2 januari 2023

Elise

PIT Roc tilburg

Spoorlaan 21j

Opensource en wat is het

Onderzoeks document

Inhoud

[Hoe zijn succesvolle opensource projecten opgezet in de afgelopen 2-5 jaar? 2](#_Toc115683392)

[Hoofdvraag 2](#_Toc115683393)

[Deelvragen 2](#_Toc115683394)

[Deelvragen 3](#_Toc115683395)

[Wat is opensource 3](#_Toc115683396)

[Is opensource software echt gratis? 3](#_Toc115683397)

[Opensource definitie: Ik neem wat en geef wat terug 4](#_Toc115683398)

[Hoe werkt een opensource project 4](#_Toc115683399)

[Wat maakt een opensource project succesvol 5](#_Toc115683400)

[De instelling en houding 5](#_Toc115683401)

[Documentatie 6](#_Toc115683402)

[Een probleem 6](#_Toc115683403)

[Quality 6](#_Toc115683404)

[Wat zijn de gevaren van een opensource project 7](#_Toc115683405)

[Wat zijn de gevaren van opensource 7](#_Toc115683406)

[Kan je de gevaren uit de vorige sub-vraag voorkomen? 8](#_Toc115683407)

[Conclusie 9](#_Toc115683408)

[Plan voor opzet 10](#_Toc115683409)

[Vooraf 10](#_Toc115683410)

[Waar te beginnen. 10](#_Toc115683411)

[Documentatie 10](#_Toc115683412)

[Readme.md 10](#_Toc115683413)

[Code of Conduct 10](#_Toc115683414)

[How to contribute document 11](#_Toc115683415)

[Wat nu? 11](#_Toc115683416)

[Bronnen 12](#_Toc115683417)

# Hoe zijn succesvolle opensource projecten opgezet in de afgelopen 2-5 jaar?

## Hoofdvraag

Voor mij is het de eerste keer dat ik een vraag kreeg om een opensource project te maken. Dit gaat gepaard natuurlijk met de vraag: Hoe zet je een opensource project op en wat maakt het succesvol?. Met dit onderzoek wil ik 3 doelen aan bod brengen:

1. Wat is opensource?
2. Wat maakt opensource succesvol?
3. Hoe zet je een opensource project op?

Mijn hoop is dat je na het lezen van het onderzoek weet:

1. Wat opensource is
2. Wat opensource succesvol maakt
3. Hoe je een opensource project opzet.

## Deelvragen

Natuurlijk komt er met een onderzoek een aantal deelvragen. De deelvragen zijn ook specifiek gericht op het onderwerp opensource. Met behulp van de volgende deelvragen wil ik de hoofdvraag beantwoorden:

1. Wat is opensource
2. Hoe werkt een opensource project
3. Wat maakt een opensource project succesvol?
4. Wat zijn de gevaren van een opensource project?
   1. Wat zijn de gevaren van opensource
   2. Hoe kan je de gevaren van 4a voorkomen?

De deelvragen zijn erop gericht om het onderwerp te begrijpen. Erna met behulp van de kennis te kijken wat het succesvol maakt. Tot slot gaan we na de conclusie nog een plan opstellen om een opensource project op te zetten.

## Onderzoek methode

Ik wil de volgende methode gebruiken om dit onderzoek te doen.

### Libary

Van de libary tak wil ik de volgende 2 methodes gebruiken. Met Literature study wil ik graag kijken hoe andere projecten het doen in de echte wereld. Met behulp van bad the good and the best wil ik kijken wat er fout ging en wat een project succesvol maakt.

### Workshop

Natuurlijk na het onderzoeken van wat fout gaat en wat goed moet er een proof of concept komen. Tenslotte moet je natuurlijk uitproberen en laten zien wat het idee voor het project. Ook moet er documentatie komen met hoe mensen het moeten gebruiken.

### Stepping stones

Ik neem ook 2 dingen van stepping stones mee in het onderzoek. Scenario`s om beter uit te kunnen leggen hoe bepaalde opensource dingen werken. Met ook toch het stukje concepten weer. Tenslotte is het pas aan en kijk hoe het werkt.

