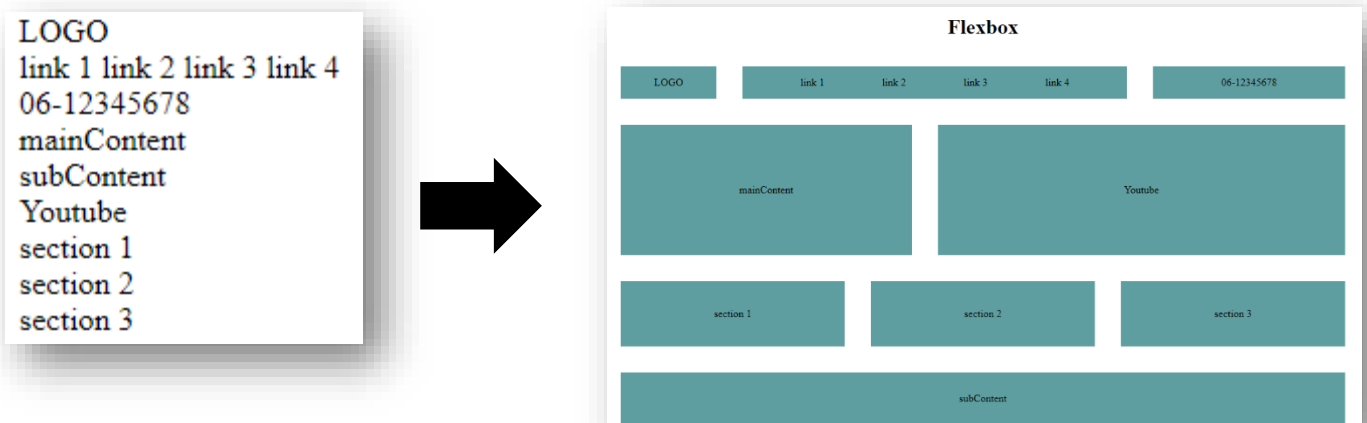
Debugging with Chrome Tools

In deze workshop vertelde Geert-Jan over het debuggen en de verschillende tools die je hiervoor kunt gebruiken. Een van de meest gebruikte dingen is het console.loggen van data in je javascript en deze weergeven in console. Hierdoor zie je soms waar het misgaat en of je bepaalde errors krijgt.

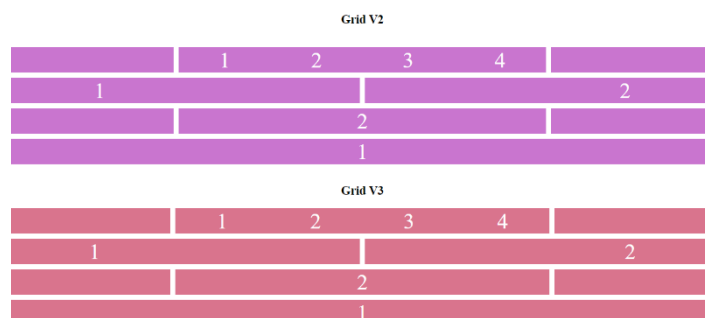
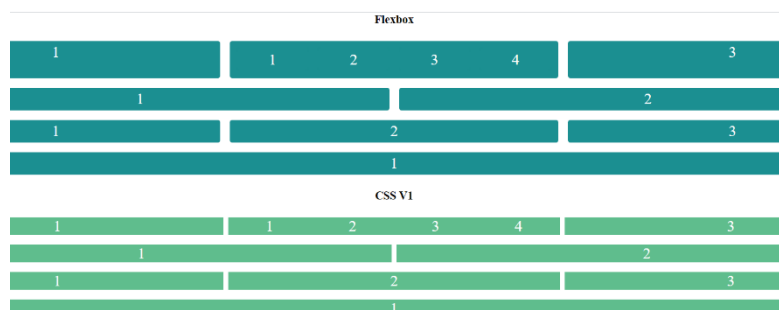
Layouts met Grid en Flexbox

Ook hebben we een workshop over layouts met grid en flexbox gehad. Hierin werd verteld wat grid en flexbox zijn en hoe je deze gebruikt. Hierna kregen we een oefening om een layout te maken met zowel grid als flexbox. Ik heb deze oefening gemaakt zodat ik deze twee manieren beter onder de knie kreeg. Ook kun je Grid en Flexbox oefenen door online games zoals Flexbox Froggy en Grid Garden.

