ICT Architecture

Groepsopdracht

Analysedocument

Saadi Mohamed, van der Avoirt Jef,

Souleymane Mohammed, Santens Stef

2ITSOF1

Titularis: Van Hansewijck Dries Academiejaar 2021-2022  
Titularis: Overdulve Kristof 1ste semester

Inhoud

[Inleiding 3](#_heading=h.gjdgxs)

[Samenvatting 4](#_heading=h.30j0zll)

[Dataflowdiagram 5](#_heading=h.1fob9te)

[Uploadproces: 5](#_heading=h.3znysh7)

[Downloadproces: 5](#_heading=h.2et92p0)

[Architectuur 6](#_heading=h.tyjcwt)

[Verantwoording keuzes 7](#_heading=h.3dy6vkm)

[Amazon S3 bucket 7](#_heading=h.1t3h5sf)

[Amazon EC2 7](#_heading=h.4d34og8)

[Amazon Cognito 9](#_heading=h.2s8eyo1)

[Amazon RDS 9](#_heading=h.17dp8vu)

[Amazon Lambda 9](#_heading=h.3rdcrjn)

[LifeCycle 10](#_heading=h.26in1rg)

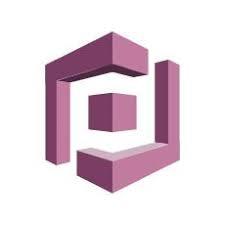
# Inleiding

Met dit analysedocument willen we een beter beeld schetsen over hoe ons project in elkaar zit en wat we allemaal gebruikt hebben om ons project tot een goed einde te brengen.

Het doel van onze applicatie is het maken van een file sharing platform waarbij gebruikers zich kunnen registreren/aanmelden om bestanden te uploaden en downloaden. We maken gebruik van Amazon Web Services (AWS) wat ervoor zorgt dat we veel minder tijd en moeite moeten steken in de configuratie van alle services en onze focus meer op onze applicatie kunnen richten.