# Deelvragen

## Wat is opensource

Opensource wordt vaak gezien als: Code die iedereen kan zien en gratis is? Vaak klopt dit niet in de praktijk. In plaats van het direct uit te leggen lijkt het mij beter een voorbeeld te geven.

Als we een project op zouden zetten dat een package is voor react. In de package kan je hulp krijgen bij het ophalen van JSON API data. Dit zal het makkelijker maken om de Api data uit te lezen en te begrijpen. Je denkt opzetten van een plek waar ik mijn code kan bewaren en klaar. Dit klopt niet helemaal.

Het klopt je code is nu zichtbaar voor iedereen die dat wilt. Iedereen kan je code gebruiken en doen wat ze willen. Alleen er is een probleem.

Ik moet wel zegen van tevoren dat mijn project op github komt te staan. Dus vandaar dat ik heb gekeken naar de regels bij github. Als je het op een andere website zet vergeet dan niet te kijken naar de regels hiervoor.

Github regels gelden dat op het moment van geen licentie in de repoisrty dan gelden de rechten van standaard copyright laws. In dit geval geldt de regel dat niemand je code zomaar:

1. De code niet mag worden gedistribueerd die in de repo staat.
2. De code niet mag worden gereproduceerd die in de repo staat.
3. Geen afgeleid werk mag worden gemaakt van je code die in de repo staat.

Natuurlijk zijn er gevallen waarin dit nog steeds gebeurt en de regels worden overtreden. Rechtmatig kan je dan naar de rechter stappen. Dit kost natuurlijk heel wat en is het meestal 9 van de 10x niet waard. Om te voorkomen dat er zoiets gebeurt gaan we een licentie toevoegen.

Een standaard opensource licentie is MIT. De MIT Licentie zorgt ervoor dat bijna alles mag met je code. Enige conditie is dat de copyright licentie meegaat in elk kopie zolang. Als je voldoet aan de conditie zijn er geen problemen.

Om het kort te houden is dit eigenlijk een beetje wat opensource is. Ook heb ik al eigenlijk uitgelegd hoe opensource werkt. Toch wil ik nog 2 kleine dingen bespreken met betrekking tot opensource.

### Is opensource software echt gratis?

Iets wat vaak voorbij kwam in het onderzoek was: Is opensource nou gratis of niet? Dit heeft een ja kant en een nee kant. Het hangt namelijk af van de soort software en wat je er mee doet. Opensource in het algemeen is altijd gratis zolang je de rechten van de licentie volgt.

In sommige gevallen wordt er software gemaakt waar je oorspronlijke voor moet betalen. Alleen voor wat voor reden dan ook besluit het bedrijf de software opensource te maken. Op dit moment moet je nog steeds voor de originele software betalen.

Er is alleen nu een aparte scenario naar voorgekomen. Aangezien er een MIT licentie op het project kan je er alles mee doen. Je mag zelf een nieuwe versie uitbrengen van het project free of charge. Met dit krijg je dus een opensource project die betaald is en gratis is. De ja en nee kant van de vraag

Gratis is een deel van opensource maar niet de definitie opzicht. Zoals je zag kan je betaalde opensource software hebben. Op den duur kunnen mensen hier een gratis versie van maken en releasen. Hands down beide gratis en betaalde opensource software.

In elk geval is het bijdragen aan een open source project gratis. Het hangt af van de opensource project owner of het een betaald of gratis project is. Een argument kan zijn: Je kan een kopie maken van de betaalde software en het dan gratis maken. Je hebt dan opensource software die gratis. Alleen het beginpunt van het opensource project is nog steeds een betaalde versie.

### Opensource definitie: Ik neem wat en geef wat terug

Persoonlijk heb ik ook de definitie gezien van: In opensource neem je wat van het project en geef je terug aan het project. Vaak zag ik dit in de Linux community voorbij komen. Het is meer een leef regel dan een daadwerlijk regel. Ook kwam dit steeds meer in mijn eigen hobby branch na een discussie tussen OBS en Streamlabs OBS.

