ANALYSE DOCUMENT AWS

Groep 1

Alan Dhooge,

Ignace Shoeib,

Kaine Van Geffen,

Massin Lajlufi

AP Hogeschool

ICT Architecture

Inhoudstafel

[1 – SAMENVATTING 2](#_Toc90491255)

[2 – SITUATIE-TO-BE 3](#_Toc90491256)

[3 – ARCHITECTUUR 6](#_Toc90491257)

[4 - VERANTWOORDING KEUZES 8](#_Toc90491258)

# 1 – SAMENVATTING

De gebruikers moeten bestanden kunnen opslaan en opvragen van een server.

Bij deze zal de server een UUID genereren voor elk bestand die de gebruiker upload. Als de gebruiker dan deze bestand heeft geüpload, dan zal de server een checksum moeten uitvoeren. Een checksum is een som gebaseerd op het aantal bits die een bestand heeft, de reden dat we dan een checksum willen is om errors te voorkomen.

Aangezien we gebruikers moeten aanmaken, zullen we gebruik maken van **Amazon Cognito**

Amazon Cognito is verantwoordelijk voor het mogelijk maken om gebruikers toe te voegen en te kunnen authentiseren via onze API. Het is ook mogelijk om deze gebruikers eigen rollen te geven.

Omdat de server een UUID genereert voor elk bestand die de gebruiker upload, zullen we deze UUID van de bestanden moeten bewaren in een Amazon Relational Database. We kunnen met deze UUID een checksum uitvoeren om te zien of deze bestand foutloos is overgebracht.

Als een gebruiker een bestand heeft geüpload, zal deze bestand(en) maar maximaal 24 uur op de server blijven. Het kost veel tijd om dit handmatig te doen, en is dus hierdoor ook niet efficiënt. Om dit te automatiseren zullen we gebruik maken van Amazon Lambda.Amazon Lambda zal op basis van onze zelf-ingestelde momenten automatisch onze S3 Bucket leegmaken.

# 2 – SITUATIE-TO-BE

**We zullen gebruik maken van de volgende Amazon Web Services:**

**Amazon S3**

**Amazon S3 is een service om objecten op te slaan. We gebruiker deze service omdat de bestanden die de gebruiker upload ergens moet bewaard worden.**

**Deze service heeft ook verbinding met een EC2 instantie om bestanden te delen met andere gebruikers.**

**Amazon EC2**

De backend van de applicatie zal zich hier bevinden. Amazon EC2 is verantwoordelijk voor het contact met het front-end van de applicatie en de S3 bucket. Deze service zorgt er ook voor dat je applicatie schaalbaar is.

**Amazon Cognito**

Amazon Cognito is verantwoordelijk voor het authenticatie van gebruikers, en toegang voorzien tussen applicaties en Amazon.

We zullen gebruik maken van deze service om ervoor te zorgen dat enkel geauthentiseerde gebruikers bestanden kunnen uploaden.

Om ervoor te zorgen dat geauthentiseerde gebruikers onderscheden kunnen worden, zullen we gebruik maken van een User Pool waarin de gebruiker zich kan registreren en inloggen via het front-end.

Alleen geauthentiseerde gebruikers kunnen uploaden en downloaden.

**Amazon RDS**

We willen dat de informatie van de bestanden ook apart wordt bewaard in een database. Zodra een gebruiker een bestand uploadt zullen de gegevens van het geüpload bestand, namelijk de FileName, UUID en de creatie datum worden opgeslagen in de database. Onze database maakt gebruik van een query, gemaakt met het programmeertaal mysql om bestanden op te slaan. Elk object heeft ook een auto incremented id.

**Amazon Lambda**

Amazon Lambda is een manier om automatisch een taak te laten uitvoeren zonder manuele interactie van de gebruiker.

We maken gebruik van Amazon Lambda om ervoor te zorgen dat alle objecten (bestanden) binnen onze Amazon S3 bucket automatisch na 24 uur verwijderd worden. We willen namelijk, volgens de opdracht, dat bestanden niet langer dan 1 op ons systeem bewaard worden.

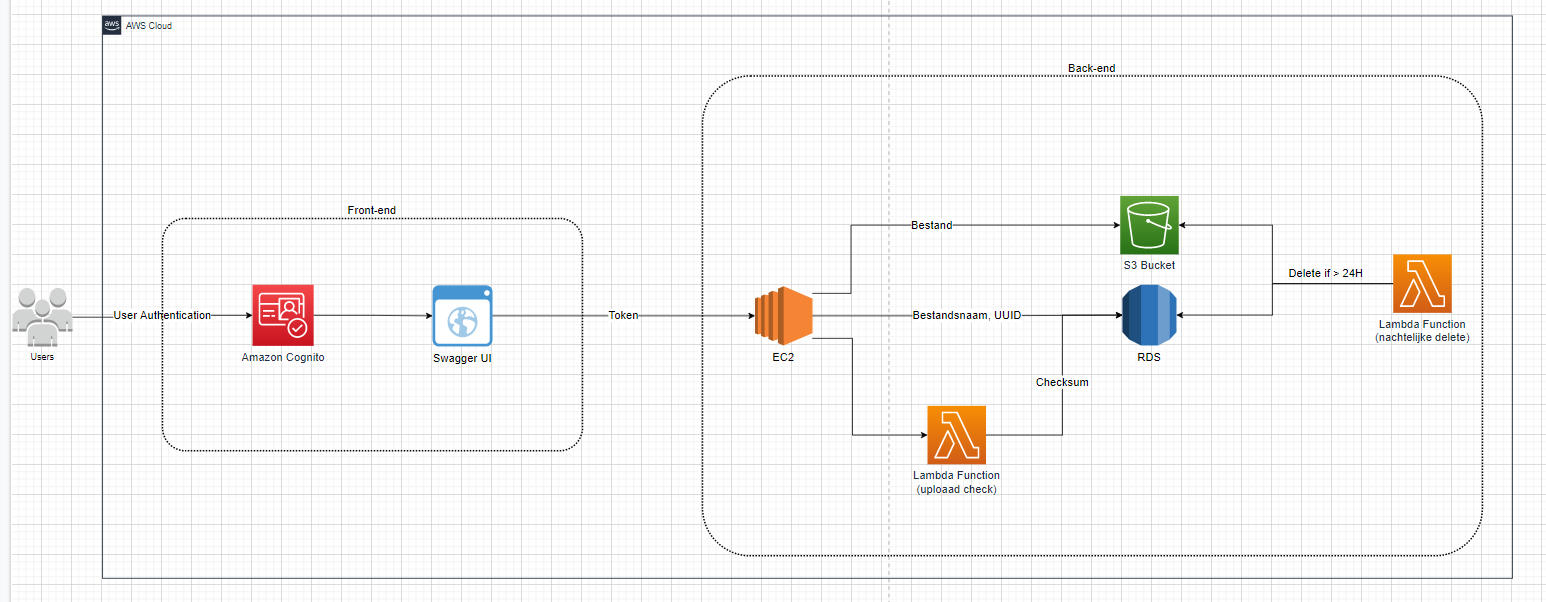
Hetzelfde moet gebeuren in onze Mysql AWS Relational Database, Als de toegevoegde record van ons bestand ouder is dan 24 uur, moet deze record verwijderd worden van de database.

