# Trunk-based branching strategie

Trunk-based branching is een strategie voor versiebeheer waarbij ontwikkelaars aan korte branches werken en wijzigingen regelmatig in de main-branch worden geïntegreerd. Deze benadering is anders dan met feature-bracning of release-branching, waarbij de ontwikkeling gedurende een langere periode afzonderlijk plaatsvindt voordat deze gemerged wordt in de main-branch. Voor deze opdracht heeft trunk-based branching een aantal voordelen:

*Vroegtijdige detectie van integratieproblemen:*
Regelmatige integratie maakt vroegtijdige detectie van integratieproblemen mogelijk. Als er conflicten of problemen zijn met de main-branch, worden deze vroeg of laat geïdentificeerd, waardoor het gemakkelijker wordt om deze aan te pakken en op te lossen.

*Continue integratie/continue levering (CI/CD):*
Trunk-based ontwikkeling sluit goed aan bij CI/CD-praktijken wat nu tijdens onze hoorcolleges ook aan bod komt. Omdat veranderingen voortdurend worden geïntegreerd, wordt het gebruik van geautomatiseerde test- en implementatiepijplijnen (zoals Jenkins) weer makkelijker gemaakt, wat leidt tot snellere feedback en deployment.

*Lean en Agile-principes:*
Trunk-based ontwikkeling is afgestemd op lean- en agile principes, waarbij de nadruk wordt gelegd op het snel leveren van waarde aan gebruikers. Het ondersteunt een incrementele en iteratieve ontwikkelingsaanpak.

*Gemakkelijker terugdraaien:*
Als er een probleem wordt gedetecteerd in de productie, is het vaak gemakkelijker om de wijzigingen te identificeren en terug te draaien, omdat de wijzigingen kleiner zijn en de integratie vaak heeft plaatsgevonden.

Ik zie deze aanpak verder voor mij in de zin dat elke issue of feature een aparte branche kan zijn die meteen geïntegreerd wordt in de main wanneer deze af is. Dan kan deze meteen door Jenkins getest worden en wanneer hier fouten zijn kunnen deze worden aangepast. Op deze manier blijft de main branch altijd in een bruikbare staat.