# Opdrachten

1. Zorg dat je een cargo aan kunt maken
   1. Open CargoController.cs in namespace Web.Controllers
   2. Ga naar de methode **public ActionResult Create(Cargo model)**
   3. Onder het commentaar “**// Create new cargo command**” maak je een nieuw CreateNewCargoCommand aan en voer je deze uit
   4. Open CreateNewCargoCommandExecutor.cs in namespace CommandExecutors
   5. Maak een nieuw Cargo object aan roep de Accept() methode van context aan
   6. Open Cargo.cs in namespace Domain
   7. Voeg onderaan een constructor toe die alle properties van het CreateNewCargoCommand accepteerd als parameters
   8. Maak in deze constructor een nieuw CargoCreatedEvent event aan en vul hiervan de properties. Gebruik vervolgens de ApplyEvent(..) methode om het event uit te voeren.
   9. Open CargoCreatedDenormalizer.cs in namespace ReadModel.Denormalizers
   10. Maak een nieuw ReadModel.Cargo object aan, vul de properties en sla deze op in de database
   11. Test het resultaat door een nieuwe cargo aan te maken
2. Valideer voor het aanmaken van een voyage dat de arrival date ná de departure date ligt
   1. Open VoyageController.cs in namespace Web.Controllers
   2. Roep een nieuwe methode GuardThatArrivalDateIsAfterDeparturetDate aan die controleert of arrival date na departure date ligt. Zo niet, dan wordt er een ArrivalDateBeforeDepartureDateException gegooid
3. Ken een cargo toe aan een voyage
   1. Open CargoController.cs in namespace Web.Controllers
   2. Ga naar de methode **public ActionResult Create(Cargo model)**
   3. Onder het commentaar “**// Add cargo to voyage command**” maak je een nieuw AddCargoToVoyageCommand object aan en voer je deze uit
   4. Open AddCargoToVoyageCommandExecutor.cs in namespace CommandExecutors
   5. Haal de voyage en cargo op met de methode context.GetById<..>(..)
   6. Voeg de cargo aan de voyage toe met <voyage object>.Add(..) en roep de Accept() methode van context aan
   7. Open Voyage.cs in namespace Domain
   8. Ga naar de methode “**public void AddCargo(Cargo cargo)**”
   9. Maak een nieuw CargoAddedToVoyageEvent aan en vul hiervan de properties. Gebruik vervolgens de ApplyEvent(..) methode om het event uit te voeren.
   10. Open CargoAddedToVoyageUpdateCapacityDenormalizer.cs in namespace ReadModel.Denormalizers
   11. Haal het juiste voyage object op aan de hand van e.VoyageId en update CapacityLeft en CapacityUsed
   12. Open CargoAddedToVoyageUpdateCargoDenormalizer.cs in namespace ReadModel.Denormalizers
   13. Haal het juiste cargo object op aan de hand van e.CargoId en update VoyageId, DepartureDate en ArrivalDate
4. Valideer dat er bij het toekennen van een cargo aan een voyage nog genoeg capaciteit is
   1. Open Voyage.cs in namespace Domain
   2. Ga naar de methode “**public void AddCargo(Cargo cargo)**”
   3. Roep de methode GuardThatThereIsEnoughRoomLeft aan die controleert of een voyage nog genoeg capaciteit heeft voor de cargo. Zo niet, dan wordt er een NotEnoughtCapacityLeftForCargoException gegooid