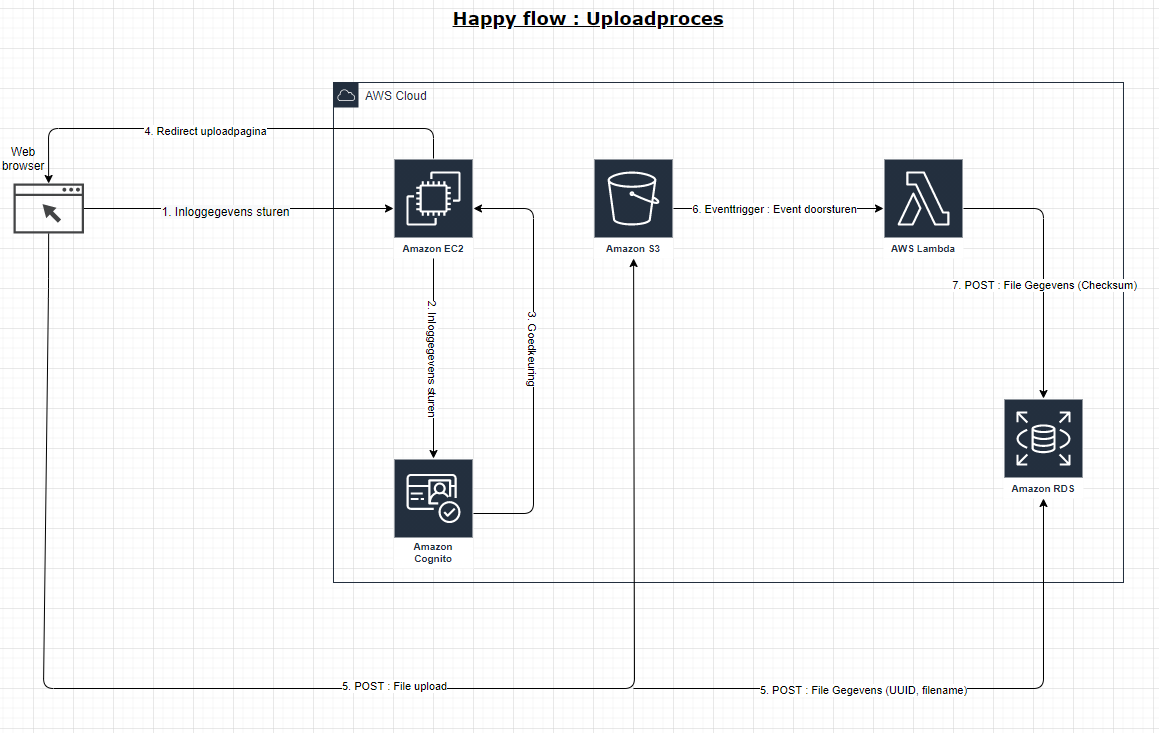
# Samenvatting

Voor het vak ICT architecture zal onze groep een systeem bouwen met als doel het onderling delen van bestanden (downloaden en uploaden van bestanden). We gaan grotendeels gebruikmaken van de Amazon Web Services (AWS) om dit te kunnen verwezenlijken. AWS heeft heel wat services die het ons stukken gemakkelijker zullen maken. Een voorbeeld daarvan is onze webserver die we zullen aanbieden via Amazon EC2 omdat deze service een basisvereiste is om de AWS cloud te begrijpen. Nu dat we een webserver hebben, zullen we ook iets nodig hebben om de bestanden te kunnen opslaan. Hiervoor maken we gebruik van Amazon S3. Amazon EC2 en S3 kunnen communiceren met elkaar door de security groups juist in te stellen. We maken daarnaast ook gebruik van een SDK om de implementatie van de webserver uit te breiden. Voor de authenticatie (user management) gebruiken we Amazon Cognito. Deze service zal ervoor zorgen dat gebruikers kunnen registreren en inloggen zodat enkel geauthenticeerde gebruikers kunnen downloaden en uploaden. Een database systeem is zeer cruciaal in ons project. Hiervoor maken wij gebruik van Amazon RDS. RDS zal alle bestandsnamen, UUID’s en checksums bijhouden. Ons systeem zal ook steeds controleren of er een bestand langer dan 24 uur in de S3 bucket zit. Als dit het geval is, moet dit bestand verwijderd worden. Hiervoor kunnen we het beste een lifeCycle op de bestanden in de S3 bucket zetten dat automatisch na 24 uur de bestanden uit de bucket zal verwijderen. Een Amazon Lambda functie zal getriggerd worden indien een bestand uit de S3 bucket verwijdert wordt, en zal de bijhorende bestandsinformatie uit de RDS-database verwijderen. Als we dit allemaal correct en werkend krijgen, kunnen we spreken over een geslaagde groepsopdracht. Al de services die we hier gebruiken zullen dan samen een functionele ICT-architectuur vormen. Afbeelding met tekst, illustratie

