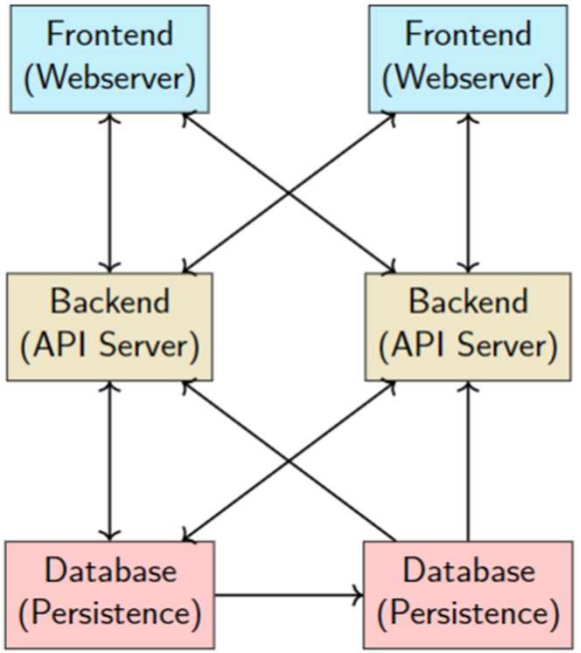
Operativ Systemer 8

Lavet af: Vivek Misra

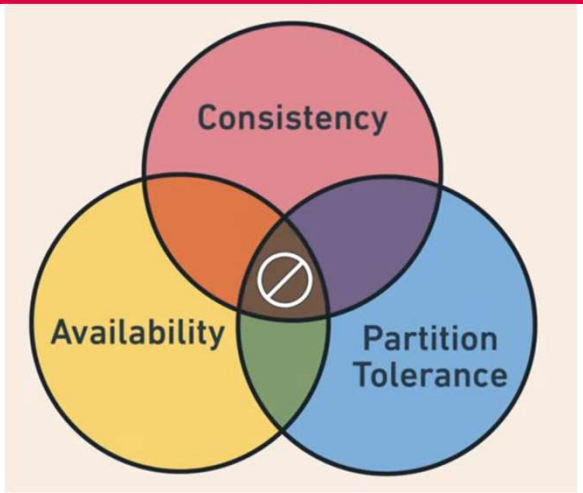
Database Centerede Arkitektur

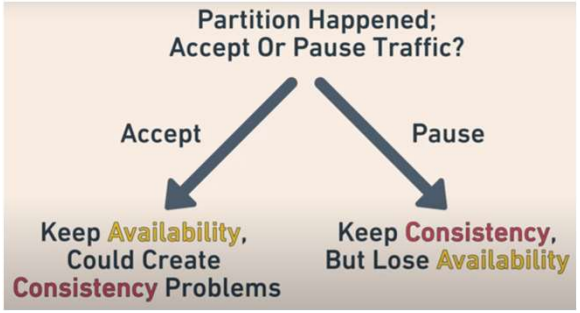
- Denne form for Arkitektur fungerer som en sikker og skalerbar repository for at gemme data, mens frontend og backend komponenterne arbejder sammen for at holde styr på præsentationen, interaktionen og processering af Database Centerede Arkitektur.
  - **Frontend: Denne del referer sig til den grafisk overblik som vi kan visualisere på skærmen. Vi kan se, at Frontenden interagerer med Backenden, hvor den sender Request og fremviser det data som der er anmodet om.**
  - **Backend: Det er denne del som processer ”requesten” der er sendt fra Frontenden. Backenden processerer dataerne nede i Databasen og sender den op mod Frontenden.**
  - **Databasen: Den sørger for, at gemme alle de nødvendige informationer og data som vi har. Når Backenden afhenter dataerne herfra, så er det Databasens arbejde i at kunne give dataerne videre til Backenden.**



CAP-Teorien

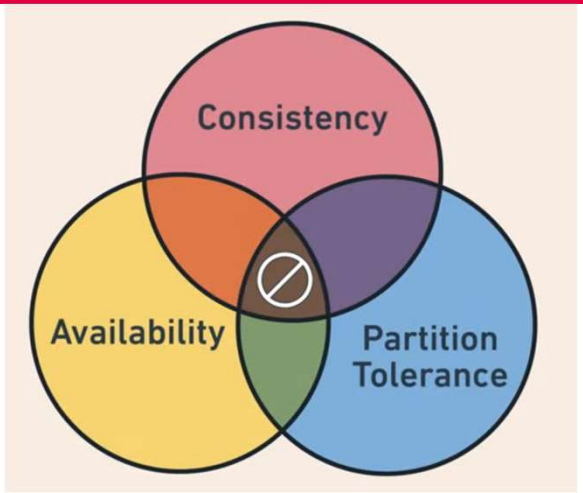
- CAP-Teorien går ud på, at man ønsker at opnå Consistency, Availability og Partition Tolerance i en Distribuerede Computersystem.
  - **Consistency: Dette går ud på, at systemet returnerer det information som den er blevet bedt om at skrive, og er altid opdateret.**
  - **Availability: Dette går ud på, at systemet altid er tilgængelig ift. Opskrivning og ændringer på distribuerede systemer.**
  - **Partition Tolerance: Når vi har forskellige systemer, der pludselig bliver opdelt og er adskilt fra hinanden. Så skal systemer være i stand til at kunne fungerer videre uden forhindringer.**
- CAP-Teorien er meget bekendt, men det er ikke altid muligt at opnå de 3 værdier som er blevet nævnt.
- Det er kun muligt at opnå 2 ud af de 3 værdier i en system. Årsagen er begrundet til højre på billedet!

