**Kort Verslag: Infrastructuur, Technologie Stack, en Toekomstige Stappen**

**verder info kan je terug kijken op chatgtp  
*(setup Communication With)***

**Django Versie:**

4.2.12

**Database:**

PostgreSQL

**Django Apps:**

Gebruik voor API's en Backend

**Huidige State van de Services**

**Communicatie tussen Windows Services:**

Er is geen specifieke informatie over de communicatie tussen de Windows services.

**Logging en Monitoring:**

Bij errors wordt er een bericht verzonden.

**Security en Autorisatie  
Security Policies:**

Communicatie tussen service en de backend moet veilig zijn en alleen toegankelijk voor bevoegden.

**Gewenste API Autorisatie:**

Opties zoals OAuth, API keys, JWT tokens worden besproken.

**OAuth:** Dit zorgt ervoor dat elke klant een unieke token krijgt die ze moeten gebruiken om toegang te krijgen tot de API.

**JWT Tokens:** JWT (JSON Web Token) Authentication Gebruik maken van google / facebook inlog systeem.

**Toekomstige Functionaliteiten  
Configuratiebeheer:**

Er is behoefte aan endpoints voor het ophalen en bijwerken van configuraties. Windows services moeten configuraties bij opstarten en op gezette tijden ophalen.

**Licentiecontrole:**

Ideeën bespreken over licentiecontrole op basis van een licentiesleutel of andere methoden.

zoals een inlogsysteem voor extra beveiliging

(*Ja, je kunt per applicatie een aparte licentie toewijzen en ervoor zorgen dat alleen geautoriseerde gebruikers toegang hebben tot die applicatie. Dit kun je bereiken door een licentiemodel en autorisatiecontrole in je Django backend te implementeren.*)

**Vragen en Aanbevelingen  
Meerdere Auth Tokens per gebruiker:** Ja, dit is mogelijk in Django.

**Beveiliging van Backend API:** Gebruik token authentication, JWT, en rate limiting. Implementeer een inlogsysteem en beheer gebruikersrechten via het Django admin panel.

**Voorkomen van Openbare Registratie:** Gebruik handmatige goedkeuring door beheerders of uitnodigingscodes.

**Belangrijke Beveiligingsmaatregelen**- Gebruik HTTPS voor alle communicatie.

- Stel eisen voor sterke wachtwoorden.

- Implementeer rate limiting. ( beperken van acties )

- Beperk toegang tot gevoelige informatie.

- Beheer gebruikerssessies en stel korte sessieduur in.

- Verander standaard URL-patronen voor admin-interfaces.

- Gebruik Sentry voor monitoring.

**Sentry:** Sentry is a software monitoring tool that helps developers identify and fix code-related issues. From error tracking to performance monitoring, Sentry provides code-level observability that makes it easy to diagnose issues and learn continuously about your application code health.[ Kosten: 30,- euro per maand | Demo Mogelijk ]

**TOOLS**

**Django Rest Framework (DRF)**

Django Rest Framework is al een krachtig hulpmiddel om API's te bouwen in Django. Het biedt ingebouwde ondersteuning voor authenticatie, permissies, en serializers.

**Django-Allauth**

Django-Allauth is een set van applicaties voor het beheren van gebruikersregistratie, authenticatie, accountbeheer en sociale authenticatie (bijv. Google, Facebook).

**Django Rest Auth**

Django Rest Auth biedt een set van REST API endpoints voor Django Rest Framework om gebruikersregistratie, login/logout, password reset en social media authenticatie te beheren.

**Django-Oauth-Toolkit**

Django-Oauth-Toolkit biedt ondersteuning voor het gebruik van OAuth2 authenticatie.

**JWT Authentatie**

Gebruik JSON Web Tokens (JWT) voor een meer schaalbare oplossing voor authenticatie en autorisatie.

**Django Guardian**

Django Guardian biedt object level permissions, waarmee je gedetailleerde controle kunt hebben over de rechten van individuele objecten in je applicatie.