#### Open Brodcast Software (OBS) en Streamlabs OBS

Een gevolgde discussie in het open source gebied was tussen OBS en Streamlabs OBS. OBS is zoals de naam zegt een Open source project voor brodcasting software. Gericht op het ondersteunen van content-creators. Streamlabs OBS heeft hun software hier van afgebouwd en er een logo aangehangen.

De drama begon eigenlijk toen de tweet van Light stream volgde. Hierin was het duidelijke dat Streamlabs hun design had gestolen. Dit heeft ervoor gezorgd dat er meer mensen hun verhaal gingen delen waaronder OBS.

Obs had duidelijke aangegeven aan Streamlabs OBS om niet de naam OBS te gebruiken. Dit leidde tot een aantal gesprekken en discussies tussen Jim met het OBS team en Streamlabs OBS. Uiteindelijke is het uitgepraat maar het leidde tot de genoemde definitie discussies in de community.

Een mooi voorbeeld is Linux. Linux is een opensource os project te vinden op github. Je kan een heleboel versies vinden van Linux. Iedere weer een aftakking van het originele project. Wat je hier ziet is dat iedere aftakking iets op zijn eigen manier toevoegt. Alleen het wordt vaak ook terug gebracht naar het originele Linux project. Zo dat andere leden er ook gebruik van kunnen maken in het project.

Dat was het probleem met streamlabs. Er zouden nog tools worden ontwikkeld die juist OBS en streamers zouden kunnen helpen in het bijdragen van content-creation. Natuurlijk was Streamlabs opensource alleen hoe merge je tools back en forth.

Hierin is waar de leefregel eigenlijk vandaag kom. Ik neem wat en geef wat terug. Het is geen definitie van opensource. Alleen het is wel vaak een uitdrukking die voor komt.

## Hoe werkt een opensource project

Eigenlijk heb ik opensource al deels uitgelegd in het algemeen. Alleen nou lijkt het me beter om je mee te nemen. Op tocht als een developer die voor het eerst een contributie maakt aan een opensource project. Laten we zeggen dat Firefox een issues heeft met het inladen van gifs. De gifs laden alleen maar half op het scherm.

Voor we beginnen, het klopt Mozilla Firefox is een open source browser. Je komt op de website waar het opensource project staat en, ziet de eerder benoemde issue. Je ziet ook dat er nog geen branch is aangemaakt voor de issue. Na het lezen van de Mozilla code of conduct besluit je bij te dragen aan het project.

Als eerste forken we de repo. Dit betekent dat we een copy maken van de repo. Hiermee maken we een veilige omgeving waar we alles kunnen doen wat we willen. Mochten we een issue fixen die iets breekt in de originele repo dan is er altijd een backup. Vandaar de keuze voor het forken van de repo.

Nou je hebt je branch aangemaakt met Issue(Number)\_Name. Even tussendoor, ik moet zeggen dat ik niks over branch naming had gevonden in de code of conduct. Nadat je de code in je branch hebt gedownload en getest heeft of alles werkt. Dan begin je met inspecteren van de files. Tenslotte moet de issue met gif loading ergens zitten right?

Nu begin het plan. Namelijk om de issues op te lossen moet je coderen. Zo je denkt laat ik de gif site tenor is openen. Op het moment heb je tenor open op je home made Firefox browser. Na een grondige inspectie heb je een vermoedelijke locatie voor de bug gevonden.

Na een paar testjes merk je dat het probleem niet verholpen is. Sommige developers laten hier de changes gewoon zitten. Alleen wij zijn netjes en maken alles wat we hebben aangepast ongedaan. Tenslotte willen we niet iets breken in de source code

Nou uiteindelijke vind je de plek waar de issue zit en los je het probleem op. Great denk je dit moet ik laten weten en terug te pushen. Stap 1 is natuurlijk kijken of er niks breekt op je eigen repo. Vandaar dat we eerder een kopie hadden gemaakt. Breekt het nu dan heeft het geen effect op het originele draaiende product. De praktijk laat zien dat er verschillende manieren zijn waarop de projecten zich zelf testen. Slaagt dit dan komt er vaak een bouw plan om te kijken of het project nog bouwt.

