Analyse

# Bedrijfscontext

## Gebruikers

Klanten kunnen via een portaal*(out of scope)* of API enkele acties uitvoeren.

* Bieding plaatsen
* Bieding aanpassen
* Alle biedingen van een auto opvragen

Medewerkers kunnen via een portaal*(out of scope)* of API nog meer acties uitvoeren.

* Bieding accepteren
  + Gekoppelde auto mag geen nieuwe biedingen meer kunnen ontvangen
* Bieding weigeren

## Schaalbaarheid

De applicatie moet schaalbaar zijn. De gebruikers moeten zo min mogelijk verschil merken – of er nu hoge of lage load op het systeem is.

# Systeem

## REST & Events

We kunnen de acties onderscheiden in twee groepen: sync en async. Gebruikers die data willen opvragen, verwachten direct hun antwoord. Gebruikers die nieuwe data plaatsen, verwachten niet meteen de aanpassing te zien.

* Create = Async
* Read = Sync
* Update = Sync

*Zie bijhorend schema voor architectuur*

## Schaalbaarheid & Load Balancing

Wanneer gebruikers hoge load maken met de async acties, zal de tijd toenemen voor de acties afgerond worden. Dit zal impact hebben op het opvragen van data (e.g. oude versie). De applicatie die de verwerking hiervan doet zal dus schaalbaar moeten zijn.

Voor de gebruikers zou de ‘huidige schaal’ van de applicatie niet mogen uitmaken. Zij gebruiken een bepaald endpoint en moeten dit blijven kunnen gebruiken.

# Technologie

## ASP.NET

Verantwoordelijk voor het portaal*(out of scope)* en de API-endpoints van de applicatie.

*Motivatie: ervaring…*

## MassTransit (RabbitMQ)

Verantwoordelijk voor de async acties m.b.v. queues. Belangrijkste voordelen van MassTransit over standaard RabbitMQ voor dit project:

* Error Handling en automatische Message Retries
* Handige (de-)serialisatie van messages

## SQL Server

Werkt zeer goed samen met EntityFramework. Andere overweging is PostgreSQL, maar persoonlijk minder ervaring mee wat voor tijdverlies gaat zorgen.

*Motivatie: ervaring…*

# Foutafhandeling en Monitoring

## Foutafhandeling

De fout die de gebruiker terugkrijgt moet duidelijk zijn voor zowel de sync als async acties. Voor de sync acties moet de gebruiker direct zien wat er misloopt en de feedback ontvangen. Voor de async acties moet er een basis-validatie gebeuren (e.g. geen negatieve bieding) – maar wat er nadien gebeurd is niet belangrijk.   
Als er in het achterliggend stuk iets misgaat, moet dit automatisch opgelost worden door het systeem.

## Monitoring

Door logging toe te voegen op nuttige punten doorheen de app zouden we elke request moeten kunnen volgen van begin tot einde.

Verschil in ernst moet ook duidelijk zijn – error als er iets niet verwerkt kan worden, warning als er iets misging maar herstelbaar, informatie voor basic tracing van request.

# Scenario’s

Alle requests moeten een basis-validatie hebben (e.g. minimum prijs, …). Indien deze zou falen, moet de juiste feedback aan de gebruiker gegeven worden.

## Bieding aanmaken

Wanneer een gebruiker een request maakt om een nieuwe bieding te plaatsen

* Request voldoet aan de validatie dus plaatst een bericht op de queue
* Gebruiker ontvangt een OK van de API, wat aangeeft dat de bieding ontvangen is
* Verwerking of eind-status van de bieding maakt voor de gebruiker niet uit, eindresultaat moet zijn: nieuwe bieding in de DB

## Bieding aanpassen

Wanneer een gebruiker een request maakt om een bieding aan te passen:

* Request voldoet aan de validatie dus zal de nieuwe data opslagen
* Geeft meteen een resultaat aan de gebruiker

## Biedingen opvragen

Wanneer een gebruiker een request maakt om de biedingen van een auto te zien:

* Geeft meteen het resultaat (= biedingen opgeslagen op dit moment) aan de gebruiker

## Bieding accepteren

Wanneer een medewerker een request maakt om een bieding te accepteren:

* Request voldoet aan de validatie dus zal de nieuwe data opslagen
* Verdere biedingen op de auto worden automatisch geweigerd
* Geeft meteen het resultaat aan de gebruiker

## Bieding weigeren

Wanneer een medewerker een request maakt om een bieding te weigeren:

* Request voldoet aan de validatie dus zal de nieuwe data opslagen
* Geeft meteen het resultaat aan de gebruiker