**Heroku Log Drains**

Heroku Log Drains kunnen je helpen bij het verzamelen en beheren van logs door ze naar een externe dienst zoals Papertrail te sturen voor verdere analyse en monitoring.

*( Door gebruik te maken van bovenstaande tools en libraries kun je de complexiteit van licentiebeheer, authenticatie en communicatie in je Django applicatie aanzienlijk verminderen. Dit biedt een meer modulaire en onderhoudbare codebasis. )*

**Aanbevolen Stappenplan voor Verbinding en Beveiliging**

1. **Configureer de Database:**

Instellen van database-instellingen in settings.py.

1. **API Endpoints in Django:**

Maak de benodigde API endpoints en gebruik Django REST framework.

1. **Beveiliging Implementeren:**

Installeer en configureer token authentication.

Gebruik HTTPS voor beveiligde communicatie.

1. **Windows Service Configuratie:**

Installeer benodigde Python packages.

Voeg functies en foutafhandeling toe aan de Windows services.

1. **Samenvatting:**

**Basisfunctionaliteit:** configureer database en API endpoints.

**Beveiliging:** voeg token authenticatie toe en gebruik HTTPS.

**Windows service configuratie:** zorg voor veilige gegevensoverdracht met robuuste foutafhandeling.

**Stappen om de Communicatie op te Zetten**

***API opzetten in Django:***

* *Installeer en configureer Django REST framework.*
* *Definieer API-endpoints voor licentiecontrole, logging ophalen, en configuraties uitlezen.*
* *Implementeer API autorisatie via OAuth2, JWT tokens, of API keys.*

***Beveiliging:***

* Zorg voor HTTPS-communicatie.
* Implementeer beveiligingsmaatregelen zoals rate limiting, logging, en monitoring.

**Aanpassen van de Windows Services:**

* Voeg functionaliteit toe voor communicatie met de Django backend.
* Implementeer authenticatie bij requests naar de backend.
* Voeg logging toe voor communicatie pogingen, successes en failures.

**Configuratiebeheer:**

* Ontwikkel endpoints in Django voor het beheren van configuraties.
* Pas Windows services aan om configuraties periodiek op te halen en toe te passen.

**Licentiecontrole:**

* Definieer een endpoint in Django voor licentiecontrole.
* Pas Windows services aan om de licentie te controleren bij opstarten en periodiek.

**Logging en Heartbeats:**

* Implementeer endpoints in Django voor het ontvangen van log data en heartbeats.
* Zorg dat Windows services periodiek hartslagen en belangrijke log data sturen.

**Meer informatie  
OAuth 2.0**OAuth 2.0 is een krachtige en flexibele authenticatie- en autorisatie standaard die vooral nuttig is wanneer je ondersteuning voor externe providers (bijv. Google, Facebook) wilt toevoegen of wanneer je gedetailleerde autorisatie nodig hebt.

**Voordelen:**

**Flexibel:** Ondersteunt verschillende grant types voor verschillende gebruiksscenario's.

**Veilig**: Scheidt authenticatie en autorisatie, en kan worden gebruikt met externe identiteitsproviders.

**Uitbreidbaar:** Kan worden uitgebreid met OAuth scopes en rollen voor fijnmazige toegangscontrole.

**Aanbevolen gebruik:**

Voor applicaties die ondersteuning nodig hebben voor sociale inlog of externe identiteitsproviders.

Voor complexe toepassingen die gedetailleerde autorisatie vereisen.

**API Keys**

API keys zijn eenvoudig te implementeren en kunnen een goede oplossing zijn voor machine-to-machine communicatie. Ze bieden echter niet hetzelfde beveiligingsniveau als JWT of OAuth 2.0, omdat ze eenvoudig kunnen worden onderschept en misbruikt.

**Voordelen:**

**Eenvoudig**: Gemakkelijk te implementeren en te gebruiken.

**Direct:** Geen extra stappen zoals token generatie of vernieuwing.

**Aanbevolen gebruik:**

* Voor eenvoudige toepassingen of interne API's.
* Voor machine-to-machine communicatie waar de beveiligingsrisico's laag zijn.