Alles werkt, bouwt en run wat gebeurt er dan? Je hebt een issue opgelost in je eigen versie van Firefox. Gepusht naar je eigen copy repo en het werkt. Tijd om een pull request maken naar de Mozilla Firefox repo. Wat is een pull request? Een vraag om je gemaakte code te Mergen met de code van Mozilla Firefox. Natuurlijk wordt dit niet zomaar geaccepteerd. Stel je voor dat er een hidden bug in je code zit. Iets dat compleet Firefox breekt.

Elke opensource project heeft wel een aantal aangewezen maintainers die bijhouden wat er binnen komt aan soorten pull requesten. Deze mensen approven of disapproven je pull request. Laten we zeggen dat er een aantal vragen zijn van hun over je project. Je beantwoord ze en laat zien dat je geen kwaadaardige doel einde heb voor het project. Na dit overleggen de maintainers even en approven je pull request. Dat is het einde van een opensource loop

Nu heb je een contributie gemaakt aan een opensource project. Ook heb je gezien hoe een proces loopt voor een opensource project.

## Wat maakt een opensource project succesvol

Je zal vast denken nu: Wat maakt opensource nou echt zo succesvol? Helaas is er geen sleutel tot succes hier. Sommige projecten zullen slagen en sommige zullen eindigen in het water. Alleen succesvolle project zoals:

1. Google Android
2. Mozilla Firefox
3. Apache

Hebben toch wel wat dingen in het gemeen. Ik neem je mee door een aantal eigenschappen die voorkomen in de benoemde projecten. Laten we een paar details in kleinere stukjes bespreken en er dieper op in gaan.

### De instelling en houding

Laten we zeggen dat je de instelling hebt om een super project te maken. Je maakt alles tot op de punt netjes. De gedachte komt vat naar boven van: Ik heb hier zoveel interesse in. Andere developers zullen dat vast ook hebben

Hate to break it alleen meeste software developers zullen niet zomaar bijdragen aan je project. De meeste zitten waarschijnlijk op github of stackoverflow omdat ze een probleem hebben. Het kan ook zijn dat ze een probleem willen oplossen. Een postivie punt is het net houden van het project hou dat vast. Een net project trekt meer developers aan.

De praktijk laat zien dat developers niet zo snel zullen bijdragen aan een project. Vandaar dat ik toch wil zeggen hou de houding vast. Er zijn nog een aantal punten die zullen helpen. Laten we daar is naar kijken.

### Documentatie

In elke succesvol project dat je opzoekt kom je documentatie tegen. Een readme die uitlegt helder en duidelijke wat het project inhoudt. Dit bestand kan ook bevatten hoe je bijvoorbeeld kan bijdragen aan het project. Badges die bewijzen dat je het project kan runnen en bouwen als voorbeeld.

Een readme is vaak gelimiteerd tot wat het project inhoudt. Badges die bepaalde dingen kunnen bewijzen als voorbeeld: Project runt. CI Pipelines runt. Of bepaalde packages die werken. Vaak begint hier al het thema. **Duidelijkheid en begrijpelijke documentatie.** Dit is komt altijd voor in de succesvolle projecten.

Wat is duidelijke en begrijpelijke documentatie. Je moet het zo zien als je hebt een idee in je hoofd. Alleen als je het probeert uit te leggen en de persoon snapt er niks van. Documentatie of Visuele tekening kunnen hier helpen. Zeker als je een openbaar project wilt leiden. Het idee is om iedereen op een lijn te krijgen in een opensource project. Het zou jammer zijn als een toekomstige contributer weggaat omdat iets onduidelijke is.

