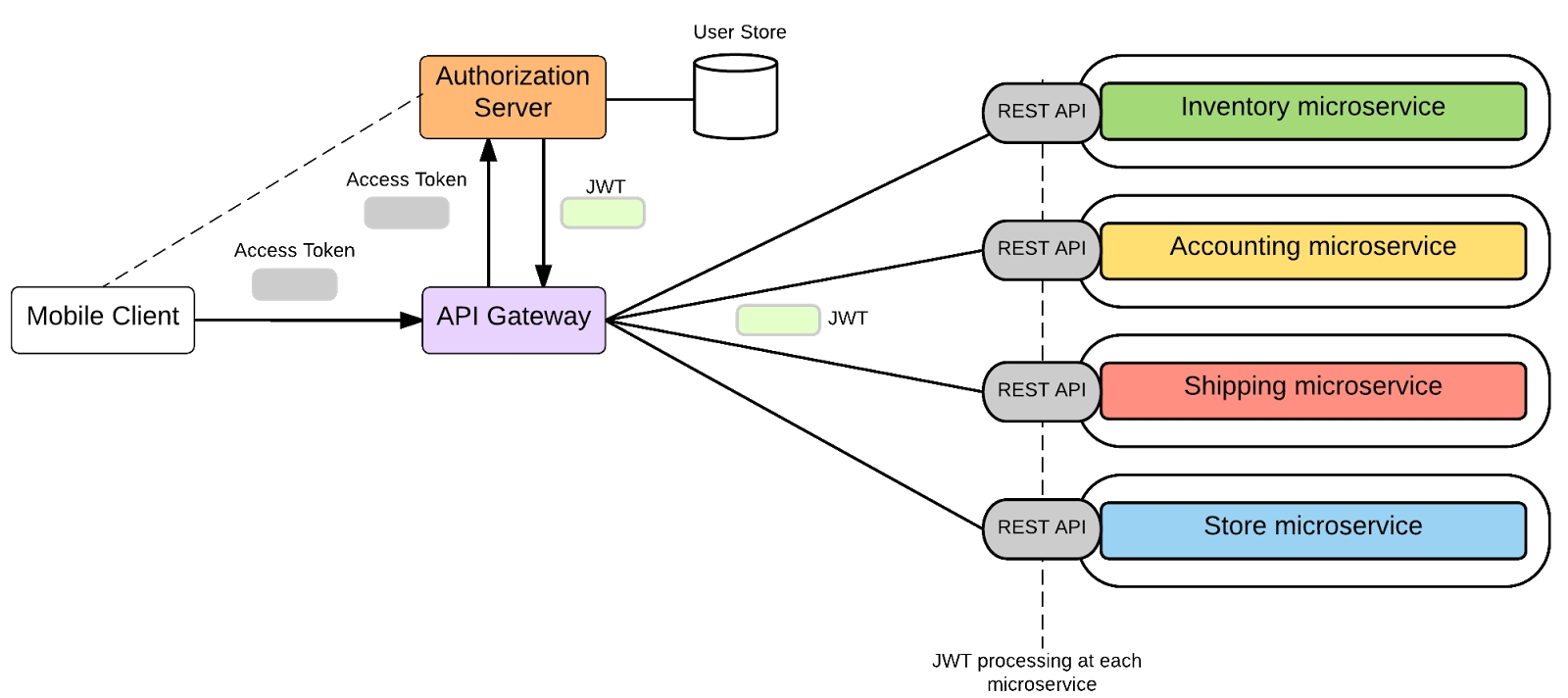
# Microservice arkitektur

Af: Anders

## Hvad er microservice?

Microservice er en software arkitektur som består af en eller flere små applikationer som for det meste vil være implementeret i skyen som eksempelvis en virtuel maskine. Ved at de køre på en virtuel maskine giver det bedre mulighed for at administrere hvor meget CPU samt hukommelse der bruges af servicen. Microservice arkitektur bruges til at lave enkelte applikationer som passer til andre servicer. Hver microservice vil normalt have deres egen database frem for en stor samlet.  
For at der kan kommunikeres mellem de andre services bruges der API’er.



## Hvem bruger microservice?

Hvis du har en stor applikation der har behov for at være hurtig, kan det være en fordel at bruge microservice, da der lettere kan arbejdes på flere områder indenfor applikationen, samt mindre risiko for at applikationen crasher da den er delt op i mange små uafhængige services i stedet for en stor. Det giver bedre mulighed for test og opdatering uden behov for at lukke resten af applikationen ned.

Fx Netflix benytter sig af microservices for at sikre at de holder hastigheden i top. I stedet for der kun er en service der skal håndtere de flere millioner request fra flere forskellige enheder hver dag. Sørger microservices for at dele arbejdet op, så de forskellige microservices specialisere sig inden for et bestemt område som så til sidst for sendt filmen frem til brugeren.

## Fordele ved at bruge microservice:

Ved brugen af microservice vil du som udvikler arbejde i mindre fokuseret grupper da der kun skal fokuseres på en service. Det kan også have effekt på koden da der er betydeligt mindre end hvis du stod med en almindelig service, der er mindre kode at sætte sig ind i når det skal implementeres.

Hvis en service går ned betyder det at det kun er den ene service der ikke virker mere, da en service er uafhængig af alle andre services.

Det giver mulighed for at opdatere en service uden at man er nød til at ligge hele applikationen ned, og hvis der under opdateringen skulle ske nogle bugs eller andre mindre fejl, vil det betyde at det stadig kun er den ene del der ikke er aktiv og lettere at fokusere på at få det fikset.

## Ulemper ved at bruge microservice:

Det skal sikres at den microservice du laver ikke kan være skyld i crash fordi din microservice er afhængig af andre services. Hvis f.eks. en anden microservice er blevet opdateret kunne det ske at din service ikke er kompatibelt mere, så det kræver et godt design for at undgå crash.

Du kan risikere at hver microservice er skrevet i et nyt sprog eller med nyt framework, hvilket kan være med til at gøre det svært at opretholde servicen. Derfor kan det være en fordel der er sat nogle få regler som man skal holde sig indenfor, det vil dog være i strid mod at hvert udviklingsteam selv bestemmer deres arbejdsmetode.

## Microservice med Budgetmanager

Det vil ikke give meget mening at bruger microservices i vores budgetmanager da der ikke er behov for splitte arbejdet ud over flere services og processere. Det vil sagtens kunne laves som microservice, men vi har valgt at Client-Server arkitekturen passer bedre med budgetmanager. Der er heller ikke behov for flere forskellige databaser, som vi ville få med microservices hvor vi med Client-Server også bare vil have en database.