Automatisch gegenereerde beschrijving

# Dataflowdiagram

## Uploadproces:

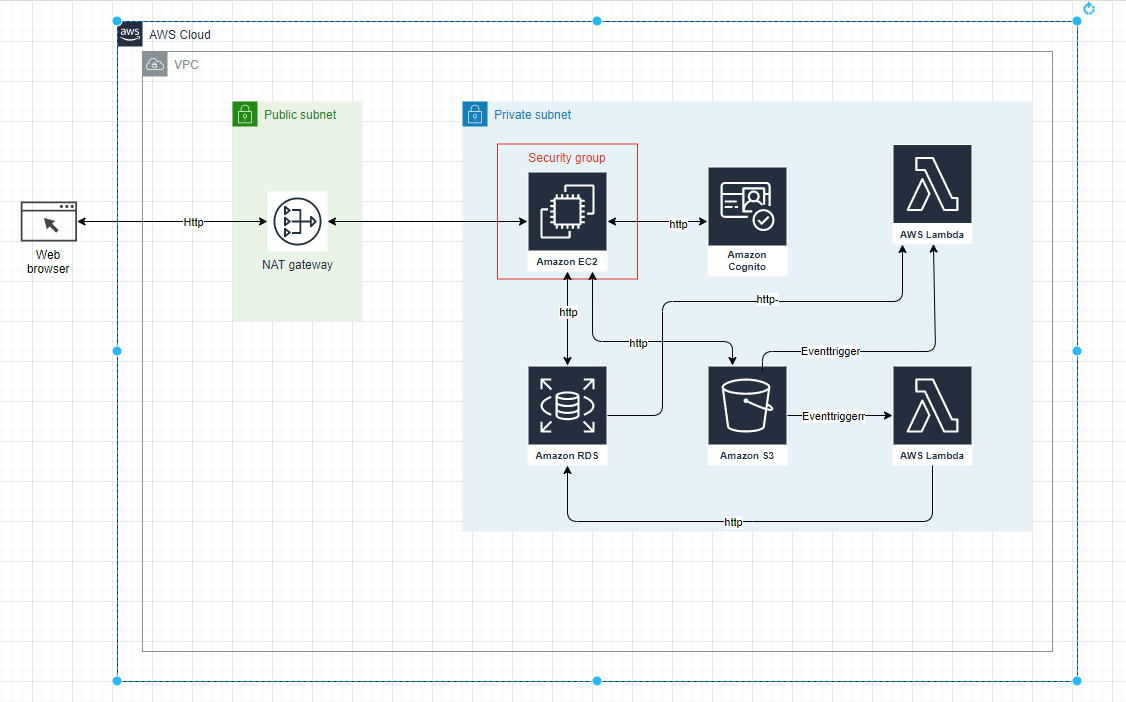


## Downloadproces:

Afbeelding met tekst, lucht, dag

Automatisch gegenereerde beschrijving

# Architectuur



Het doel van het project is een applicatie te maken waarop je bestanden kan uploaden en downloaden. Dit doel hebben we gerealiseerd door als eerste een user aan te maken. De user is zelf aan te maken door een e-mailadres en password door te geven en te klikken op de registreer knop. Als dit gebeurt, zal de systeembeheerder deze user eerst nog moeten goedkeuren. Eenmaal dat dit gebeurd is, zal de gebruiker kunnen inloggen en dus ook kunnen uploaden en downloaden. Dit gebeurt doordat we gebruikmaken van Amazon Cognito.

Upload

Als het inloggen gelukt is, zal de gebruiker worden doorgestuurd naar de uploadpagina. Hier zal de gebruiker een knop “uploaden” zien. Als de gebruiker hierop klikt, zal hij een bestand kunnen selecteren met behulp van een verkenner-venster. De gebruiker zal een melding krijgen op zijn scherm als dit succesvol is gelukt. Deze melding zal de UUID bevatten en de naam van de geüploade file. Deze file zal in de S3 bucket bewaard worden met de UUID als naam. Dit doen wij zodat er geen fouten kunnen gebeuren met 2 files die dezelfde naam hebben. De filenaam zal samen met de UUID en een checksum in de RDS bewaard worden. De checksum verkrijgen wij door een lambda functie die getriggerd wordt als er een bestand in de S3 wordt geüpload. Uit het meegegeven event haalt de lambda functie dan de checksum.

Download

Als de gebruiker een file wil downloaden van de applicatie, zal hij in de URL de UUID moeten meegeven. Hierdoor kan hij de UUID delen met andere gebruikers waardoor ook zij dat specifieke bestand kunnen downloaden. Eenmaal de UUID is meegegeven via de URL, zal deze UUID worden doorgestuurd naar de EC2. Deze zal de UUID op zijn beurt doorsturen naar de RDS en S3 bucket. Met de UUID halen wij de file uit de S3 bucket en de filename uit de RDS. Deze file en filename geven wij terug aan de user.

Clean Up

Om de opslagkosten van de S3 zo laag mogelijk te houden, verwijderen wij alle bestanden die langer dan 24 uur op de S3 staan. Hiervoor gebruiken wij een Lifecycle op de S3 files.

De Lifecycle zorgt ervoor dat files die 24 uur op de S3 bucket staan automatisch worden verwijderd. De Lifecycle triggert ook een Lambda functie die de informatie over het net verwijderde bestand uit de RDS-database verwijdert.

# Verantwoording keuzes

## Amazon S3 bucket

Wij kozen ervoor om voor de cloud storage van onze bestanden gebruik te maken van een S3 bucket. S3 staat voor Amazon simple storage service en is een object level storage. Dat wil zeggen dat het objecten opslaat zoals files. Dit deden wij omdat onze bestanden kleiner zijn dan 3 GB en omdat S3 gemakkelijker te gebruiken is in vergelijking met Amazon EBS.