Vaak komt de readme gepaard met een Code of Conduct. In sommige gevallen heb je een aparte Code of Ethics. Alleen vaak komt voor in de praktijk dat de Code of Ethics samen gaat met de Code of Conduct. Dit document is het regelboek van een opensource project. De regels zijn vaak gericht op hoe ga je met het project om en gedrag regels. Sommige bedrijven vinden het belangrijk die te scheiden. Vandaar dat sommige project het delen in een Code of Conduct en Ethics

Zoals eerder benoemd is de Code of Conduct het regelboek van je project. Er staan regels in van: Hoe ga ja om met andere mensen, Wat zijn de coding regels, Op wie geldt het en waar geldt het. Zo kan ik de hele lijst op noemen maar je krijgt het idee. Je kan zien waarom deze documenten vaak duidelijke en begrijpelijke worden geschreven. Reden hiervoor is zodat er geen onduidelijkheid komt of ruzie vanwege een verkeerds gedrags probleem.

### Een probleem

Van het een probleem naar het andere. Software komt altijd ergens vandaan. Developers hebben vaak ook het idee: Mijn project is world changing. Alleen dat is het probleem met opensource projecten. Voor ik verder ga moet ik zeggen: De praktijk laat die meer zien als een advies dan een daadwerlijk probleem. Ook al is het een advies misschien is het toch handig om te bekijken.

Met dat uit het zicht. Mensen zullen minder snel geïnteresseerd zijn in een project dat al bestaat op de markt. Het is een advies die best vaak wordt gegeven. Betekent dit dat je huidige idee die bijvoorbeeld je huisdieren automatische voert weg moet gooien? Nee natuurlijk niet der is kans dat het groots uitpakt. Alleen de werkelijkheid laat vaker zien dat er kans is dat het mis gaat dan slaagt.

### Quality

Het laatste punt dat ik wil bespreken is quality. Quality is vaak iets wat je terug ziet in succesvolle project. Hoe hoger de kwaliteit hoe beter het project gaat. Of bedriegt het oog hier. Om een idee te geven wil ik je een scenario geven om het goed in beeld te brengen.

Stel je voor je gaat naar de mediamarkt om een nieuwe koptelefoon te komen. Je ziet drie koptelefoons. Ze hebben allemaal dezelfde prijs. 40 euro. Ook draait op iedere koptelefoon hetzelfde nummer en staat het geluid op 70%. Nu komt natuurlijk de vraag welke koop je? Laat me de kenmerken opschrijven voor ze alle 3:

1. Deze koptelefoon zit goed op je oren. Alleen de muziek hapert een paar keer door het nummer. Het geluid is verder prima te horen voor de volume van 70%
2. Deze koptelefoon zit goed op je oren. Muziek is normaal te horen. Alleen het muziek klinkt te hard voor 70%.
3. Deze koptelefoon zit niet lekker op je oren. Muziek hapert af en toe. Verder is het geluid vrij goed voor 70%

Nu komt de vraag welke zou je kopen. Ik kan je nu vertellen dat je waarschijnlijk aan 2 zat te denken. Waarom vraag je af? Mensen tegenwoordig gaan vaker voor quality dan prijs. Zo zal je sneller een koptelefoon kiezen die langer mee gaat en wat duurder is, met een beter quality dan een goedkopere.

Waarom vertel ik dit verhaal eigenlijk? De reden dat mensen voor quality gaan is ook iets wat bij opensource projecten hoort. Stel je voor je hebt een 40u werkweek baan als software developer. Je wilt bijdragen in je privé leven aan een project op het internet.

De vraag komt: Zal jij je tijd besteden aan een project dat een mess is?. Groot en deel van de mensheid zegt nee. Paar dit met de documentatie over het project. Dan heb je ongeveer 20 seconde om de developer over te halen. Als de readme al een mess is dan stop het daar. Dan houdt het al op en ben je weer een contributer kwijt.

Laten we zeggen de readme was een mess maar genoeg om de developer over te halen. Laten we zeggen de developer gaat kijken in de code. 2 dingen kunnen gebeuren nu: De developer bedenkt zich en sluit af van het project of De developer wilt het een kans geven sluit zich aan.

Quality is een belangrijke aspect en zal altijd na geleefd moeten worden. Doe je het niet dan is de kans groot dat je project faalt.

