# 1. [Figur 1] Use case model

\* Vigtig, første model, især god ved uklart problemområde.

\* Frembringer aktører (primære, sekundære), kandidatklasser og associationer mellem disse.

\* Leder til domænemodel (struktur) og designmodel (adfærd).

# 2. [DEMO] Demonstrer website, start med carportforespørgsel

\* Kan oprettes af anonym og indlogget bruger.

\* Bruger oprettes ikke pt.

# 3. [DEMO] Færdiggør carportforespørgslen, vis også validering af brugerinput.

\* serverside validering i v2 – f.eks. hvis browser ikke understøtter html5/js validering.

# 4. [DEMO] Log ind for at se tegning og stykliste. Fortæl om rank.

\* 2 samtidige nedgraderinger kræver lås/synchronized metodesignatur for at undgå race condition. (2 igennem samme dør samtidig).

# 5. [Figur 2] Sekvensdiagram for opret forespørgsel og bruger i samme forretningsgang.

\* Først oprettes bruger, siden carport.

\* Alternativt flow hvis eksisterede bruger skal opdateres.

# 6. [Figur 3] Fortæl om håndtering af FogException i views.

\* Exceptions fanges f.eks. i DAO eller beregnere ”dybt” nede i arkitekturen.

\* Omdannes til FogException som FrontController forventer i sidste ende.

\* [DEMO] Prøv med forespørgsler nederst i listen => Tagtype/hældning giver FogException.

# 7. [Figur 4] Løsningsmodel for at undgå indbyrdes afhængigheder i commands .

\* PRG afhjælper også dobbelt-post problem.

\* Logik i FC som afgør om bruger må køre et command.

\* Logik i command som kaster exception hvis bruger ikke må køre det.

# 8. [Figur 5] Trådede beregnere.

\* Vis diagram / kode.

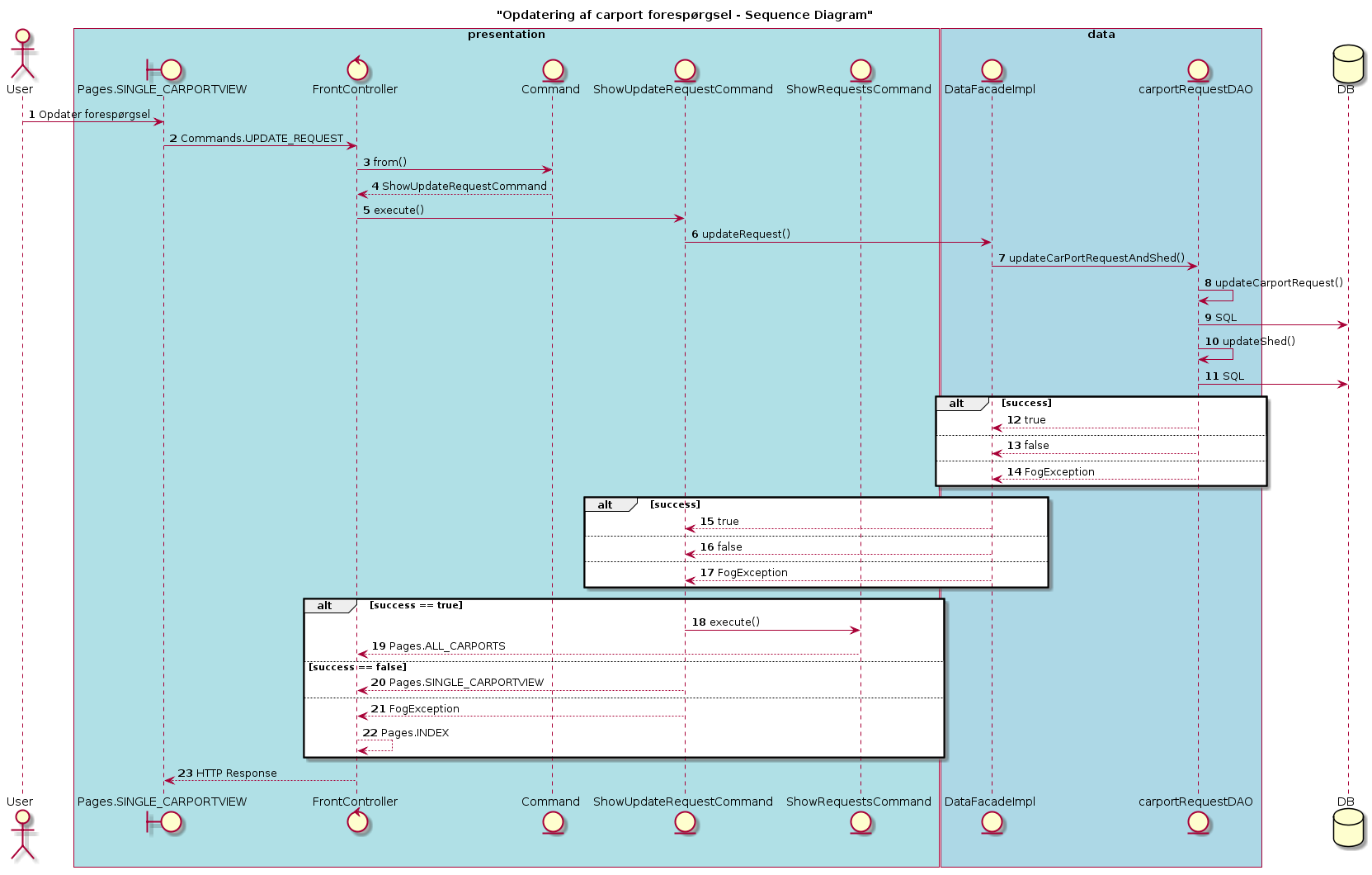
\* Forklar hvorfor Ajax/XHR ville være bedre.