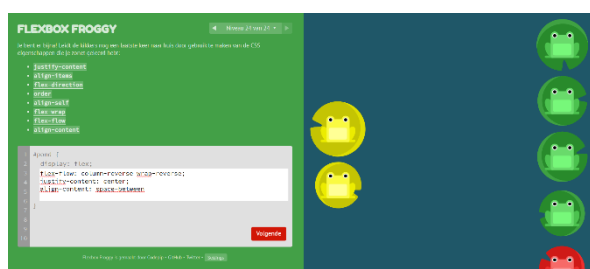
In de eerste oefening kregen we een simpele html file met daarin wat content. Deze content moesten we doormiddel van flexbox in een goede layout zetten. We kregen een voorbeeld te zien in de presentatie en zo moest de layout eruit komen te zien. Links zie je de content zonder flexbox en rechts met flexbox. De code hiervoor vind je op <https://git.fhict.nl/I437759/semester-3-media/-/tree/master/Poc's/Flexbox%20en%20Grid%20Layout%20test/Test%201>



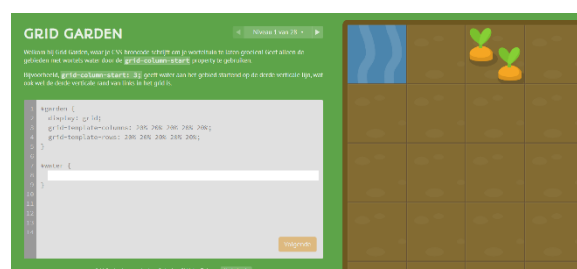
Verder heb ik nog geoefend met Grid. Hieronder zie je dezelfde layouts maar dan met verschillende code gemaakt. De code hiervoor kun je vinden op <https://git.fhict.nl/l437759/semester-3-media/-/tree/master/Poc's/Flexbox%20en%20Grid%20Layout%20test/Test%20>



Daarnaast zijn er nog leuke spelletjes die je kunt spelen om meer bekend te worden met Flexbox en Grid, hier heb ik veel van geleerd en deze kun je ook spelen door op onderstaande links te klikken.



<https://flexboxfroggy.com/#nl>



<https://cssgridgarden.com/#nl>

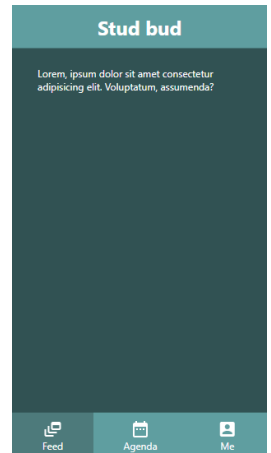
Companion Web App

In de development workshops werken we soms ook aan onze companion app om een aantal dingen te leren. We mochten zelf een app bedenken die ons van pas zou komen tijdens het semester. Deze webapp kan gebruik maken van een API en database en zou bijvoorbeeld je rooster kunnen laten zien door deze informatie op te vragen en weer te geven. Hier hoorde ook een aantal workshops bij zoals de workshop over app shells of API's.

App Shell

De eerste stap voor het maken van de app was een app shell maken. Dit is de html, css en javascript waar de basis van de app uit bestaat en die vaak als cache opgeslagen word waardoor de app snel blijft functioneren. Hieronder zie je hoe de App Shell eruit ziet, de code is te vinden op <https://git.fhict.nl/l437759/semester-3-media/-/tree/master/Companion%20web%20app>

De App Shell bestaat uit 1 Html file met daarin alle schermen. Door 1 scherm de hele hoogte en breedte van een normaal scherm te geven staan de andere 2 schermen als het ware buiten beeld. Geert-Jan had laten zien hoe je met javascript tussen deze pagina's kan switchen door op een knop te drukken. Ik was er achter gekomen hoe dit ook kan met alleen css. Omdat dat simpeler is heb ik dat dus via de css gedaan. Wel heb ik javascript gebruikt om een class weg te halen en toe te voegen aan de bottom navigation. Bij de code staan comments die precies uitleggen hoe het werkt. Hieronder zie je een stukje javascript uit de companion app code.

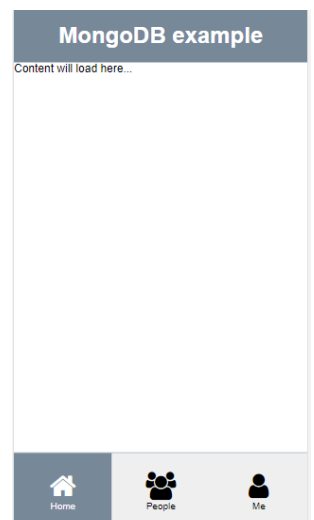


```
Companion web app > # scripts > window.addEventListener("load") callback
1  window.addEventListener("load", (e) => { //dit voert alle code uit die erin staat wanneer alles is geladen, soms werkt je javascript code niet als nog niet alles is geladen
2    console.log("DOM loaded. Yay!"); //dit kun je ook oplossen door de javascript aan het einde van je HTML op te roepen ipv bovenaan
3
4
5    //add event listeners for 'click' to our nav links
6
7    var navButtons = document.getElementsByClassName("tab"); // zorgt ervoor dat de class met name tab word geselecteerd en in var navButtons word gezet
8    // var navButtons = document.querySelectorAll(".tab"); kan ook maar dan moet je wel .tab gebruiken omdat dit meer kan zijn dan alleen een class
9    console.log(navButtons); //hiermee zie je in de console van de inspector wat navButtons is
10
11   for(navButton of navButtons) {
12     navButton.addEventListener('click', navClickHandler);
13   }
14
15   function navClickHandler(event) {
16     // make all buttons inactive (remove class active)
17     for (navButton of navButtons) {
18       navButton.classList.remove("active");
19     };
20
21     // make the clicked button active
22     this.classList.add('active');
23   }
24
```