## Wat zijn de gevaren van een opensource project

Gevaren elke project heeft ze. Alleen wat brengt opensource met zich mee. Laten we dit stukje delen in 2 stukken. Zijn er gevaren in opensource. Zou je deze gevaren kunnen voorkomen als ze er zijn.

### Wat zijn de gevaren van opensource

Zo de main vraag zijn er gevaren in opensource. Om te beginnen moet je in gedachten houden dat opensource projecten toegankelijke zijn tot iedereen die dat wilt. Ik denk dat het handig te beginnen is met het volgende onderwerp

#### Security

Het grootste probleem met opensource is security. Vaak zijn deze projecten community driven. Vaak wordt er door de community niet gekeken naar security. Meeste software teams hebben een expert die kijkt naar de security. Je hebt ook teams die het uitbesteden bij bedrijven die het regelen.

#### Vulnerabiltes

Een bekend gevaar is dat opensource toegankelijk voor iedereen is. Zo stel je voor een contact formulier dat de mogelijkheid heef tot een XSS attack. De bug blijft onopgemerkt en wordt gepushte naar een master release. Een bad actor of te wel vaak een hacker met kwade bedoelingen besluit een kijkje te nemen in de open source code.

Na grondige te kijken vind de hacker de XSS bug. Wel nou komt het probleem met de bad hacker. Ze hebben toegang en kunnen gebruik maken van het gat in de code. Data kan worden gestolen, De hacker kan toegaan krijgen tot pc`s, De hacker kan data monitoren en zien wat gebruiken doen en etc.. Zo kan je zien dat opensource projecten niet altijd veilig zijn.

#### Long term sustainability en warranty

2 dingen die onder een categorie vallen in dit geval. De praktijk laat zien dat opensource projecten vaak worden gemaakt en beheert door de community` s eromheen. Het probleem hier is. Als mensen interesse verliezen en weggaan. Krijg je een project dat outdated

Achtergelaten projecten worden vaak niet meer beheert. Dit kan leiden tot outdated issues met het project. Op een moment krijg je dus een project met outdated issues. Gevolg hiervan kan zijn dat je meer risico’s in het project krijgt. Wat weer leidt tot de eerdere benoemde gevaren bij een opensource project.

Zoals je ziet zijn eigenlijk de mensen rondom het project het grootste gevaar. Een project dat actief is en veel aan wordt onderhouden heeft minder risico`s. Dit in vergelijking tot een project met een community die langzaam aan het afzakken is.

### Kan je de gevaren uit de vorige sub-vraag voorkomen?

Er zijn manieren om de gevaren te voorkomen. Alleen het is vaak niet 100% werkende weg. Sommige projecten hebben de gevaren nou eenmaal en werken. Andere projecten kunnen geen gevaren hebben en niet werken. Het begin is een actieve community. Een actieve community leidt vaak tot minder kans op gevaren.

Als je de leider van een opensource project ben kan je kiezen om security uit te besteden aan een ander bedrijf. Hiermee kan jezelf dekken op het security geval. Alleen hoe doe je dat als je niet de leider ben. Dan hangt het compleet af van de persoon die het opensource project bezit of beheert.

Zoals eerder gezegd verhelpt een actieve community rondom het project de meeste dingen. Een actieve community zal ervoor zorgen dat de code up to date blijft. Ook kijkt de community naar gevallen waar er een leak of Vulnerability zit.

Met deze tips zou je de meeste voorkomende gevaren uit de praktijk voorkomen. Alleen het leert dat niet alles voorkom baar is. Tenslotte weetje nooit wat de dag zal brengen of, wie de volgende is die je opensource project bekijkt.

# Conclusie

Zoals je kan zien is opensource toch wat breder dan code online en klaar. Er komen meer zaken kijken bij opensource: Van wat voor idee wil ik open source maken, Hoe pak ik dit aan, Gevaren en oplossingen. Opensource kan eenvoudig worden opgepakt. Alleen is het soms lastiger dan gedacht omdat je niet weet waar je moet beginnen. Laten we is kijken naar successen van andere projecten.