Figur 1 – use case diagram

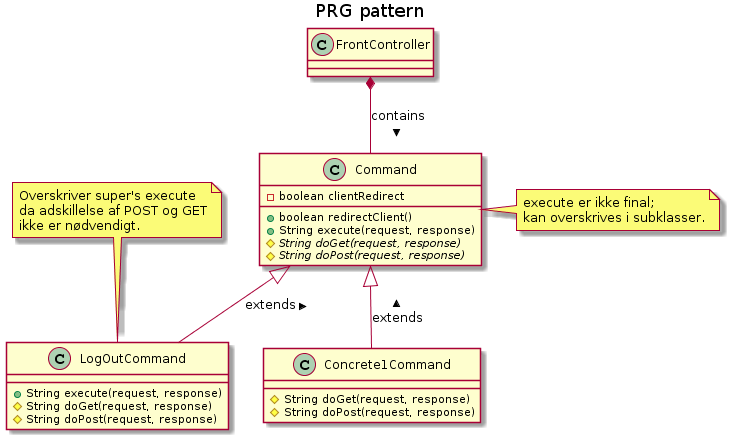


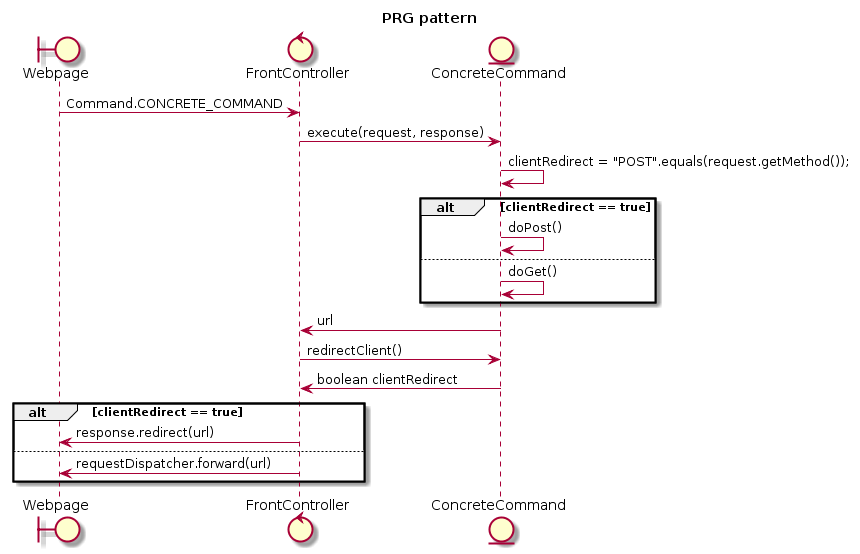
Figur 2 – Oprettelse af bruger OG carport.

Figur 3 – Opdatering af carport forespørgsel – sekvensdiagram.



Figur 4 – PRG pattern arkitektur og sekvens





Figur 5 – Trådede beregnere.