**Interface/ API**

We maken gebruik van de Swagger API als front-end. Met deze API kun je eventuele informatie per controller ingeven.

Deze informatie wordt verwerkt in de code en naar de corresponderende Service gestuurd.

# 3 – ARCHITECTUUR

****

**Registreren/inloggen:**

De user gebruikt een email en passwoord om te registreren. Deze data worden dan naar de Amazon Cognito service verstuurd door een SignUpRequest te creëren. Hierna wordt dit opgeslagen en na het inloggen wordt de data ook opgeslagen in cookies.

**UI:**Voor onze UI maken we gebruiken van de Swagger UI bij asp.net met swashbuckle. Dit zorgt dat we een duidelijke en complete UI hebben voor al onze gebruikswaardigheden.   
  
**Upload:**

Bij het uploaden van een bestand zal er een Lambda functie worden uitgevoerd die een checksum uitvoert en de output dan opslaat samen met de bestandsnaam en UUID in RDS. Het bestand zelf wordt dan verstuurd naar de S3 bucket samen met de UUID.

**Download:**

Een request wordt verstuurd naar ons programma voor een item te kunnen downloaden hierbij wordt een kopie van het gekozen item terug verstuurd naar de user.

**Lambda (upload check):**

Bij het uploaden zal deze Lambda functie starten een checksum aanmaken en dan deze opslaan.

**Lambda (nachtelijke delete):**  
Elke nacht om 3u zal een automatische check worden uitgevoerd die objecten dat ouder zijn dan 24u verwijdert uit zowel de RDS als de S3 bucket.

**Opslag (S3 Bucket):**De S3 Bucket is een eenvoudige opslag service beschikbaar gesteld door Amazon. Hier zullen we al onze bestanden opslaan. De bestanden worden ook beschikbaar gesteld om gedownload te kunnen worden met de UIID van het bestand.

**Opslag (RDS):**

In de RDS wordt de data van bestanden opgeslagen. Deze data zijn de bestandsnamen, UUID’s en de checksum gegenereerd door Lambda.

# 4 - VERANTWOORDING KEUZES

**Programmeertaal**

We hebben besloten om met C# te werken aangezien we meer kennis en ervaring hebben met deze programmeertaal.

Om onze Lambda functie te programmeren hebben we gebruik gemaakt van Javascript. Deze is beter compatibel met de editor die beschikbaar is in AWS zelf, en werkt ook eenvoudiger. Bij C# is dit helaas niet het geval.

We hebben ook gebruik gemaakt van Mysql, deze enkel om een query uit te voeren.

Voor de connectie met de S3 bucket, RDS en de checksum op te halen maken we gebruik van javascript, dit hebben we gekozen omdat we hier de meeste documentatie over vonden.

**API**

We maken gebruik van Swagger.

Swagger is een soort template die de functies van onze .ASP-applicatie weergeeft, dit is handig indien de gebruiker een gestructureerde, overzichtelijke interface heeft van zijn code.

De gebruiker kan hierbij zijn eigen informatie ingeven (Files, Tekst, Getallen, afhangend op welke datatype de parameter van de functie is ingesteld). Met deze informatie wordt de functie uitgevoerd waar de gebruiker deze informatie heeft ingegeven. Het resultaat van deze functie wordt in een kader weergegeven onder het inputveld.

**Services**

We maken gebruik van:

* **Amazon S3**

Omdat we een opslagplaats willen voor onze bestanden. De functionaliteit van Amazon S3 (Opslagplaats voor objecten) is daarom het meest toepasselijke voor deze functie.

* **Amazon RDS**

We willen de informatie van onze bestanden behouden nadat de gebruiker deze upload. RDS is het meest geschikte om informatie te bewaren omdat het een database is.

* **Amazon Cognito**

We willen ervoor zorgen dat gebruikers zich moeten authentiseren, Cognito is precies voor deze doeleinden geschikt.

* **Amazon Lambda**

Met deze service kunnen we gebruik maken van talloze triggers om zo onze functies uit te voeren bij bepaalde gebeurtenissen of tijdstippen. Hier maken we gebruik van de trigger EventBridge, die zal nakijken of er objecten verwijderd zullen moeten worden.