Een aantal voorkomende successen die gezien zijn in opensource projecten. Waarvan ze nog steeds draaien en actief zijn. De volgende eigenschappen kwamen voor in elk project:

1. Een nieuw of interessant idee voor op de markt
2. Duidelijke en Heldere Documentatie
3. Actieve community rond om het project
4. In sommige gevallen zagen we ook een visuele demo in de documentatie

Het zijn factoren die niet bepalen of je succesvol opensource project hebt. Het zijn wel factoren die voorkomen in succesvolle projecten.

Om te beginnen moet je komen met een idee dat interessant is of nieuw. Zoals eerder aangegeven in het document zijn de meeste contributers gewoon mensen die 36-40u werken in de week. Win je de interesse niet in hun eerste blik? Dan is het al vaak over.

Zijn deze stappen een sleutel tot succes? Nee in dit geval heeft een opensource project geen sleutel tot succes. De boven benoemde stappen zijn factoren die voorkomen in een project met succes. Alleen in het onderzoek was ik ook een aantal project tegen gekomen met deze kenmerken waar het is afgelopen in een slecht teken.

Een opensource project succesvol maken is eigenlijk zoiets als twee dobbelstenen rollen en hopen dat ze dezelfde nummer zijn. Je kan een pracht project opzetten. De beste documentatie hebben en een top idee. Alleen je bent ook afhankelijk van contributers. Praktijk laat namelijk zien dat het project echt anders niet van de grond komt.

Tenslotte wil ik het onderzoek afsluiten met een plan voor het opzetten voor je eigen project. Het opzetten van een succesvol opensource project is lastig. Dus hieronder kan je nog een guiding light vinden. Zoals het onderzoek aantoont is het niet altijd makkelijke te beginnen.

# Plan voor opzet

## Vooraf

Om te beginnen heb je natuurlijk een idee nodig. Dit is helemaal op tot jou. Ik raad je ook aan de starting-a-project guide te lezen in de bronnen lijst. Natuurlijk komt ook de vraag ik heb al een idee wat nu Elise? Ik raad je aan te kijken of er geen zelfde soort projecten al zijn.

In geval van ja raad ik aan te kijken hoever er verschil in zit. Misschien is het namelijk beter te contributen aan hun project dan het maken van je eigen. Als voorbeeld laten we zeggen dat er geen andere soort gelijken projecten zijn.

Voor je begint met opzetten een tip. Dit is een fout die ik heb gemaakt een mee wil geven. Waarschijnlijk gaan er al ideeën van branches X, Y, Z, U door je hoofd heen. Dat is een goede gedachte maar hou het vast. Voor opzet raad ik aan met een branch the beginnen dit scheelt heel veel tijd.

## Waar te beginnen.

Natuurlijk moet je code ergens online komen te staan. Persoonlijk gebruik ik veel github maar er zijn genoeg alternatieven zoals: Gitlab, Git, Azure dev ops, etc. Dus de eerst vraag is waar wil je project hosten onderzoek dat en kijk wat voordelen en nadelen er zijn voor iedere platform.

Een tip voor de volgende hoek is Licenties. Uit me hoofd durf ik niet te zeggen hoe het zit voor andere platformen maar, Github heeft het zo dat zonder licentie in de repo gelden default copyright laws.

Vandaar het idee ga is op onderzoek kijken hoe dat werkt in je platform. Bij github zou dit betekenen dat niemand iets met me code mag doen zonder mijn permissie. Vandaar de licentie keuze. Persoonlijk zeg ik dit met de reden dat er ook heel wat verschillende soorten opensource licenties zijn. Dus kijk is welke het beste bij je idee past. MIT is de meeste voorkomende opensource licentie en geeft je misschien een start om ergens te kijken.

## Documentatie

Nou je hebt je locatie en licentie. Wat is er nu te doen. 2 documenten aan te maken om te raden. Readme.md en een code of conduct. Ik zal kort uitleggen wat elke van de 2 documenten is. Ook zal ik zeggen wat hun taak is.