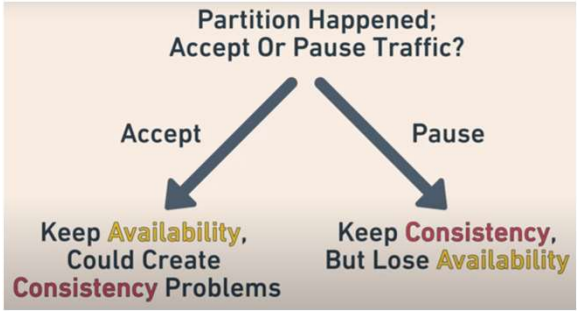




CAP-Teorien

- Her forklarer når to af værdierne er blevet opfyldte.
- **Consistency & Availability:** Hvis der er ingen Netværksopdeling, så kan det garanteres at der er consistency og tilgængelighed. Dette betyder, at vores system kører normalt i up to date form, men også med god tilgængelighed.
- **Consistency & Partition Tolerance:** Hvis der er en Netværksopdeling, så kan det ses at Systemet opskriver det som den er blevet bedt om at gøre, men systemet er ikke altid tilgængeligt. Fordi hvis Partition Tolerance fjernes, så kan det ses at der pludselig er slåskamp over hvem der har den seneste information.
- **Availability & Partition Tolerance:** Hvis systemet stopper med, at modtage information eller trafik. Så er consistency forsvundet, men availability er bevaret fordi systemet er tilgængelig til at svare i Netværksopdelingen.





ACID Egenskaber af Transaktioner

- Når vi snakker om Transaktioner af Databaser, så benytter vi os af begrebet ACID:
- ACID består af begreberne: Atomicity, Consistency, Isolation og Durability.
  - **Atomicity:** Hvis der sker noget nedbrud i systemet, som gør at transaktionen bliver forhindret i dets udførsel. Så går Atomicity ind og beslutter om Transaktionen skal eksekveres igen eller så skal den eksekveres fuldt ud.
  - **Conistency:** Dette betyder, at vores Transaktion skal være up-to-date. Eksempelvis, hvis der er trukket 400 kroner fra bank-kontoen. Så skal der stå -400 eller den beløb som resterer fra de 400.
  - **Isolation:** Det betyder, at Transaktionen bliver eksekveret alene. Det indebærer også, at en transaktion må ikke kigger over på hvad der foregår i de andre transaktioner. Samtidig med, at de transaktionerne må ikke påvirke hinanden.
  - **Durability:** Dataerne som bliver opdateret i systemet, og permanent gemmes i deres opdaterede form er kaldt for Durability. Dataer som opdateres, men gemmes midlertidigt er IKKE kaldt for Durability.

Kvalitetsattributter

- Når vi snakker om Kvalitetsattributter, så plejer vi at henvende os til de begreber som står forneden:
- **Availability:** Det betyder, at systemet skal være tilgængelig hele tiden. Her skal systemet være i stand til at kunne fikse en fejl, hvis det opstår.
- **Deployability:** Det betyder, at vi har det nemt med at overføre vores Udvikling til Produktion.
- **Energi Efficency:** Det betyder, at det system som der udvikles skal være designet således at det ikke opkræver meget energi.
- **Modifiability:** Det er der, hvor det er muligt at kunne skabe ændringer i systemet. Især små ændringer, så det ikke skaber problemer.
- **Performance:** Det er der, hvor vi udfører arbejder indenfor et specifik tidspunkt som kræves af os.
- **Safety:**
- **Security:**
- **Testability:**

Kvalitetsattributter

- Når vi snakker om Kvalitetsattributter, så plejer vi at henvende os til de begreber som står forneden:
- **Safety:** Det er der, hvor man kan få systemet til at undgå at komme i problemer eller skade.
- **Security:** Det er lidt henad ad Safety, men her forsøger vi detekterer ting således at vi er parate til at bekæmpe imod problemet. Vi beskytter herhenne det data som vi har, for fremmede invasioner.
- **Testability:** Det skal være nemt for systemet, at blive testet. På denne måde, skal det være muligt at opdage fejl i systemet og dermed skabe ændringer i forhold til det.

SLUT 8

Lavet af: Vivek Misra