API, NodeJS & Postman

In de workshop werd uitgelegd hoe we een Rest API konden maken, hiervoor gebruikte we MongoDB en moesten we NodeJS installeren. Ik heb de database opgezet met MongoDB en vervolgens de connectie string in server.js vervangen met die van de database, hiervoor moest ik mijn wachtwoord en database naam toevoegen.

Vervolgens kan je Postman downloaden, hiermee kun je dingen posten in de data van je database. Je moet er wel voor zorgen dat je API server runt door dit in de terminal in te voeren. Nu kun je bijvoorbeeld gegevens van een persoon posten en deze verschijnt daarna in de appshell. Ik heb dit uiteindelijk alleen gedaan in deze oefening en niet in mijn companion app omdat we dit later toch zouden gebruiken voor ons Eurocom project. Je ziet hieronder hoe de testversie eruit zag.



Conclusie

Ik weet nu beter hoe Git en Gitlab werken. Ik kan nu samen in een team werken door het gebruik van branches en het oplossen van merge conflicts. Ik heb geleerd hoe je een goede Readme maakt, een project kunt forken, wat een .gitignore is en wat goede manieren zijn om te debuggen.

Ik kan nu goede layouts maken door Flexbox en Grid en weet welke ik kan gebruiken in bepaalde situaties. Ook weet ik nu hoe ik een Rest API maak, hoe ik een noSQL database opzet met MongoDB en hier data naar kan pushen met postman. Ook kan ik deze data fetchen in mijn app. Hiervoor heb ik ook NodeJS moeten gebruiken.

Feedback

Bij elke workshop kregen we een moment om aan de opdrachten te werken en daarna een moment om vragen te stellen als je deze had. Ik heb bij elke workshop laten zien wat gelukt was en wat niet gelukt was. Geert-Jan zei dat mijn code er goed uitzag en vond het ook goed dat ik op een andere manier gecodeerd had dan hij had voorgedaan (door iets met css te doen ipv javascript). Hij zei dat er altijd meerdere manieren zijn om iets te doen en het belangrijkste is dat alles goed werkt en de code netjes blijft.

Ook moet je blijkbaar branches maken voor de onderdelen in je project, niet voor de personen in een project. Ik dacht eerst dat iedereen zijn eigen branch krijgt, maar het is de bedoeling dat je bijvoorbeeld een branch maakt voor de back end, front end en andere bepaalde functies.

De API werkend krijgen vond ik moeilijk en ik heb veel hulp gevraagd aan Geert-Jan. Uiteindelijk gaf hij me een paar tips zoals het gebruik van Postman en hoe ik data moet fetchen en pushen. Ik moet goed de documentatie lezen als deze aanwezig is omdat hier vaak goed beschreven staat wat je moet doen.

Reflectie

Ik wist al veel van Git maar had nog nooit echt git gebruikt in teamverband. Omdat ik samen met Coen een repository aanmaakte konden we veel oefenen met branches en merge conflicts, hierdoor heb ik een beter beeld gekregen van hoe het is om in een team aan een project te werken via Git en Gitlab.

Ook wist ik al hoe ik met Flexbox moest werken maar had nog nooit met Grid gewerkt. Ik heb hier een paar oefeningen mee gedaan en kreeg dit goed onder de knie, toch vind ik flexbox over het algemeen fijner.

Ik had nog geen enkele ervaring met databases en API's. Ik vond dit gedeelte ook wat lastiger. Uiteindelijk lukte het me om een API en database connectie te maken. Hierdoor kon ik data pushen naar de database en fetchen in de app. Ik heb geleerd hoe ik met Postman en MongoDB te werk ga. Dit lijkt me enorm handig voor ons Eurocom Project en ik ga hier zeker meer gebruik van maken.

Voor de volgende keer zou ik meteen goed de documentatie lezen als deze aanwezig is, hier staat vaak goed in beschreven wat je moet doen en als ik dit meteen had gedaan, was ik niet zo vaak vastgelopen op dingen.