**Bachelor Thesis**

\_\_\_\_\_\_\_\_\_\_\_\_\_

Een grondige analyse van API-opties voor efficiënte en

veilige toegang tot klantgegevens in externe applicaties

|  |  |
| --- | --- |
| Bachelor | Toegepaste Informatica |
| Keuzetraject | Software Engineer |
| Academiejaar | 2023 - 2024 |
| Student | Lukas Olivier |

# Onderzoeksfase

Om een grondige vergelijking tussen verschillende API's mogelijk te maken, is het essentieel om een diepgaand onderzoek uit te voeren naar de diverse mogelijkheden. Met als doel een breed scala aan perspectieven te verkrijgen, streef ik ernaar een vergelijkbare demo-API te repliceren in verschillende programmeertalen. Voor elke programmeertaal zal ik op zoek gaan naar het meest populaire en/of effectieve framework om een API te ontwikkelen.

## Demo Project: Student Management API

Het demo-project omvat het beheer van studenteninformatie met CRUD-operaties, waaronder het toevoegen van nieuwe studenten, het ophalen van de lijst met studenten, het bijwerken van studenteninformatie en het verwijderen van studenten die de school hebben verlaten. De API wordt ontwikkeld in verschillende programmeertalen om een vergelijkende evaluatie mogelijk te maken.

## Node.js - Express

### Inleiding

De keuze voor Node.js in combinatie met het Express-framework is gebaseerd op de brede acceptatie van Node.js en de bewezen effectiviteit en eenvoud van Express voor het ontwikkelen van webapplicaties en API's. Node.js biedt een asynchrone, niet-blokkerende runtime, terwijl Express een lichtgewicht en flexibel webframework is dat goed aansluit op de Node.js-omgeving.

De combinatie van Node.js en Express biedt een moderne en efficiënte benadering van API-ontwikkeling, met een eenvoudige syntax en goede prestaties. Deze keuze is relevant voor implementatie in verschillende programmeertalen binnen het kader van dit onderzoek, gezien de populariteit en flexibiliteit van Node.js met het Express-framework.

## C# - ASP.NET Core

### Inleiding

De keuze voor C# en ASP.NET Core wordt ingegeven door de gevestigde positie van C# als een veelgebruikte programmeertaal en ASP.NET Core als een krachtig en veelzijdig framework voor web-API's. C# biedt sterke typen, een geavanceerd ontwerppatroon, en uitgebreide ondersteuning in de ontwikkelgemeenschap. ASP.NET Core, met zijn modulaire en cross-platform karakter, is ideaal voor het bouwen van schaalbare en moderne API's.

Deze combinatie van C# en ASP.NET Core biedt een robuuste basis voor API-ontwikkeling, met uitgebreide functionaliteiten en brede acceptatie in de industrie. Dit maakt het relevant voor de implementatie in verschillende programmeertalen binnen het kader van dit onderzoek.

## Python – Django

### Inleiding

De keuze voor Python in combinatie met het Django-framework is gebaseerd op de wijdverspreide adoptie van Python als programmeertaal en de reputatie van Django als een krachtig webframework. Python biedt een heldere syntax, uitgebreide bibliotheken en een actieve gemeenschap, terwijl Django zich onderscheidt door zijn elegantie en gebruiksgemak bij het ontwikkelen van webtoepassingen.

Deze combinatie van Python en Django vormt een solide basis voor de ontwikkeling van API's, met aanzienlijke flexibiliteit en efficiëntie. De keuze voor dit duo is relevant voor de implementatie in verschillende programmeertalen binnen het kader van dit onderzoek.

## Java - Spring Boot

### Inleiding

De keuze voor Java in combinatie met Spring Boot is gestoeld op de brede acceptatie van Java in de softwareontwikkelingsindustrie en de robuuste mogelijkheden van het Spring Boot-framework. Java staat bekend om zijn platformonafhankelijkheid, sterke typen en uitgebreide ecosysteem. Spring Boot, als een opvatting van het Spring-framework, biedt een vereenvoudigde manier om snel en efficiënt op Java gebaseerde API's te ontwikkelen.

De combinatie van Java en Spring Boot levert een solide basis voor de ontwikkeling van schaalbare en betrouwbare API's. Deze keuze is relevant voor implementatie in verschillende programmeertalen binnen het kader van dit onderzoek, gezien de veelzijdigheid en populariteit van Java en de efficiëntie van het Spring Boot-framework.

## ****Node.js -**** Apollo Server ****(****GraphQL****)****

### Inleiding

De keuze voor Node.js in combinatie met Apollo Server, dat is geoptimaliseerd voor GraphQL, is gebaseerd op de wijdverspreide adoptie van Node.js en de effectiviteit van Apollo Server voor het implementeren van GraphQL-API's. Node.js biedt een asynchrone, niet-blokkerende runtime en heeft een grote ontwikkelaarsgemeenschap. Apollo Server is een gespecialiseerd framework dat gericht is op het bouwen van efficiënte en flexibele GraphQL-API's.

Deze combinatie van Node.js en Apollo Server (GraphQL) biedt een moderne en krachtige benadering van API-ontwikkeling. Het gebruik van GraphQL als querytaal biedt flexibiliteit voor client-applicaties om specifieke gegevens op te vragen, waardoor het relevant is voor implementatie in verschillende programmeertalen binnen het kader van dit onderzoek.