Omdat wij niets buitengewoon doen, is de Amazon S3 standaardversie goed genoeg. Dat is ook de goedkoopste versie.

We communiceren met onze S3 bucket met gebruik van een SDK wat ervoor zorgt dat wij gemakkelijker kunnen coderen in andere talen namelijk node.js/javascript.

## Amazon EC2

Wij hebben EC2 gekozen omdat dit de meest geschikte compute service is op basis van het doel van ons project, namelijk het downloaden en uploaden van files.

EC2 oftewel Amazon Elastic Compute Cloud is een schaalbare virtuele machine. Dat wil zeggen dat afhankelijk van de load, EC2 meer of minder resources zal vragen en dus de kost zo laag mogelijk houdt. EC2 is een IaaS oplossing, hierdoor hebben wij dus volledige controle over onze instantie.

Bij het aanmaken van een EC2 instantie hebben we enkele keuzes moeten maken. Namelijk AMI, Instantie Type, IAM Role, Data, Opslagopties Tags, security groups, Key pair.

AMI:

* Als AMI hebben we gewoon de default en gratis versie (Amazon Linux) gekozen, omdat dit voldoet aan onze projectdoelen en omdat het gratis is.

Instantie Type:

* Wij hebben hier de t2.micro instantie type gekozen omdat dit de gratis versie was en wij ook niet beter dan dit type nodig zullen hebben voor ons project.

Opslagopties:

* Om onze operating system op te slagen in onze root hebben we de default SSD gekozen, omdat dit voldoet aan de vereisten voor ons project.

Security groups:

* Bij de security groups hebben we voor de zekerheid ervoor gezorgd dat iedereen kan connecteren met de EC2 door het default source adres te gebruiken namelijk 0.0.0.0/0. Amazon Cognito zorgt ervoor dat niet-geautoriseerde gebruikers geen toegang (username/password) krijgen tot onze website. Hier gaan wij zometeen verder op in.

Afbeelding met tafel

Automatisch gegenereerde beschrijving

## Amazon Cognito

Zoals eerder aangekaart zal Amazon Cognito ervoor zorgen dat gebruikers zich in onze applicatie kunnen authentiseren met een username en password. Dit is zeer makkelijk en handig om te implementeren doordat Amazon Cognito zo goed als alles doet voor ons. Amazon Cognito heeft ook veel extra functionaliteiten die we bij dit project niet allemaal gebruikt hebben maar wel interessant kunnen zijn.

Wij hebben gebruik gemaakt van User Pools waardoor gebruikers zich kunnen registreren en later na goedkeuring (van de systeembeheerder) kunnen aanmelden op onze applicatie.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

## Amazon RDS

Als opslag gebruiken we Amazon Relational Database Service, beter bekend als RDS. Als database engine voor RDS hebben wij gekozen voor MySQL, maar dit heeft geen grote impact op ons project. Er kan zelfs een NoSQL database engine gebruikt worden. Hier slagen wij dan de filenaam, UUID en checksum op.

## Amazon Lambda

Amazon Lambda zorgt ervoor dat je code serverless kunt uitvoeren. Deze code zal uitgevoerd worden wanneer er een event getriggerd is. Dit kan verschillende oorzaken hebben zoals een tijdstip (routine), voor-gedefinieerde trigger (van Amazon zelf) en een zelf-gedefinieerde trigger (zelfgemaakte trigger). De code die we gebruiken (uploaden) voor de Lambda kan in verschillende talen geschreven worden (Java, Go, Powershell, Node.js, C#, Python en Ruby).

## LifeCycle

Wij hebben gekozen voor Lifecycle omwille van twee redenen. Als eerste is het implementeren van de Lifecycle veel gemakkelijker, aangezien de S3 Lifecycle er automatisch voor zal zorgen dat bestanden na 24u worden verwijderd, i.p.v. dat we dit zelf moeten doen.

Maar misschien nog belangrijker is dat de Lifecycle veel accurater is dan een trigger op een bepaald moment. Dit komt doordat de Lifecycle het bestand na 24 uur automatisch zal verwijderen. Als wij zouden werken via een trigger op een bepaald tijdstip, dan zouden de bestanden pas verwijderd worden op dat bepaald tijdstip.