Keuzes:

# MessageQueue:

ActiveMQ:

Wij gebruiken Messagequeue voor als er een api crasht dat de data dan niet verloren gaat.

1. Een MessageQueue kan applicaties laten communiceren die verschillende talen en besturingssystemen hebben.
2. De locatie waarvan de berichten worden verstuurd wordt niet weergegeven door een MessageQueue.
3. De producent en consument hoeven beide niet tegelijkertijd beschikbaar te zijn om de berichten te ontvangen.
4. MessageQueue kun je heel gemakkelijk schalen met een groot aantal services
5. Asynchrone communicatie, De applicatie kan het bericht afgeven en vervolgens verder gaan in plaats van wachten op een antwoord.
6. Een service kan details over zichzelf wijzigen zonder de klant te beinvloeden.
7. Als een service verloren gaat gaan deze requests niet verloren.

# NoSQL:

MongoDB:

Flexiebel te combineren met het maken van een spel, schaalbaar

1. Snelle queries, naar mate de database en de tabellen groter worden, wordt een sql query steends langzamer, bij een nosql database merk je minder verschil.
2. Scaling, een nosql database stopt alles in een bestand zodat het compact blijft terwijl een sql database al snel een flinke server nodig heeft.
3. Flexiebele documentschema’s, alle data hoeft niet overeen te komen die in 1 bestand staat, zo crasht de applicatie niet als je iets verkeerd toevoegd.
4. Verandervriendelijk ontwerp, als er een onderdeel aan een object wordt toegevoegd is dit makkelijker te vervangen in een nosql database

# Angular:

FrontEnd

Overzichtelijke javascript framework, de applicatie hoeft maar 1 keer te laden.

1. Herbruikbaarheid De op componenten gebaseerde structuur van Angular maakt de componenten zeer herbruikbaar in de hele app. U kunt de UI (User Interface) bouwen met bewegende onderdelen, terwijl u ook zorgt voor een soepel ontwikkelingsproces voor ontwikkelaars.

2. Vereenvoudigde unit-tests Omdat ze onafhankelijk van elkaar zijn, maken de componenten het testen van eenheden veel gemakkelijker.

3. Verbeterde leesbaarheid Consistentie in codering maakt het lezen van de code een fluitje van een cent voor nieuwe ontwikkelaars van een lopend project, wat bijdraagt ​​aan hun productiviteit.

4. Gemak van onderhoud Ontkoppelde componenten zijn vervangbaar door betere implementaties. Simpel gezegd, het maakt efficiënt codeonderhoud en -update mogelijk.

# Gateway:

Ocelot:

Geschreven in C#, snel en klein programma met veel tools

1. Linkbaar met zowel C# en Java
2. Snelle gateway die weinig power verbruikt
3. Schaalbaar
4. Automatische authenticatie en routing
5. Deploybaar met docker

# Discovery service

Consul

Werkend in combinatie met ocelot gateway, relatief licht en snel en vol handige tools.

1. Service discovery
2. Health checks
3. Load balancing
4. Service graphs
5. Identity enforcement
6. Distributed service configuration management
7. Configuratie met C# en Java
8. <https://codingbee.net/consul/what-are-the-advantages-of-using-hashicorp-consul>

# Security testing

Owasp zap

1. Passive en active mode.
2. Intergreerbaar binnen een cicd pipeline
3. Print testrapport uit

# Code review

Sonarqube

1. Print testrapport uit
2. Code coverage
3. Configureerbaar voor zowel code smells als serieuze problemen