AP HOGESCHOOL ANTWERPEN

AP.BE

ICT architecture

Analayse document

Opleiding: Toegepaste Informatica

Academiejaar: 2

El Moukhtari Atif

Proost Robin

Inhoud

1 Samenvatting

Er is ons gevraagd om een platform te bieden waar we een veilige manier kunnen aanbieden aan gebruikers om bestanden te delen met andere gebruikers. Sinds er deze service genoodzaakt is gaan we deze proberen bieden tegen de laagst mogelijke prijs. Deze service zal bestanden tot 24 uur bijhouden zodat de opslagkosten gelimiteerd blijven.

2 Situatie To-Be

2.1 Authenticatie

We willen niet dat iedereen toegang heeft tot deze service. Hiervoor willen we de personen die gebruik willen maken van de applicatie authenticeren. Dit doen we aan de hand van logingegevens. Om in te loggen hebben we een "Username" en "Password" nodig.

Wanneer deze correct zijn krijg je een token terug. Wanneer je een token hebt kan je de gateway contacteren.

2.2 Opslag

Bestanden dat worden geüpload steken we in een database. Wanneer een gebruiker een bestand uploadt heeft deze een "Time To Live" van 24 uur. Na 24 uur verloopt dit bestand. We doen dit om de opslagkosten te limiteren.

2.3 Bestanden uploaden

Wanneer een gebruiker is geauthenticeerd zal deze bestanden kunnen uploaden. Bij het uploaden van een bestand wordt er een UUID aangemaakt. Er zal ook een checksum worden bijgehouden zodat de gebruikers de downloads kunnen verifiëren. Deze informatie wordt in het bestand bijgehouden.

2.3.1 Nice to have

Bij het uploaden van een bestand kan de gebruiker kiezen hoeveel keer het bestand gedownload mag worden. Er is ook een optie om het bestand up te loaden zonder specifieke waarde, dan zal het bestand standaard niet gelimiteerd zijn op download.

Bij het uploaden van het bestand, kan de gebruiker ook kiezen om een extra wachtwoord in te stellen. Als een gebruiker dit bestand dan wilt downloaden, moet hij een dubbele beveiliging uitvoeren:

- De gebruiker moet de UUID kennen.
- De gebruiker moet het extra wachtwoord kennen.

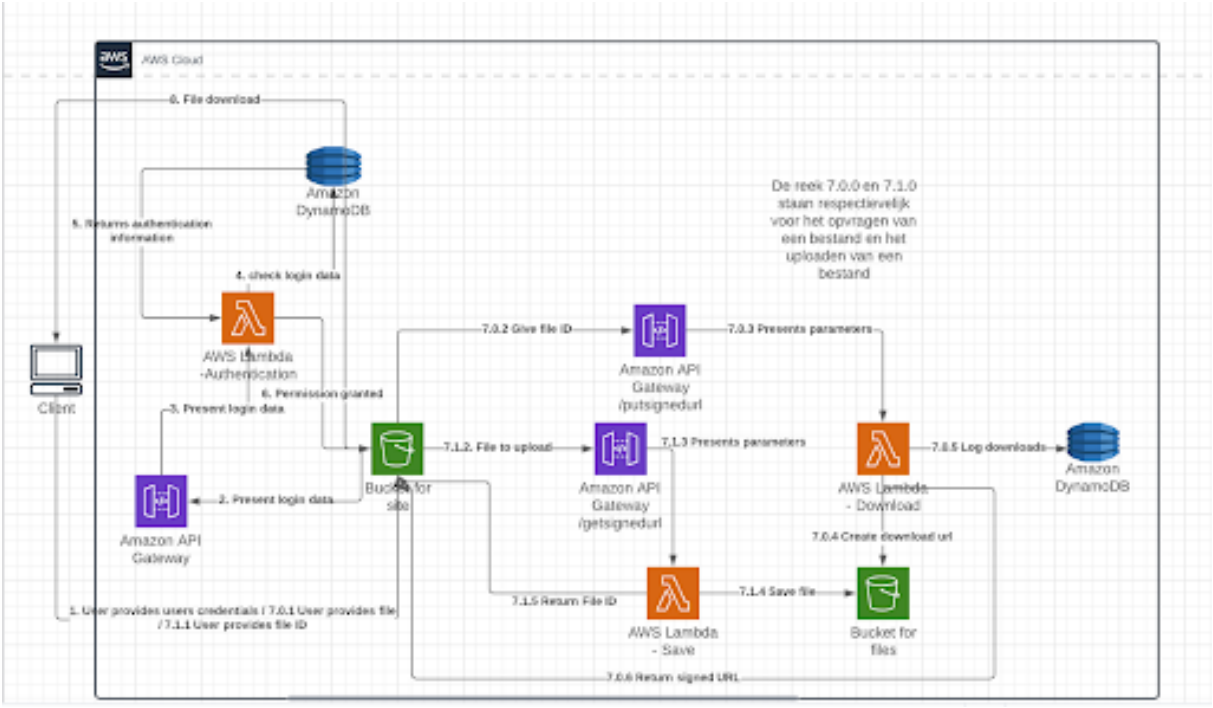
2.4 Bestanden downloaden

Zoals bij het uploaden zal de gebruiker eerst geauthenticeerd moeten zijn. Wanneer dit in orde is zal deze gebruiker de keuze hebben om bestanden te downloaden. Voor het downloaden van een bestand zal de gebruiker de UUID en mogelijks het wachtwoord moeten kennen. Wanneer de gebruiker de download uitvoert, dan zal deze gebruiker en het gedownloade bestand in een log komen. Zo houden we bij wie welke bestanden heeft gedownload e.d.