### Readme.md

Je moet de readme zien als je informatie brochure naar het project. Het is de entry van je code plek. Meestal zijn deze bestand ook direct leesbaar voor het publiek. Als dit bestand niet duidelijke en helder is verlies je 90% van het publiek. De andere 10% zal het nog wel een gok willen geven.

### Code of Conduct

Zie een code of conduct als het heilige boek van je opensource project. In dit document staat hoe je moet gedragen. Ook staat erin wat er gebeurt op het moment dat iemand zich niet gedraagt in het project.

Wat je vaak ziet voorbij komen is dat die document 2 versies heeft. Een code of ethics en een code of conduct. Met de code of ethics vaak gericht op hoe je moet denken en wat juist goed en slecht is. Als voorbeeld denk aan het opslaan van data wat is ene goede manier om dat te doen en wat is fout.

Code of conduct richt zicht meer op de common policys zoals gedrag regels, Commnuicatie regels en etc. Sommige gevallen zie je ook coding rules in het code of Conduct document. In de meeste gevallen is er een los bestand waarin de coding guidelines staan.

### How to contribute document

Het laatste document dat voorbij komt in opensource projecten is een bestand met how to contribute. Dit document leegt uit hoe je bijdraagt in het project. Vaak begint dit document met te zeggen van:

Read the code of conduct and the coding guidelines

Make sure to follow the next steps to start:

Dit zijn een paar onderwerpen die vaak voorkomen. Zoals het zegt is het een document waarin wordt uitgelegd hoe je bijdraagt aan het opensource project.

## Wat nu?

Nu heb je een idee, documentatie opgezet, een plek voor je project. Ik kan je horen denken wat komt nu? Niet veel meer eigenlijk het opzetten is zo goed als af. Nu komt het gedeelte eigenlijk dat een opensource project maakt of faalt.

Bouwen van een community. Hierin zijn niet echt tips te geven. Community komen en gaan en hangen af van de mensen die er tijd in willen steken. Nog als een laatste tip:

Ik raad aan nogmaals de eerste bron te checken van een opensource guide. Hun hebben ook een plan met uitleg erbij en waarom zo iets werkt. Dit gepaard met informatie van mensen uit het veld geeft aardig wat inzicht in opensource. Als laatste hun hebben ook een checkmark box waarin je bij kan houden wat je al hebt en wat niet.

# Bronnen

Hieronder is een lijst te vinden met bronnen die ik heb gebruikt:

* <https://opensource.guide/nl/starting-a-project/>
* <https://www.linuxfoundation.org/tools/starting-an-open-source-project/>
* <https://opensource.com/resources/what-open-source>
* <https://www.techtarget.com/whatis/definition/open-source>
* <https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/licensing-a-repository>
* <https://www.dexerto.com/entertainment/obs-streamlabs-reach-agreement-plagiarism-controversy-1727742/>
* <https://www.reddit.com/r/OutOfTheLoop/comments/qw5e2u/whats_going_on_with_streamlabsobslightstream/>
* <https://twitter.com/Lightstream/status/1460709404609757185?ref_src=twsrc%5Etfw%7Ctwcamp%5Etweetembed%7Ctwterm%5E1460709404609757185%7Ctwgr%5Ec2a4d5b7fb1c88086c7c5dc4e422e267a44552ed%7Ctwcon%5Es1_&ref_url=https%3A%2F%2Fwww.dexerto.com%2Fentertainment%2Fobs-streamlabs-reach-agreement-plagiarism-controversy-1727742%2F>
* <https://skillcrush.com/blog/what-is-open-source/>
* <https://dmitripavlutin.com/how-to-make-your-open-source-project-successful/>
* <https://devblog.xero.com/three-tips-on-running-a-successful-open-source-software-project-ce3028ed0231>
* <https://durabilitymatters.com/quality-over-price/>
* <https://lessbutluxe.com/quality-over-quantity/>
* <https://www.datto.com/blog/7-risks-posed-by-open-source-software-and-how-to-defend-yourself>
* <https://snyk.io/learn/risks-of-open-source-software/>