2.4.1 Logboek

Wanneer een gebruiker een bestand wilt downloaden, zal de gebruikersnaam van de gebruiker worden opgeslagen in een logboek. Deze lijst geeft aan welke gebruiker welke file heeft opgevraagd op welke dag en ook het uur.

3 Architectuur



Alles begint bij de client. De client zal eerst bij de site moeten inloggen bij de site. Deze site is gehost op een bucket. Deze inloggegevens zullen via een API en een lambda functie verwerkt en gecontroleerd worden. Hiervoor wordt ook een dynamoDB betrokken voor het opslaan van de wachtwoorden.

Als de user credentials zijn goedgekeurd door de lambda functie, heeft de gebruiker toegang tot de volledige applicatie. De gebruiker kan dan een file meegeven aan de site om deze up te loaden. De site zal deze dan aan de hand van een API en een lambda functie opslaan in een bucket. De ID van de file zal ook door de lamda functie worden teruggestuurd naar de gebruiker.

De andere functionaliteit die de gebruiker zal hebben is om files te downloaden. Dit kan hij doen door de ID van de file in te geven. Er zal dan een API call worden aangemaakt die de lambda functie zal triggeren. De lambda functie zal de webapplicatie een presigned url en een checksum leveren. Verder zal er ook worden bijgehouden wie deze file heeft gedownload en wanneer in de logboek. Uiteindelijk zal de file worden gedownload en krijgt de gebruiker het bestand en een checksum.

4 Verantwoording keuzes

4.1 S3 Bucket

We maken gebruik van een s3 Bucket. Hierop voeren we zowel “get” als “put” requests uit. We kunnen deze acties toelaten via een specifieke link(signed url) en alle andere calls weigeren. Zo zit er toch een beveiliging op. In de bucket worden de files ook verwijderd na 24 uur via een lifecycle rule. Verder hebben we ook aan de hand van de cors configuratie getracht ervoor te zorgen dat we enkel de eerder vernoemde “put” en “get” requests toelaten.

4.2 API gateway

De API gateway zorgt voor meer security. De client moet specifieke requests gebruiken om toegang te krijgen tot zaken die al ingesteld zijn.

4.3 Login

Om in te loggen, maken we gebruik van een lambda functie en dynamoDB . De gebruiker geeft zijn username en wachtwoord in en deze zullen worden gecontroleerd in de lambda functie. De reden waarom de inloggegevens in een aparte database worden opgeslagen is omdat de gevoelige credential gegevens dan niet op dezelfde niveau staan als de gegevens van de webapplicatie. Het is dus een veiligere manier van werken.

4.4 Upload

Voor de Upload functionaliteit hebben wij gekozen om te werken met een lamda, bucket en een API. De lambda voert functies uit bij het binnenkrijgen van een bestand. Deze lamda functie zal dan ook een UUID toekennen aan het nieuwe bestand en zal deze in een bucket opslaan. De gebruiker krijgt een antwoord als de operatie is geslaagd of niet.

Wanneer de gebruiker een bestand heeft gekozen kan hij op de "upload" knop drukken. Wanneer dit gebeurt zal de website een get-request sturen naar de API gateway die verbonden is met de lambda. Deze get request stuurt de website nog 2 parameters mee(bestandstype en contenttype van bestand). Door deze 2 parameters mee te sturen, kan het bestand met de juiste extensie worden opgeslagen en kan deze weer gedownload worden.

Daarna zal de gebruiker een UUID terugkrijgen deze heeft hij nodig om het bestand terug te downloaden. Er zal ook een url worden meegegeven die de webapplicatie zal nodig hebben om een put-request uit te voeren op de S3 bucket waar de file wordt opgeslagen. Hierna is heel het upload proces afgerond en zit de file in de bucket.

4.5 Download

Voor de download functionaliteit hebben wij gekozen om te werken met een API, een lambda functie en een dynamoDB database.

Bij het downloaden moet de gebruiker de ID van de file ingeven en moet hij dan op het download knop drukken. De api zal dan een lambda functie aanroepen. Deze lambda functie gebruikt de ID om de juiste file te zoeken in de bucket. De functie zal nu een signed URL meekrijgen die de gebruiker nodig heeft om het bestand te downloaden. Ook metadata als bestandextensie zal worden meegestuurd naar de gebruiker. De lambda functie zal ook naar de database moeten sturen wie de file heeft gedownload. Dit doet hij door een UUID te creëren en deze dan met de ID van de opgevraagde file te sturen naar de dynamoDB tabel.

De reden waarom wij hebben gekozen om met een API te werken is dat de lambda functie zo gemakkelijk kan worden getriggerd en ook omdat de API de juiste lamda functies op de juiste momenten aanspreekt. Verder werden de permissions gewijzigd zodat de lambda functies toegang hebben tot de dynamoDB en buckets.